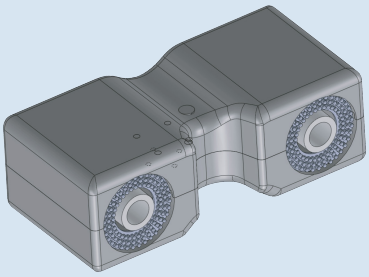
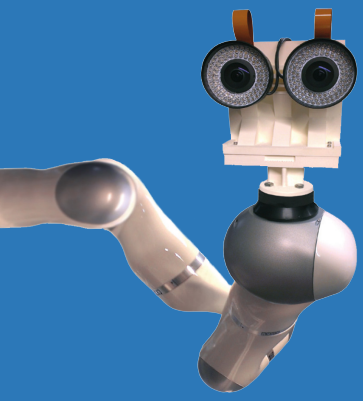
Computer Aided Medical Procedures
Prof. Dr. Nassir Navab

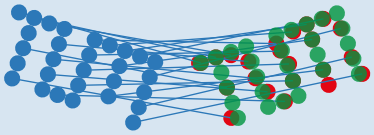


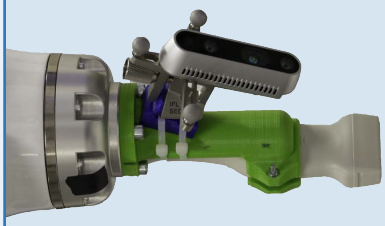


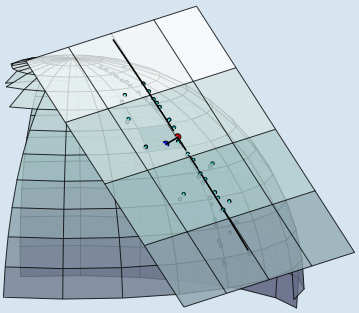
Dissertation

High Performance Visual Pose Computation

Benjamin Busam







Fakultät für Informatik
Technische Universität München

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Informatikanwendungen in der Medizin

High Performance Visual Pose Computation

Benjamin Emanuel Busam

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender: Hon.-Prof. Dr. Carsten Steger

Prüfer der Dissertation: 1. Univ.-Prof. Dr. Nassir Navab

2. Univ.-Prof. Dr. Markus Ulrich

Die Dissertation wurde am 10.02.2021 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 02.07.2021 angenommen.

Benjamin Busam
High Performance Visual Pose Computation
Dissertation, Version 1.1

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Informatikanwendungen in der Medizin
Boltzmannstraße 3
85748 Garching bei München

Abstract

To enable reliable and safe human-machine interactions, an automatic analysis of the surrounding environment by the machine is key. At the core of these geometric interpretations based on sensor signals lies the ability to estimate the rotation and translation of objects in space, and computer vision allows to realize such systems with image data. This thesis focuses on high performance 6D pose pipelines leveraging visual sensing to formulate algorithms that allow us to put solutions into practice and enable efficient pose computation methods.

We investigate an image processing pipeline for sub-pixel precise ellipse detection that meets the accuracy requirements of industry and the reliability needs for medical applications while being able to run in real-time. The algorithm is used for camera calibration and marker triangulation with a binocular stereo system. We propose and implement an optical tracking system both in hardware and software. The focus of our considerations is an optical pose computation algorithm that precisely determines the pose of an object marked with small self-adhesive, retro-reflective circular markers that generate an adaptable object point cloud. Efficient methods such as ICP can determine poses between point clouds, however, are sensitive to initialization and fail on sparse and corrupt input. To overcome these downsides, we formulate sparse point cloud matching as an energy optimization problem and reshape it from a probabilistic perspective. This allows to design an efficient yet robust solver based on dual quaternion 6D pose parametrization which we extend to a pose tracker.

Outside-in tracking suffers from line of sight restriction if an occluder appears in the field of view. We miniaturize the system and fix it on the object by inverting the camera view towards the surrounding environment. Utilizing advances in SLAM literature, we establish a markerless inside-out stereo method which demonstrates its benefits in rotational accuracy over marker-based outside-in tracking. Further reducing constraints, we investigate the ill-posed problem of marker-free monocular pose estimation. Many previous methods either treat the problem as a regression task or discretize pose space. They use convolutional neural networks and train one model per object. We reformulate this paradigm and look at the problem as an action decision process where the next best pose is determined using a render-and-compare strategy. It turns out that this simpler task can be solved reliably with a lightweight neural architecture that is instance agnostic such that our pose computation generalizes to unseen objects. Temporal considerations accelerate the process and allow for dynamic complexity reductions.

Finally, we apply our 6D pose estimation results for the task of sensor fusion. Concepts from differential geometry allow for pragmatic pose modifications and improve computations in presence of noise. We exemplify the practical impact of the developed pipelines in three orthogonal use case scenarios for industrial manufacturing, mobile augmented reality and cooperative medical robotics where multiple modalities are spatially fused.

Zusammenfassung

Automatische Umgebungsanalyse ist ein Schlüsselkonzept verlässlicher Mensch-Maschine Interaktion. Die Fähigkeit zur Bestimmung von Objektrotation und -translation auf Basis von Sensordaten ist hierbei eine Kerngröße, die mit Hilfe von maschinellem Sehen und der Auswertung von Bilddaten realisiert werden kann. Die vorliegende Dissertation beschäftigt sich mit Hochleistungs-Algorithmen zur 6D Posenbestimmung durch visuelle Sensorinputs, die erlauben Lösungswege zu definieren und effiziente Berechnungen praktisch umzusetzen.

Wir untersuchen zunächst ein Bildverarbeitungskonzept zur subpixel-genauen Ellipsendetektion, welches sowohl den Genauigkeitsansprüchen der Industrie als auch den Verlässlichkeitsansprüchen der Medizin genügt. Neben Kamera-Kalibrierung wird der Algorithmus zur Marker-Triangulation in einem Stereo-Kamera-Verbund benutzt. Hierfür wird ein optisches Tracking System entwickelt, welches die Grundlage für einen Posen-Algorithmus bietet. Objektposen werden präzise anhand von kleinen, selbstklebenden und retro-reflektierenden Markern bestimmt, die sich veränderliche Punktwolken erzeugen. Effiziente Methoden wie ICP können zwar Posen zwischen Punktwolken bestimmen, sind jedoch stark abhängig von Initialisierung, Messfehlern und Punktanzahl. Um diese Probleme zu eliminieren, formulieren wir ein Optimierungsproblem, welches wir in einer probabilistischen Relaxierung lösen. Dazu verwenden wir einen effizienten und robusten Ansatz, und parametrisieren Posen als duale Quaternionen in zeitlichen Video Sequenzen.

Im Falle von Verdeckung im Sichtfeld kann ein solches outside-in Tracking ausfallen. Um diese Problematik zu adressieren, miniaturisieren wir das System und fixieren es mit in den Raum gerichtetem Sichtfeld am Objekt. SLAM-Algorithmen helfen hierbei, ein markerloses inside-out Stereo-System umzusetzen, welches bezüglich seiner Rotationsgenauigkeit dem marker-basierten outside-in Tracker überlegen ist. Darüber hinaus untersuchen wir Lösungsansätze zur marker-freien Posenbestimmung aus einem einzelnen Bild. Viele Methoden betrachten das Problem entweder als Regressionsaufgabe oder diskrete Klassifikation. Üblicherweise werden CNNs eingesetzt und ein Netz pro Objekt trainiert. Wir reformulieren dieses Paradigma, indem wir das Problem als Aktions-Entscheidungs-Prozess betrachten und die nächstbeste Pose bestimmen. Es stellt sich heraus, dass diese einfachere Frage verlässlich mit einem Netzwerk niedriger Komplexität gelöst werden kann, welches zudem Instanz-agnostisch ist, also dessen Posen sich auf unbekannte Objekte übertragen lassen. Darüber hinaus lässt sich der Komplexitätsgrad mittels Video-Input dynamisch reduzieren und der gesamte Prozess beschleunigen.

Schließlich wenden wir die entwickelten Konzepte in der Praxis an, wobei Konzepte aus der Differentialgeometrie helfen, die Posen zu verbessern. Wir illustrieren die praktische Bedeutung der entwickelten Algorithmen anhand von drei Anwendungsfällen aus industrieller Fertigung, mobiler Augmented Reality und kooperativer Medizinrobotik.

Thank You

After spending some years in the research domain of 3D computer vision, I am still thrilled by its problems and super curious to answer more of the open questions in the field. The main reason for this is that I was lucky enough to work with many talented people from whom I could learn and get infected by their enthusiasm while we were collaborating.

Working on an interdisciplinary topic at the intersection of computer science, mathematics and medicine, I had the honour to be supervised by two great researchers. Thank you Prof. Dr. Nassir Navab and Prof. Dr. Ulrich Bauer for guiding me, taking the time for discussions, and giving me the freedom to explore the problems I considered exciting. Nassir, you are literally the incarnation of a "Doktorvater". Not only where you there when I had tough questions often driven by industry needs, but also did you give the room for me to become an independent researcher, and you always trusted in my decisions. Uli, I value all our discussions, your inspirations from the theoretical side and the balancing act of co-supervising Master students in mathematics from which I always requested also a practical outcome.

Thank you Dr. Simon Che'Rose for teaching me the art of industrial research and development even in difficult situations as a mentor and for giving me the chance to lead the research activities at FRAMOS. While this was sometimes a challenge, I learned a lot from it and am grateful for this experience. I also want to thank Prof. Dr. Carsten Steger and Prof. Dr. Markus Ulrich for serving on my dissertation committee. I strongly appreciate all your insight from both the industrial and academic research perspective and consider your thoughts paramount for the evaluation and process of discussing my thesis.

The main interactions on research content, I had with my incredible collaborators. I want to thank in particular Tolga Birdal, Marco Esposito, Hyun Jun Jung, Patrick Ruhkamp, Mahdi Saleh, Fabian Manhardt, Matthieu Hog, and Yannick Verdie. You guys are insane! Thank you for our intense interactions and demonstration of research dedication, thanks for showcasing enthusiasm for novel ideas and for everything you taught me! Also thank you Adrian Lopez-Rodriguez, Axel Barroso-Laguna, and Daniel Hernandez-Juarez for the innovative ideas we discussed and thanks Steven McDonagh, Benjamin Frisch, Christian Rupprecht, Christoph Hennersperger, and Sarah Parisot for being able to always take a different perspective on things and for your constant inspiration. Thank you Aleš Leonardis, Gregory Slabaugh, Slobodan Ilic, Federico Tombari, Krystian Mikolajczyk, and Peter Sturm for your invaluable tips and for contributing to this work with years of broad expertise in computer vision research. Special thanks also to Julia Rackerseder, Beatrice Lentes, and Salvatore Virga for your help to prepare and conduct medical and robotic experiments and paper writing sprints, and thanks to Ruiqi

Gong, Shervin Dehghani, Diego Martín Arroyo, and An Lu for your hard work and contributions to realize and implement demos of our ideas. I also want to express my sincere gratitude for the many helpful conversations with various researchers via e-mail and during virtual and in-person meetings at different conferences as well as the helpful discussions with my research colleagues at CAMP. An extra big thanks goes also to Martina Hilla who holds together all people and projects at CAMP while always keeping the overview. You are one of a kind, Martina!

Dear anonymous reviewers, in particular the ones of you that voted for reject, you were an inspiration to rethink our work and enabled us to improve it even though it was disappointing at first. Thank you for your feedback!

A convincing computer vision idea always benefits from an efficient realization in code. I want to thank Sebastian Stelting, Paul Kreuzer, Svetlana Levit, Dirk Adler, Nicolas Rébena, and Niklas Küppers for their explanations and patience with a mathematician by education when they introduced their best practices for image processing with C++ to me.

From all students with whom I had the pleasure to do a semester project, discuss a paper in detail for a seminar or supervise their thesis, I always learned something. Thanks for exploring new areas and understanding the last detail of an algorithm in computer vision with me (in chronological order) Daoyi Gao, Hanzhi Chen, Barnabé Mas, Muhammad Faizan, Ihsan Balaban, Jeremias Neth, Konstantinos Zacharis, Elias Marquart, Tobias Eder, Varnika Tyagi, Nour Al Orjany, Ritvik Ranadive, Stefano Gasperini, Hooriya Anam, Nitin Deshpande, Miguel Trasobares Baselga, Matthieu Hongtao Zhang, Antoine Keller, Violeta Sofia Morales, Lennart Bastian, Tomas Bartipan, Tobias Valinski, Lu Sang, Michael Haberl, Raphael Ullmann, Aleksandr Bulankin, Ekaterina Kanaeva, Ester Molero Hidalgo, Charalampos Papathanasis, Mahdi Hamad, Ramona Schneider, Felix Scheidhammer, Nan Yang, Faisal Kalim, Ahmed El-Gazzar, Thomas Sennebogen, Stefan Matl, Christoph Baur, Markus Herb, Christos Stoilas, Amin Ahantab, and Ahmed Matar.

Finally, I want to say “Danke!” to my family. My parents Elena and Joachim constantly remind me what it means to work hard and stay optimistic no matter how difficult the situation is. You always supported me although I decided for a very orthogonal career. And Eva, my wife and partner in life, I am deeply grateful for your continuous support not just during my PhD, but always. Danke!

Contents

Part I

Introduction & Background

1 Introduction

> *"What is the use of a book," thought Alice,*
> *"without pictures or conversations?"*
>
> **– Alice**
> (Alice's Adventures in Wonderland)[1]

Knowing the position and orientation of objects in space is a basic concept that enables us to understand and interact with our environment. The world around us has a spatial extend in three dimensions and objects can move in these directions and rotate around these three axes. An intelligent vision system that passively observes or is used to actively manipulate its surroundings requires a solid understanding of the six parameters from geometric displacements caused either by ego-motion or the movement of objects in its line of sight.
The human visual apparatus is our broadband interface to the world and as such heavily interconnected with the brain that facilitates fast interpretation of this sensory impression enriched in real-time with information from all our other senses and the experience reflected by our mind.

A plethora of analogue and digital sensors have been inspired by our own information processing pipelines and mimic the gathering of environmental information. These artificial devices are capable of detecting information often far beyond the accuracy and abilities of humans. Digital sensors are nowadays omnipresent and part of your watch, laptop and mobile phone. They guarantee cars to drive safely, activate the light at night and enable video conferences with your colleagues overseas. However, the design of systems that reliably evaluate data from optical sensors such as digital cameras to enable measurements of displacements is an intricate process. And the fusion of information from multi-modal sensor inputs to combine a variety of orthogonal events is a non-trivial task for machines.

In this thesis, we focus on designing and improving pipelines and algorithms in silico that appear natural for our biological system and programmatically expand their capabilities with accurate mathematical models. We thereby look at computational pose estimation and multi-modal sensor fusion with 3D computer vision systems and teach experience in a data-driven fashion. In the spirit of Alice, we try to motivate our thoughts and ideas visually throughout the thesis and discuss dialectically the disadvantages and benefits of system designs when we explore the wonderland of 3D computer vision.

[1] Lewis Carroll. Alice's Adventures in Wonderland [Chapter I, p. 3]. Sam'l Gabriel S.& C., 1916.

1.1. Motivation & Objectives

The measurement of 6D poses with its three translational and three rotational degrees of freedom plays a key role in modern computer vision pipelines. It allows the interaction of robotic manipulators with their surrounding,[2] helps to self-localize mobile agents in unknown environments[3] and is the backbone for augmented and mixed reality applications.[4] These setups oftentimes acquire temporally connected video sequences of consecutive images where additional sensors can provide meaningful insights.[5] In particular the medical field has a wide variety of sensing modalities that provide orthogonal information and the spatially correct fusion of such data can be integral for the outcome of medical treatments.[6] Thus, reliable and accurate optical pose estimation and visual tracking systems are essential tools in practice and the improvement for 6D pose estimation pipelines constitutes an active research domain of interdisciplinary interest.[7]

The task of pose estimation can be separated into two branches. **Inside-out** methods perform camera self-localization by observing its surrounding while moving through the scene. Aside of early markerless ego-motion estimation with visual odometry,[8] a series of pipelines for simultaneous localization and mapping (SLAM) exists. The most prominent directions involve direct approaches such as LSD-SLAM[9] and DSO[10] on one side and feature-based pipelines such as ORB-SLAM[11] on the other side. The latter rely on detector-descriptor backbones such as SIFT,[12] ORB[13] or learning based alternatives.[14]
Outside-in and **camera-in-hand** algorithms give the second branch of pose estimators. They compute the object pose relative to a static or moving camera.

Classical outside-in methods for pose estimation rely on **fiducial markers**.[15] The systems that use these markers can also encode an object ID.[16] They detect the displacement of the marker instead of the object itself whose pose is defined with a relative offset. Besides planar markers, both passive and active targets exist for accurate optical measurements.[17] These are usually attached to a rigid metal frame that is fixed to the object of interest and restricts its ergonomics. Optical systems that utilize these markers can track with a sub-millimeter precision using stereo camera setups which triangulate spheres from the rigid body marker.[18]
The triangulated spheres form a sparse 3D point cloud which is fitted to the reference cloud of the rigid body marker. With a close initial estimate, one can find the local optimal pose with

[2] Cf. Calli et al. [56].
[3] Cf. Engel, Sturm, and Cremers [99].
[4] Cf. Tateno et al. [406].
[5] Cf. Wendler et al. [449].
[6] Cf. Vorst et al. [438].
[7] Cf. Garon, Laurendeau, and Lalonde [136].
[8] Cf. Nistér, Naroditsky, and Bergen [307].
[9] Cf. Engel, Schöps, and Cremers [98].
[10] Cf. Engel, Koltun, and Cremers [97].
[11] Cf. Mur-Artal, Montiel, and Tardós [297], Mur-Artal and Tardós [298] as well as Campos et al. [58].
[12] Cf. Lowe [264].
[13] Cf. Rublee et al. [356].
[14] Cf. DeTone, Malisiewicz, and Rabinovich [86].
[15] Cf. Kato and Billinghurst [204], Naimark and Foxlin [303], Fiala [111], as well as Olson [316].
[16] Cf. Garrido-Jurado et al. [137].
[17] Cf. Marinetto et al. [276].
[18] Cf. Elfring, Fuente, and Radermacher [96].

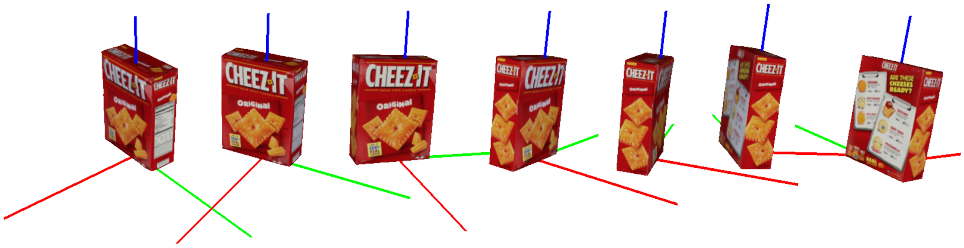


Fig. 1.1. Object motion in temporal sequence. An object moves throughout a sequence of seven frames. The object pose is indicated by a reference frame with x-axis in red, y-axis in green and z-axis in blue.

the iterative closest points (ICP)[19] algorithm or its variants[20] which provide fast convergence or one uses the centroid-aligned Kabsch approach[21] to determine the optimal rotation. While these methods are computationally efficient they can lead to incorrect results under partial occlusion or in the presence of imprecise initialization ultimately restricting the measurable movements of the object of interest as full marker visibility is required at all times.

Markerless outside-in **methods** fully rely on the visual content acquired by the camera to estimate the pose and are less precise. Early methods use parametric models and geometric primitives such as quadrics[22] and superquadrics[23] in combination with depth sensors to estimate objects of simple shapes. The advent of machine learning made it possible to **learn appearance-based object poses** from annotated datasets such as LineMOD[24] and its successors[25] some of which also included temporal video sequences for optical tracking[26] where poses change smoothly in time as depicted in Fig. 1.1. Modern RGB-D tracking methods can generalize to unseen objects,[27] but are very sensitive to their initialization and likely to drift. The learning paradigm for markerless pose estimation either follows a discretization approach of pose space where a classifier is trained[28] or uses regression of the pose parameters directly.[29] A final improvement is then often reached with the help of a depth sensor.

Most of these methods rely on real data which suffers from a time-consuming annotation process, however, a recent trend also leverages synthetic renderings of 3D object models and investigates the reduction of the resulting domain gap.[30] The learning for multiple objects is very time-consuming as one pipeline is usually specifically trained for each individual object. However, a step towards more generic frameworks is done by Wang et al. [443] who estimate parameters for unseen objects with similar appearance from RGB-D inputs.

To describe rigid body displacements, the literature leverages a variety of **pose parametrizations**. Oftentimes, 6D poses are treated with parameters in $\mathrm{SO}(3) \times \mathbb{R}^3$ where rotation and

[19] Cf. Besl and McKay [23].
[20] Cf. Rusinkiewicz and Levoy [359].
[21] Cf. Kabsch [202].
[22] Cf. Cross and Zisserman [75].
[23] Cf. Leonardis, Jaklic, and Solina [243].
[24] Cf. Hinterstoisser et al. [174].
[25] Cf. Tejani et al. [408], Hodaň et al. [178] as well as Kaskman et al. [203].
[26] Cf. Xiang et al. [454] as well as Garon, Laurendeau, and Lalonde [136].
[27] Cf. Garon, Laurendeau, and Lalonde [136].
[28] Cf. Kehl et al. [210].
[29] Cf. Wang et al. [441].
[30] Cf. Rad, Oberweger, and Lepetit [340].

translation components are considered separately.[31] The rotational component enjoys a variety of frequently used parametrizations such as Euler angles, rotation matrices and quaternions. One can also use dual number theory[32] to jointly describe rigid poses as elements of a quadric in the 7-dimensional real projective space $\mathbb{RP}^7$. While these unit dual quaternions appear less intuitive at first, they provide a low-complexity tool for fast pose computation[33] and the Riemannian structure of their parameter space can be beneficial for efficient sensor fusion.[34]

Our first objective in this light is to design a robust and flexible image processing pipeline that enables highly **accurate marker detection** without the requirement for rigid body frames. Using epipolar geometry and a careful hardware design that allows for precise calibration, we then want to overcome current practical obstacles when using **optical tracking systems** and improve pose computation from video sequences targeting **sensor fusion** applications with multi-modal input. A third investigation aims to provide a generalization of current **markerless pose estimation** pipelines by means of problem reformulation.

1.2. Key Contributions

To fulfill these objectives we provide a set of contributions which we detail hereafter. The relevant dissemination platform to these results is indicated below the contribution.
The first three contributions lead to a novel marker-based optical tracking system:

1. Robust algorithm for sub-pixel precise ellipse detection in real-time.
An accurate detection pipeline is designed and implemented to locate image coordinates of self-adhesive circular markers attached to an object of interest in the presence of partial occlusion and illumination changes. The algorithm can be leveraged for accurate calibration routines and replaces bulky and inflexible rigid body markers and facilitates individual marker setups.

> Benjamin Busam, Marco Esposito, Simon Che'Rose, Nassir Navab, and Benjamin Frisch. "A Stereo Vision Approach for Cooperative Robotic Movement Therapy". ICCV, ACVR, 2015. [49]. [Oral Presentation].

2. Hardware design of optical tracking system and miniature versions.
An optical outside-in stereo system prototype is designed and manufactured. The system consists of a hardware-synchronized binocular stereo global shutter camera pair with actively triggered infrared illumination, strobe controlling and a band-pass filter to allow retro-reflective transmission. A custom carbon-fiber mount stabilizes the calibration. A miniature camera-in-hand and an inside-out version are also developed and validated in medical applications.

> Benjamin Busam. "Adaptable High-resolution Real-time Stereo Tracking". EMVA Young Professional Award 2015 from the European Machine Vision Association (EMVA).

[31] Cf. Jia and Evans [195].
[32] Cf. Kenwrigth [215].
[33] Cf. Kavan et al. [207].
[34] Cf. Varghese, Chandra, and Kumar [433].

3. High-performance optical pose computation pipeline.
An algorithm for sub-millimeter precise 6D pose estimation and tracking in real-time is contributed. We formulate an accurate stereo triangulation to enable robust sparse point set registration in the presence of noise, outliers and occlusion. A probabilistic formulation of the interdependent correspondence and pose estimation problem is solved with an energy optimization where mutual updates for pose and correspondence with gradual confidence level increase allow for a large convergence basin. Describing the displacement with dual quaternions results in an efficient optimization where an online learning algorithm can dynamically adjust the marker representation. We use the robust approach to prototype a cooperative robot movement therapy for hemiparetic patients.

> Benjamin Busam, Marco Esposito, Simon Che'Rose, Nassir Navab, and Benjamin Frisch. "A Stereo Vision Approach for Cooperative Robotic Movement Therapy". ICCV, ACVR, 2015. [49]. [Oral Presentation].

Contribution 4. and 5. target markerless pose computation:

4. Inside-out tracking system for 3D ultrasound.
Outside-in trackers suffer from line-of-sight issues. We place a miniature stereo system inversely onto an object with field of view into the operating room and utilize a SLAM pipeline for inside-out (IO) tracking to study the advantages in comparison with a commercial outside-in system. Due to a rotation leveraging effect, small rotational motions change the visual image content for the IO tracker significantly which results in higher rotational pose resolution and improves 3D ultrasound compounding accuracy.

> Benjamin Busam, Patrick Ruhkamp, Salvatore Virga, Beatrice Lentes, Julia Rackerseder, Nassir Navab, and Christoph Hennersperger. "Markerless Inside-Out Tracking for 3D Ultrasound Compounding". MICCAI, POCUS, 2018. [53]. [Oral Presentation and Live Demonstration].

5. Reformulation of object pose estimation as an action decision process.
We propose a markerless monocular 6D object pose estimation pipeline based on a lightweight neural network that is trained only with synthetic data. While pose estimation pipelines either follow a regression or classification scheme, we redefine the pose prediction paradigm and look for the next best pose given an observation and a current estimate. We gradually move a virtually rendered object model in incremental steps towards the observation. Learning this action decision process reinforces correct updates and allows for both training and testing on a laptop. The question we ask leads to an object agnostic tracking algorithm that consequently extends to a 6D pose detector which can estimate the pose of objects that were never seen during training. We improve the accuracy of recent approaches while being able to dynamically reduce the computation cost in case of insignificant motion. An additional attention mechanism makes the model robust to occlusion.

> Benjamin Busam, Hyun Jun Jung, and Nassir Navab. "I Like to Move It: 6D Pose Estimation as an Action Decision Process". arXiv:2009.12678, 2020. [52]. [arXiv preprint].

The last three contributions target sensor fusion in heterogeneous environments. They improve the poses and facilitate spatial modality-fusion:

6. Efficient pose upsampling with dual quaternions.
A unified approach for efficient interpolation and extrapolation of poses with concepts from differential geometry is proposed. This facilitates synchronization between hybrid pose tracking systems for sensor fusion and provides a tool to reduce temporal delays and lags.

> Benjamin Busam, Marco Esposito, Benjamin Frisch, and Nassir Navab. "Quaternionic Upsampling: Hyperspherical Techniques for 6 DoF Pose Tracking". 3DV, 2017. [50].

7. Pose denoising with local regression geodesics.
We suggest an approach for camera pose filtering on the Riemannian manifold of dual quaternions. The accuracy of a temporal pose stream is improved by active denoising in the presence of outlier estimates using the structure of the pose space itself. The novel method utilizes principal component regression in a local linearization of the pose space. By using parallel transport we map a temporally connected moving pose window into the Lie algebra of dual quaternion space represented by the tangent space at the identity where an outlier-aware robust regression corrects the window centre pose.

> Benjamin Busam, Tolga Birdal, and Nassir Navab. "Camera Pose Filtering with Local Regression Geodesics on the Riemannian Manifold of Dual Quaternions". ICCV, MVR3D, 2017. [48]. [Oral Presentation and Best Student Paper Award].

8. Spatial modality-fusion with accurate pose estimates.
The advantages of our pose estimation systems allow to combine orthogonal modalities. Besides contributing an industrial and mobile mixed reality setup, we exemplify the benefit for sensor fusion with a collaborative medical robot that holds a camera-in-hand miniature system and a gamma detector array to improve a breast cancer staging procedure. Real-time fusion of functional nuclear gamma and anatomical ultrasound images enable the robot to assist a sentinel lymph node punch biopsy.

> Marco Esposito, Benjamin Busam, Christoph Hennersperger, Julia Rackerseder, Nassir Navab, and Benjamin Frisch. "Multimodal US–Gamma Imaging using Collaborative Robotics for Cancer Staging Biopsies". IJCARS, 2016, Volume 11, Issue 9. [102]. [Best Paper Award].

> Marco Esposito, Benjamin Busam, Christoph Hennersperger, Julia Rackerseder, An Lu, Nassir Navab, and Benjamin Frisch. "Cooperative robotic gamma imaging: Enhancing us-guided needle biopsy". MICCAI, 2015. [101]. [Oral Presentation].

1.3. Outline & Overview

Before we start with the detailed scientific explanations and discussion, we give a brief overview over the content of this thesis. Throughout the chapters, we try to make the content self-contained by introducing relevant and necessary concepts as well as high-level connections in the specific domain. The chapters are structured as follows:

Chapter 1 introduces and motivates our work, points out the key contributions and provides an overview of the topics in the thesis.

Chapter 2 explains the fundamental concepts of an image acquisition system and defines basic structures.

Chapter 3 details the necessary steps of a 2D image processing pipeline that robustly detects ellipses and extracts centre coordinates with sub-pixel precision in real-time.

Chapter 4 explains the camera geometry and defines the camera model used. We detail its calibration routine using the ellipse detector from chapter 3.

Chapter 5 illustrates the use of artificial neural networks for data-driven image processing. We state the potential and explain basic concepts of convolutional neural network inference and training.

Chapter 6 summarizes 3D depth sensing concepts and compares 6D pose parametrizations. The chapter discusses the geometry of pose parameter spaces and provides insights in epipolar geometry. This part further describes point triangulation from binocular stereo and two-view depth estimation with neural networks. We end with an algorithm for point cloud triangulation from self-adhesive, retro-reflective circular markers.

Chapter 7 is the core of our marker-based high performance optical tracking system (OTS). We investigate hardware design choices for an outside-in OTS and its miniature versions and details a robust real-time pose estimation algorithm. The algorithm utilizes joint dual quaternion parametrization to fit a point cloud observation to a source point cloud with mutual correspondence and pose improvements while increasing a fitting confidence. We further validate the OTS, explain the pose communication interface and detail co-calibration routines. Limitations and potential solutions are discussed and we apply a camera-in-hand system prototype in a medical robotic environment where a collaborative robotic arm performs a movement therapy targeted to assist hemiparetic patients in the re-education of upper limb movements.

Chapter 8 focuses on markerless pose estimation and consists of two parts. In the first part, we analyse the capabilities of SLAM-based inside-out tracking for 3D ultrasound compounding in comparison with a commercial outside-in tracking system. The rotation leveraging effect that significantly changes the image of an inside-out camera during a rotational motion improves the rotational accuracy in difficult medical ultrasound procedures such as transrectal prostate fusion biopsy and improves the quality of 3D ultrasound compounding.
The second part looks into other means of object pose estimation without markers such as

parametric primitives, shape decomposition and feature extraction pipelines. We then redefine the 6D object pose estimation paradigm as an action decision process comparing a current rendering of the 3D model with the image observation. A lightweight neural network decides for incremental updates of the rendering which brings it gradually closer to the observation. The resulting network is trained fully on synthetic data and provides an improved accuracy while being able to even track the poses of unseen objects.

Chapter 9 looks at ways to adjust pose measurements. We start with a unified formulation for pose interpolation and extrapolation. Using (dual) quaternion parametrization allows for simple yet efficient and smooth interpolations along geodesic trajectories in pose space that are physically interpretable. We validate the pose extrapolation accuracy for natural hand motion and see that we can use the approach to synchronize hybrid systems and reduce lag and latency for sensor fusion.
We then investigate local regression geodesics to denoise temporally connected pose signals in non-Euclidean pose spaces in presence of outliers. A local linearization of the pose space for unit (dual) quaternions is used to smooth the temporal signal via principal component regression in a moving local pose sequence window around a specific pose estimate.

Chapter 10 showcases the use of accurate and reliable pose estimation for multi-modal spatial fusion. We exploit the practical benefit in three orthogonal scenarios. A high-energy 3D printer melts metal particles with a laser for additive manufacturing in an industrial environment where we fuse thermal and geometric information from various locations by accurately measuring relative poses for quality control. A second use case illustrates a mobile augmented reality application that aims at providing anatomical pose guidance for ultrasound scans. The final example illustrates a medical robotic setup where we equip a cooperative medical robot with a camera-in-hand system to assist a medical expert during breast cancer staging. The optical tracking makes the fusion of ultrasound and nuclear imaging possible during a sentinel lymph node biopsy procedure.

Chapter 11 critically summarizes the thesis and points out limitations. We discuss potential solutions to it and show promising future directions and prospects.

The **Appendix** finalizes the thesis and includes detailed derivations and additional complementary results. It lists the set of authored and co-authored publications and supervised academic projects as well as the managed research funding and acquired grants. We end with lists for the proposed algorithms, figures and tables and lastly the bibliography.

2 Fundamentals

> *Omnium enim rerum*
> *principia parva sunt.*
>
> – **Cicero**
> (De finibus bonorum et malorum)[1]

Multiple mathematical concepts serve as the foundation for the developed techniques and algorithms in this thesis. Before we start to investigate digital image processing and 3D geometry, we present the used physical and mathematical models and look at our data source in a traditional image acquisition pipeline.[2]

2.1. Data Acquisition

We start by considering the physical components of a classical machine vision system in industry to build a foundation for hardware components used in computer vision. If one wants to detect, for instance, the position, shape or the movement of a certain object within a volume or if one wants to check whether an industrially manufactured product is complete or possesses special geometric parameters, one is often not able to do such extremely precise and sometimes highly complex tasks manually but has to rely on different computer-based alternatives. In the industry, challenges like that are frequently solved with the help of machine vision.
A typical **machine vision system** to tackle these kind of problems consists of several components which can interact with each other. Fig. 2.1[3] shows the specific setting for such a purpose.
An image of the object of interest (1) is acquired while the illumination (3) illuminates the scene in the classical **outside-in** scenario where a static camera (2) observes the scene. If the camera is mounted on the moving object instead, the setup becomes inverted and we call the view **inside-out**; this will be the case for some of our later investigations in chapter 8.1, but we commonly consider the static case. In both cases, however, strobe illumination and image acquisition are triggered by a camera-computer interface (4) which stores the image data in the memory (5) of a computer where the machine vision software (6) evaluates the image and returns an inspection result (7).

[1] "The beginnings of all things are small.", M. T. Cicero. De finibus bonorum et malorum [5.58, p. 460]. William Heinemann, 1914.
[2] We follow the concepts and notation of Busam [47].
[3] Figure based on Steger, Ulrich, and Wiedemann [395, p. 2].

The actual process of image acquisition with its physical background, its hardware components, and its intrinsic data structures and signal processing tasks plays an important role for the realization of automated inspections and other typical scenarios. If a defined industrial problem has to be solved, one is forced to think about all the different components in Fig. 2.1. However, the main focus of this thesis will be on the software- and algorithm-based part of the process chain within the computer. We consider (2), (3), and (4) mostly to be part of a black box which provides an image stream for our software solutions. Although these components are essential for the whole process, we do not research them in more detail since this has already been done by several authors before. An elaborate introduction to the single hardware parts, their functions, and the pros and cons of different component designs can for example be found in the book of Steger et al. [395, pp. 5–63].

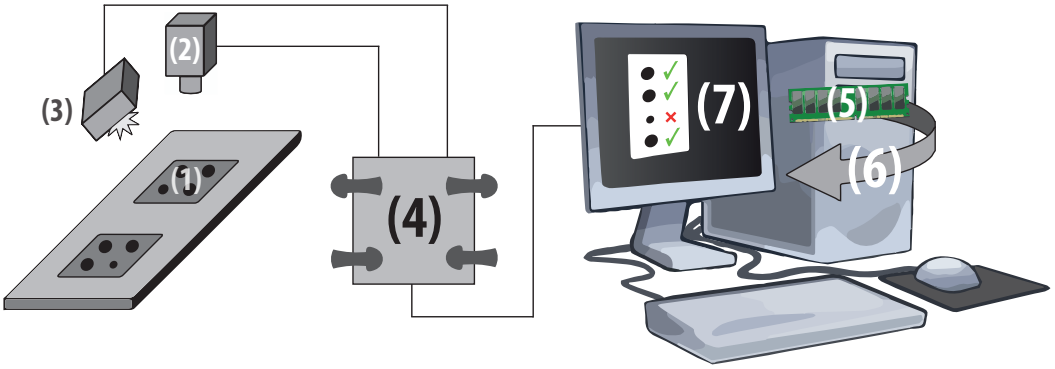


Fig. 2.1. Components of a typical machine vision system. The object of interest (1) is observed by a camera (2) with an active illumination (3). A camera-computer interface (4) triggers the acquisition and stores the image data in the memory (5) of a computer where vision software (6) processes the image and returns information (7).

Now that we know about the processing chain of an image acquisition system, we intend to focus on the image itself and establish precise mathematical definitions of images and videos as a basis for our analysis afterwards.

2.2. Image & Video

Image and video data is the essential input for our ideas and approaches. Subsequently, we define the data description in uniform, mathematical terms that allow us to define pixels inside monochrome and colour images and temporal sequences of images in a video. Moreover, we define different neighbourhood terms for local pixel regions.

2.2.1. Image Data

From now on, if we talk about images we implicitly refer to digital single-channel raster grey scale images which consist of pixels. This is to say, whatever comes from the image acquisition task explained beforehand, gives a certain digitized discrete pixel domain with some grey value for every single element in it.[4] In case we use colour images, we focus on 3-channel RGB sensors or explicitly mention if a different domain is used.
Depending on the number of possible values for one pixel, we give the definition below.

Definition 2.1
*The **discrete pixel depth** d of an image is given by the number of used bits per pixel (bpp) to store the colour information.*
The set of 2^d different values to describe a pixel with d bpp is given by

$$\mathbb{G}_d = \left\{0, 1, \ldots, 2^d - 1\right\} \subset \mathbb{N}_0. \tag{2.1}$$

In the literature the terms *bit depth* or *colour depth* are also commonly used.
Usually we work with a discrete pixel depth of 8 bits (1 byte) per pixel which gives $2^8 = 256$ different possible shades of grey with the representation $\mathbb{G}_8 = \{0, 1, \ldots, 255\}$ where the smallest value 0 is assigned to black, whereas 255 codes white. As conventional cameras produce two-dimensional rectangular pixel domains,[5] we can regard an image as an integer valued function over a discrete grid whose values can be visualized with a bar plot where the bar heights represent the pixel values of the underlying grid as illustrated in Fig. 2.2.

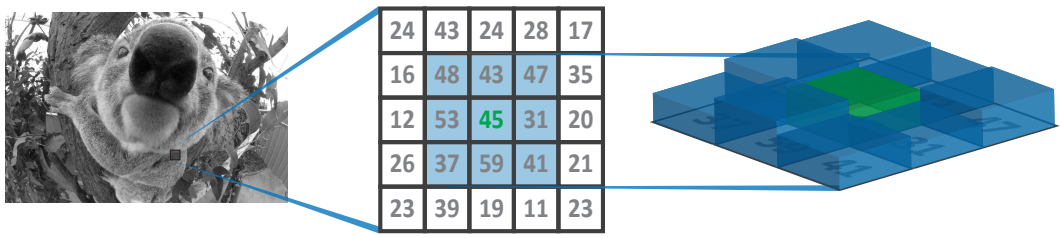


Fig. 2.2. **Image description by matrix and bar plot over discrete pixel grid.** A single channel image with 8-bit pixel depth with values in $\mathbb{G}_8 = \{0, 1, \ldots, 255\}$ is shown. The value 0 describes a fully black pixel while 255 decodes white.

The visualization with spatial bars may be unconventional. A planar way to look at the scenario would just be to think of the image as a matrix with the dimensions of the number of pixels in both width w and height h and the grey value $g_{x,y} \in \mathbb{G}_8$ at entry (x, y).
In a formalized manner, we therefore describe an image as a function I in the following way:

[4] In the following, see Steger, Ulrich, and Wiedemann [395, p. 66].
[5] We want to neglect line-scan cameras with only a single row of pixel sensors here.

Definition 2.2
*A **multi-channel image** of width w, height h, and pixel depth d, and c channels is a function*

$$I : D \to \mathbb{G}_d^c \tag{2.2}$$
$$(x, y) \mapsto g_{x,y}^k \tag{2.3}$$

where

$$D = \{0, 1, \ldots, w-1\} \times \{0, 1, \ldots, h-1\} \subset \mathbb{N}_0^2 \tag{2.4}$$

*is the **image domain** and*

$$\mathbb{G}_d^c = \underbrace{\mathbb{G}_d \times \mathbb{G}_d \times \ldots \times \mathbb{G}_d}_{c \text{ times}}, \tag{2.5}$$
$$k \in \{0, 1, \ldots, c-1\}. \tag{2.6}$$

In terms of Fig. 2.2, I simply gives the height map for the pixel grid D per channel. Sometimes it can be advantageous to speak about an image as the actual representation of I in $\mathbb{G}_d^c$.
For most programming languages, the first index of an array is denoted by 0. This is the implementation-driven reason for us to start counting at 0 here. We thereby directly avoid differences or misunderstandings within the computation part of the processing chain from Fig. 2.1. If the pixel depth d is not our main issue or if it is clear what we look at, we drop the index. The channel number $c \neq 0$ is relevant for onscreen colour images because most video screens work with an additive mixture of colour stimuli with an underlying model consisting of three different colours.[6] The **RGB** model has three channels, one for each **r**ed, **g**reen, and **b**lue and using them at different intensities produces a wide variety of colours for the human visual system which works in a similar way.[7] Fig. 2.3 visualizes these different sources for an RGB colour image. In this important case, it is $c = 3$. However, for plenty of different tasks we can treat the channels separately and therefore remain with $c = 1$ to also simplify notation.

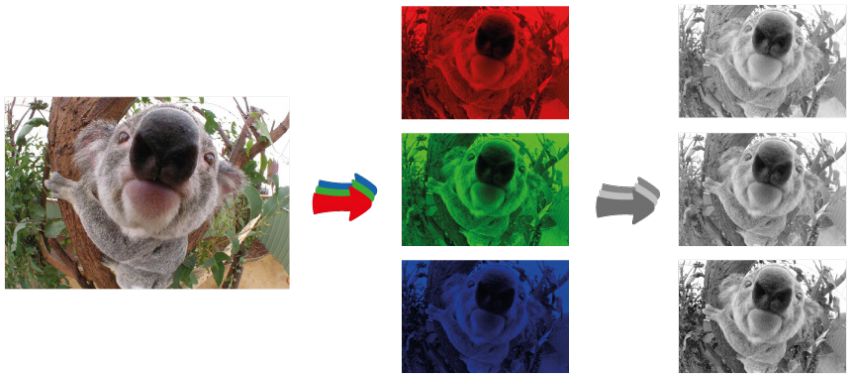


Fig. 2.3. **RGB image with separated channels.** For visualization purposes, the 3-channel image is separated into its red, green and blue components. Every single component can be viewed as an individual grey scale image.

[6] Cf. Brockhaus [41]. Catchword: RGB-Farbmodell.
[7] Humans are trichromats. They have three different types of colour receptors with special absorption spectra, see Brockhaus [41]. Catchword: Farbensehen beim Menschen.

2.2.2. Video Data

Temporal information is crucial for specific computer vision tasks which include movement such as object tracking and camera pose estimation. Thus, we formally extend the domain for the function in Definition 2.2 by adding a third dimension (time) to D which brings

Definition 2.3
*A **single-channel video** of width w, height h, pixel depth d with f frames is a function*

$$I : D \to \mathbb{G}_d \tag{2.7}$$
$$(x, y, z) \mapsto g_{x,y,z} \tag{2.8}$$

with

$$D = \{0, 1, \ldots, w-1\} \times \{0, 1, \ldots, h-1\} \times \{0, 1, \ldots, f-1\} \subset \mathbb{N}_0^3. \tag{2.9}$$

Varying only the parameter of the third dimension of D gives the set of considered frames. A fixed $f_l \in \{0, 1, \ldots, f-1\}$ yields the image domain corresponding to frame f_l and a frame rate declares how many **f**rames **p**er **s**econd (**fps**) are streamed from the camera.

2.3. Neighbourhood

It is necessary to sometimes restrict the area for a special image processing operation locally. For this reason we analyze the surroundings of a certain pixel p in more detail. Depending on the context, we use different approaches of the theory of cellular automata[8] to define various neighbourhoods for p.

Definition 2.4
*If it exists, the **von Neumann neighbourhood** of a pixel $p \in D$ is given by*

$$U_{\text{von Neumann}}(p) = \left\{ q \in D \mid \|p - q\|_1 \leq 1 \right\}, \tag{2.10}$$

*the **Moore neighbourhood** of p by*

$$U_{\text{Moore}}(p) = \left\{ q \in D \mid \|p - q\|_\infty \leq 1 \right\}, \tag{2.11}$$

*and the **extended Moore neighbourhood** of p by*

$$U_{\text{extended Moore}}(p) = \left\{ q \in D \mid \|p - q\|_\infty \leq r \right\}, \tag{2.12}$$

where $r \in \mathbb{N}_0$ with $r \geq 2$.

[8] Cf. Müller and Kuttler [295, p. 350].

The norms used in the definition are the standard taxicab norm and the maximum vector norm.[9] Fig. 2.4 illustrates the three types of neighbourhoods. This explains why **4-neighbourhood** or **8-neighbourhood** are also possible terms for the two different surrounding pixel sets.

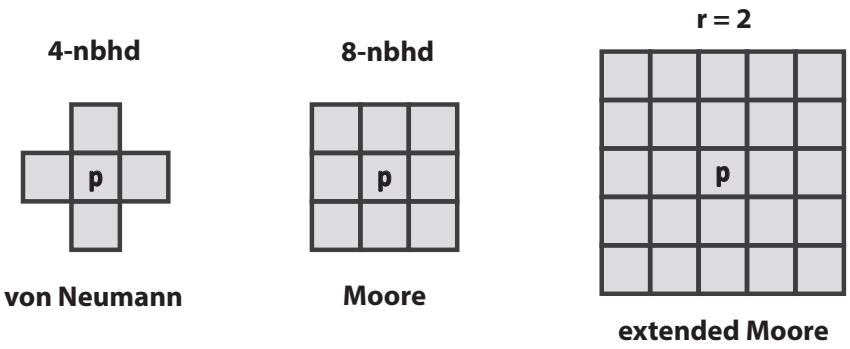


Fig. 2.4. **Different neighbourhoods for pixel p.** The neighbourhood definitions include four and eight pixels for Neumann and Moore and can be extended with increasing threshold r.

What happens if a certain neighbourhood of pixel p does not exist? This is equivalent to the question of how we define the neighbourhood of p at image boundaries. Depending on the context, there certainly are possible solutions on how to fill the neighbourhood pixels outside of the image domain. For our purposes, though, we just truncate the pixel set to make it fit the image as Fig. 2.5 shows.

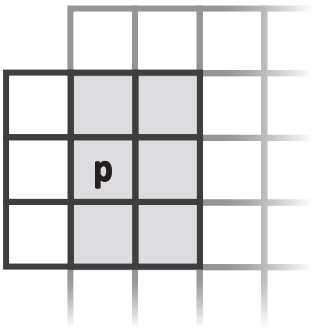


Fig. 2.5. **Truncated Moore neighbourhood.** The neighbourhood size is decreased at image boundaries according to the image domain.

As before, we want to define a time-dimensional neighbourhood. This is done by defining the frames f_{l-1} and f_{l+1} as neighbour frames of f_l if they exist. Definition 2.4 for pixel neighbourhoods translates directly into the new context and our previous truncation rule still holds true for all boundary regions. Fig. 2.6 exemplifies a pixel p of video frame f_l with its neighbour frames and its von Neumann neighbourhood within the video. These thoughts are likewise applicable to every channel of our definition of multi-channel images from Definition 2.2.

The elementary definitions for our further research are made and we established how image data is acquired.
The concepts we developed so far help to describe the subject of our following discussions by

[9] Definitions for the vector norms $\|\cdot\|_1$ and $\|\cdot\|_\infty$ can be found in Bronstein et al. [42, p. 267]. For general information on ℓ_p norms, see Rudin [357, p. 78].

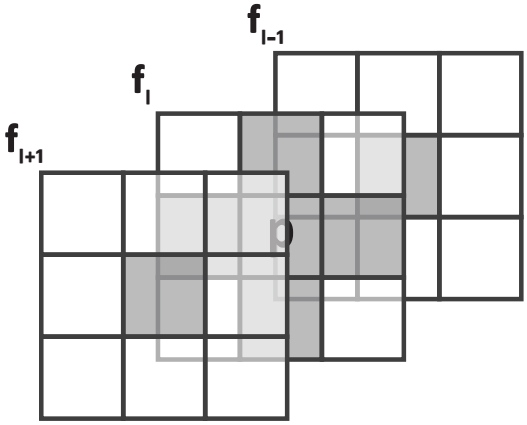


Fig. 2.6. **Von Neumann neighbourhood of pixel p within a video.** The neighbourhood extends both in image space and across the temporal dimension.

enabling us to formulate highly specific algorithms to process images in order to acquire a variety of content information to tackle numerous imaging problems.

Henceforth, we develop applicable tasks and algorithms based on the presented concepts and models. As a first step, we describe ways on how to get high-level information like area connection or contour paths from low-level pixel grid data. Besides, we examine ways to improve image quality as a preprocessing step and formulate an algorithm to automatically detect marker locations from images to perform a camera calibration.

Part II

Image Analysis

3 Image Processing

> *We often plough so much energy*
> *into the big picture,*
> *we forget the pixels.*
>
> – **Silvia Cartwright**

This part of the thesis concentrates on processing 2D images with classical and data-driven methods. We formulate ways for automatic pixel processing and detail a parametric camera model which we aim to optimize. Our investigations remain fully on the projected flat sensor grid where we intend to elaborate principle approaches and algorithms before we apply them to 3D computer vision problems.[1]

A grid with pixel values as formally described in the previous chapter does not directly reveal high level content information of what is visible. Measuring intensities and collecting their relative values does neither give size nor location or amount of particles or some objects of interest.
The essence of the following paragraphs is thus to find and describe efficient methods that are able to extract abstract image content.
In the first part of this chapter, we focus on data optimization for standard images given by typical image acquisition systems as described in section 2.1. Subsequently we discuss classical ways to deal with high level image information and conceptualize single view treatment for object segmentation and the detection of special features within image subdomains. At the end of this chapter, we formulate an algorithm that combines the previously discussed methods and detects centroids of elliptic shapes on a sub-pixel precise level which will pave the way for a camera calibration presented thereafter in part 4.2.
As a starting point, we concentrate on some preliminary work by improving the captured camera data.

3.1. Image Enhancement

High accuracy in industrial detection tasks and reliable quality control standards can not be achieved with noisy, inaccurate or lossy image data. We are therefore interested in reducing the influence of the main interference factors by optimizing the image in order to gain the best possible visual data by **image preprocessing**.

[1] We reprint Busam [47] in parts to detail our description.

3.2. Grey Value Transformation

In the beginning, we focus on position independent modifications and formulate a robust way to prepare pixel data for further analysis. From a mathematical point of view such global operations can be described as follows:

Definition 3.1
*A **grey value transformation** h of a single-channel image with values $g_{x,y}$ is a function*

$$h\colon \mathbb{G} \to \mathbb{G} \tag{3.1}$$
$$g_{x,y} \mapsto h\left(g_{x,y}\right). \tag{3.2}$$

Using a transformation h on an image f is then equal to the function composition $h \circ f$ where the transformation h is arbitrary and does not depend on the actual pixel position but only on its value. Depending on the pixel depth, an acceleration of this task in practical applications can often be achieved by using **look-up tables** where the function values for every grey level are stored and calculating operations can be omitted.[2]
An important role in this context plays the **linear grey value scaling**

$$h\colon \mathbb{G} \to \mathbb{G} \tag{3.3}$$
$$g_{x,y} \mapsto a g_{x,y} + b \tag{3.4}$$

with $a \in \mathbb{R}^+$ and $b \in \mathbb{R}$. The parameter b influences the overall brightness of the image whereas a affects the contrast by either enhancing or diminishing the pixel value $g_{x,y}$.
Equation (3.3) and (3.4) allow for normalization of the contrast of an image with pixel depth d – this is essential to achieve illumination-invariance for algorithms. If the minimal and

[2] Cf. Demant, Streicher-Abel, and Waszkewitz [81, pp. 40–42].

maximal grey levels are g_{min} and g_{max}, algorithm 3.1 gives such a contrast enhancement.[3] Fig. 3.1 shows some linear grey value transformations with different parameters.

Algorithm 3.1. Contrast Normalization

Input parameters:

- Image f with grey values $g_{x,y}$, domain D, and pixel depth d

Computation steps:

1. Get minimal grey level $g_{min} = \min_{(x,y)\in D} g_{x,y}$
2. Get maximal grey level $g_{max} = \max_{(x,y)\in D} g_{x,y}$
3. Calculate $a = \dfrac{2^d - 1}{g_{max} - g_{min}}$ and $b = -a g_{min}$
4. Perform linear grey value scaling $g_{x,y} = a g_{x,y} + b$

Output:

- Normalized image f with grey values $g_{x,y}$

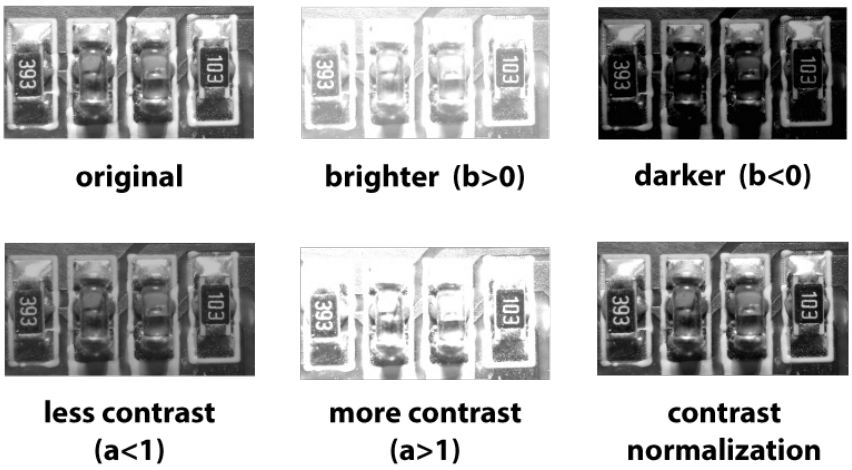


Fig. 3.1. **Different grey value transformations.** The original image is shown on top left. Note the dark regions and strong metallic reflections. The top row shows normalization with different brightness while the bottom row gives examples with different contrast parameters. The bottom right image shows the results of Algorithm 3.1.

In real images shades and highlight spots often cause some few pixels to be very dark or exceedingly bright. These pixels prevent the above transformation to scale the image to an optimal grey value range. A less error-prone way to handle this is via **robust contrast normalization** with a cumulative histogram.[4]

[3] It is true that this is not a grey value transformation in the sense of Definition 3.1 since g_{min} and g_{max} depend on the entire image. However, we can think of these extreme values as given scalars from some pre-calculation step and thus keep definitions simple.

[4] Cf. Schaeffel et al. [364, pp. 517–519].

This works as follows: At first, we calculate the relative frequency (i.e. the probability of the observation) of a certain grey value v within the image by

$$f_v = \frac{\sum_{x,y} \mathbb{1}\{g_{x,y} = v\}}{\sum_{x,y} 1} = \frac{\sum_{x,y} \mathbb{1}\{g_{x,y} = v\}}{h \cdot w}, \tag{3.5}$$

$$v \in \mathbb{G}, \tag{3.6}$$

where $\mathbb{1}\{g \in V\}$ represents the indicator function on V, w is the image width, and h the image height respectively.
Then, we calculate the cumulative histogram

$$q(g) = \sum_{v=0}^{g} f_v, \tag{3.7}$$

$$g \in \mathbb{G} \tag{3.8}$$

and cut off the upper and lower percentiles with parameters p_{up} and p_{low}.
The remaining pixels are then used to calculate g_{min} and g_{max} for the normalization. In the end, values $v \notin \mathbb{G}$ (i.e. below 0 and above $2^d - 1$) become leveled.
In conclusion we retrieve Algorithm 3.2 for this procedure. Fig. 3.2 demonstrates a robust grey value normalization with $p_{low} = 0.02$ and $p_{high} = 0.92$ where this method significantly improves the image with its problematic reflecting parts.

Algorithm 3.2. Robust Contrast Normalization

Input parameters:

- Image f with grey values $g_{x,y}$, width w, height h, and pixel depth d
- Truncation parameters p_{low}, p_{up}

Computation steps:

1. Relative frequencies: $\forall v \in \mathbb{G}: f_v = \dfrac{\sum_{x,y} \mathbb{1}\{g_{x,y} = v\}}{h \cdot w}$
2. Cumulative histogram: $\forall g \in \mathbb{G}: q(g) = \sum_{v=0}^{g} f_v$
3. Cut histogram: $Q_g = \{g \mid p_{low} \leq q(g) \leq p_{high}\}$
4. Remaining extremal values:
   - $g_{min} = \min(Q_g)$
   - $g_{max} = \max(Q_g)$
5. Normalize contrast:
   - $g_{x,y} = \dfrac{2^d - 1}{g_{max} - g_{min}}(g_{x,y} - g_{min})$
   - Level outliers: $g_{x,y} = \begin{cases} 0, & \text{if } g_{x,y} < 0 \\ g_{x,y}, & \text{if } 0 \leq g_{x,y} \leq 2^d - 1 \\ 2^d - 1, & \text{if } g_{x,y} > 2^d - 1 \end{cases}$

Output:

- Robust normalized image f with grey values $g_{x,y}$

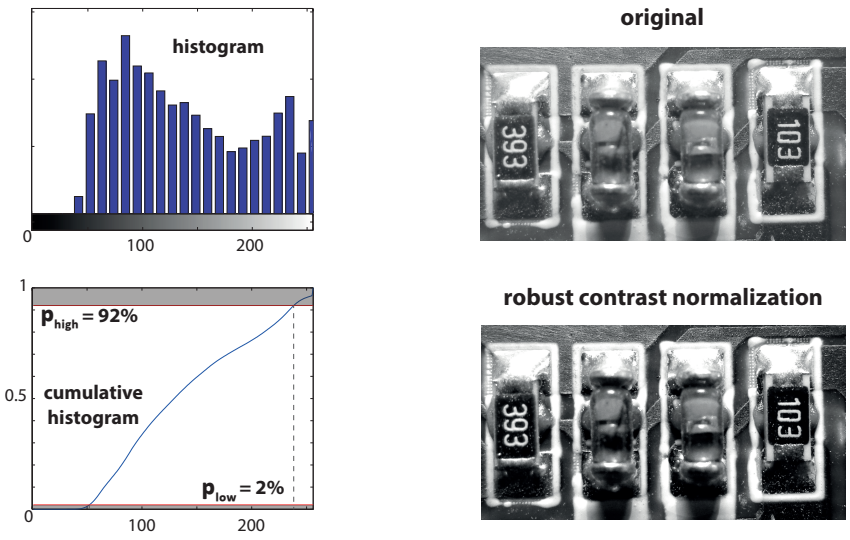


Fig. 3.2. **Robust contrast normalization.** On the top right, the original picture is shown. The left column shows the image and the cumulative histogram with the cut off percentiles illustrated in grey. Bottom right shows the result of Algorithm 3.2 with $p_{low} = 0.02$ and $p_{high} = 0.92$.

3.3. Filtering

In this section, we deal with the reduction of noise. We do so by using the neighbourhood concepts of Definition 2.4 to formulate image filtering operations.

3.3.1. Spatial Domain

A spatial filter is a local transformation around pixel $p_{x,y}$. The filtered value $g_{x,y}$ is a result of some predefined operation and the values of the Moore neighbourhood of radius r. For our purposes linear filters are relevant.

Definition 3.2
*A **linear spatial filter** h with kernel K is a function*

$$h\colon \mathbb{G} \to \mathbb{G} \tag{3.9}$$

$$g_{x,y} \mapsto \sum_{|s|,|t|\leq r} k_{s,t}\, g_{x+s,y+t}. \tag{3.10}$$

Fig. 3.3[5] illustrates the process of linear filtering, where every image pixel is visited once by the filter kernel K of size r filled with the coefficients $k_{x,y}$ as given in Definition 3.2.
Of course, it is also possible to choose non-quadratic neighbourhood ranges or domains with special shapes, but selecting a quadratic Moore neighbourhood is most common and is part of our preprocessing algorithm later on.

[5] Figure based on O'Gorman, Sammon, and Seul [309, p. 61].

Linear filters are strongly correlated with convolutions as we can see by the definition of convolution.[6]

Definition 3.3
*The **discrete convolution** on k and g is given by*

$$(k \star g)_{x,y} := \sum_{s,t} k_{s,t}\, g_{x-s,y-t}, \tag{3.11}$$

where elements with non-existent indices are treated as zeros.

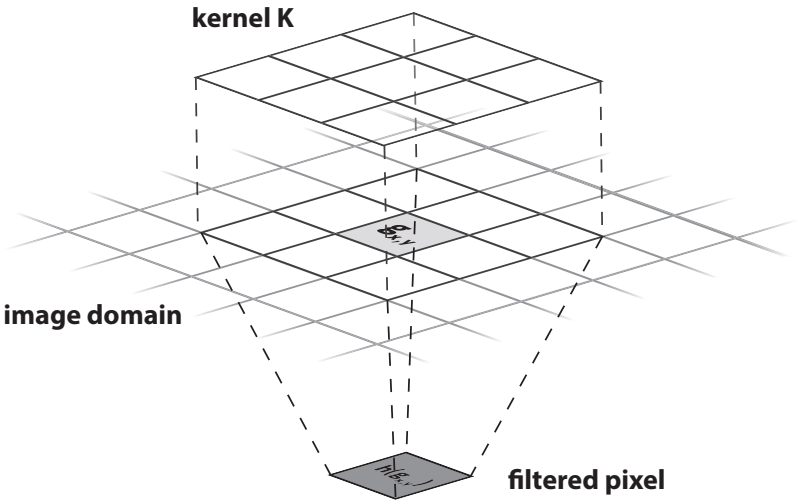


Fig. 3.3. **Linear filter with kernel K.** The image domain with the Moore neighbourhood around pixel $p_{x,y}$ with value $g_{x,y}$ is illustrated with a black grid. The kernel K of the same size acts on this domain. The filtered pixel has the value $h\left(g_{x,y}\right)$.

In other words, a convolution with a kernel function k is just a linear filter with a kernel K that has been rotated 180° since flipping g instead of k does not make a difference but has conventional reasons. In actual implementations, rotating the smaller K is often favourable. This means, whenever we perform linear filtering, we can do so by pre-rotating the kernel and performing a convolution afterwards. In fact, we will use kernels that are invariant under these rotations so that there is no need to rotate these masks beforehand.
Convolutions have become majorly important in computer vision in particular in the field of data driven methods where machine learning is used to build artificial neural networks capable to process image data. These structures involve sets of filters whose weights are learnt. Due to the close relation between filters and convolutions, such pipelines are often called convolutional neural networks (CNNs). Part 5 discusses these concepts in more detail.

[6] Cf. Gonzalez and Woods [151, pp. 146–150].

3.3.2. Frequency Domain

Filtering an image by computing the convolution can be very time consuming depending on the size of the kernel. Some precalculation steps by transforming the image into phase space by change of basis can decrease the number of operations especially if large kernels are used or multiple filters work consecutively. In order to understand this properly, we repeat the definition of the discrete Fourier transform.

Definition 3.4
*The **discrete Fourier transform** F of a function $f : D \to \mathbb{G}$ on a domain D of width w and height h is given by the operator $\mathscr{F}$ with*

$$\mathscr{F}(f(x,y)) := F(u,v) = \sum_{x=0}^{h-1}\sum_{y=0}^{w-1} f(x,y)\exp\left(-2\pi i\left(\frac{xu}{h}+\frac{yv}{w}\right)\right), \tag{3.12}$$

where $(u,v) \in \left[-\frac{w}{2}, \frac{w}{2}-1\right] \times \left[-\frac{h}{2}, \frac{h}{2}-1\right]$.

Lemma 3.1
*Its **inverse transform** can be calculated as*

$$\mathscr{F}^{-1}(F(u,v)) = f(x,y) = \frac{1}{hw}\sum_{u=-h/2}^{h/2-1}\sum_{v=-w/2}^{w/2-1} F(u,v)\exp\left(2\pi i\left(\frac{xu}{h}+\frac{yv}{w}\right)\right), \tag{3.13}$$

where $(x,y) \in [0, w-1] \times [0, h-1]$.

We do not want to go into the analytical details for the inversion formula since this has been addressed already by various authors. Further details can be found for example in the work of Rudin [357, pp. 215–221]. In the literature, the discrete Fourier transform is sometimes referred to as **DFT** and its inverse as **IDFT** and since there is a one-to-one correspondence, the Fourier pair occasionally is symbolized by $f \circ\!\!-\!\!- F$ or $F -\!\!-\!\!\circ f$ respectively. We will use this symbolic abbreviation, too.
With this transformation and its inverse, it is now possible to think about images either in their spatial or in their frequency domain. Fig. 3.4 shows a few images in both domains. Since the DFT gives values in $\mathbb{C}$, their amplitudes are used for the images and normalized to the given grey scale. In general, fast changes in the image content like edges or small details cause high spatial frequencies whereas slow changes or large plane regions are of low spatial frequencies and code sinusoids with longer periods. Fig. 3.5 shows the effect of keeping only high or low frequencies of an image.

As we can only use a finite number of coefficients to calculate a Fourier transform, we cannot avoid getting ringing artifacts especially at sharp edges. Figure 3.6 illustrates this problem for one dimension. This depicts that there will be a loss of image data when we perform a DFT and we shall be aware of it.
What is the practical reason for us to speak about Fourier transforms here? The answer is being given by the following theorem.

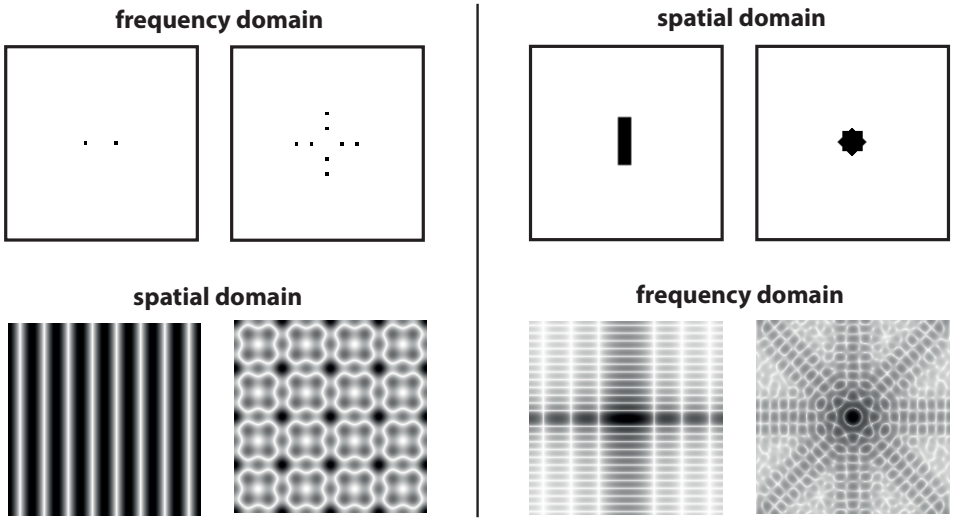


Fig. 3.4. **Images in frequency and spatial domain.** The left pair shows two sparse examples in frequency domain and their spatial counterparts. On the right, two images with sparse spatial data are illustrated together with their data in frequency domain. For visualization purposes, the grey levels are on an inverse logarithmic scale and the axes of the left two images in frequency domain are scaled by a factor of 40.

Fig. 3.5. **Image filtered with lowpass and highpass.** The source image is shown in the middle. The lowpass filter result on the left has suppressed high frequency details while the highpass filter used for the right image keeps only high frequency information.

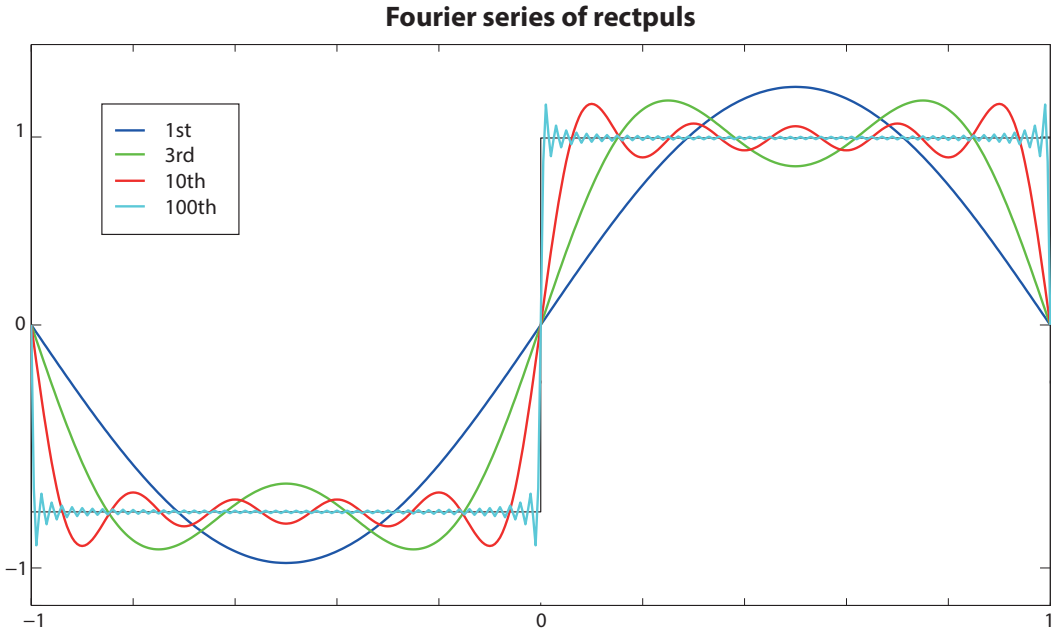


Fig. 3.6. **Gibbs phenomenon.** Due to the restriction on a finite number of coefficients in the Fourier basis, artifacts appear at simple discontinuities such as the sharp edges of this square wave. Illustrated is an approximation with a Fourier series approximation of increasing eigenfunction cardinality from one basis coefficient up to 100.

Theorem 3.1 (Convolution theorem)
The relation between convolution and discrete Fourier transform is being given by

$$(f \star g)(x,y) \circ\!\!-\!\!- F(u,v) \cdot G(u,v). \tag{3.14}$$

Proof. We prove this in the theoretical case using Laurent series

$$\mathscr{F}((f \star g)(x,y)) = \sum_{x=-\infty}^{\infty} \sum_{y=-\infty}^{\infty} (f \star g)(x,y) \exp\left(-2\pi i \left(\frac{xu}{h} + \frac{yv}{w}\right)\right) \tag{3.15}$$

$$= \sum_{x,y} \sum_{s,t} f(s,t)\, g(x-s, y-t) \exp\left(-2\pi i \left(\frac{xu}{h} + \frac{yv}{w}\right)\right) \tag{3.16}$$

$$= \sum_{s,t} f(s,t) \sum_{x,y} g(x-s, y-t) \exp\left(-2\pi i \left(\frac{xu}{h} + \frac{yv}{w}\right)\right) \tag{3.17}$$

$$= \sum_{s,t} f(s,t) \exp\left(-2\pi i \left(\frac{su}{h} + \frac{tv}{w}\right)\right) \tag{3.18}$$

$$\sum_{x,y} g(x-s, y-t) \exp\left(-2\pi i \left(\frac{(x-s)u}{h} + \frac{(y-t)v}{w}\right)\right) \tag{3.19}$$

$$= \sum_{s,t} f(s,t) \exp\left(-2\pi i \left(\frac{su}{h} + \frac{tv}{w}\right)\right) \sum_{x,y} g(x,y) \exp\left(-2\pi i \left(\frac{xu}{h} + \frac{yv}{w}\right)\right) \tag{3.20}$$

$$= F(u,v) \cdot G(u,v). \tag{3.21}$$

□

This means that the discrete Fourier transform of a convolution is a point-wise product of Fourier transforms. Instead of calculating a convolution directly, we can use a DFT to transform the image and its kernel first, calculate the product and do an IDFT afterwards. The actual process of filtering becomes a point-wise multiplication in Fourier space.
Is this detour worth it? Surely this is not necessary, if we want to use just one small mask to filter the image. However, for large kernels or multiple masks a noteworthy acceleration can be performed using discrete **fast Fourier transform** (FFT). For an image of size $n \times n$ and a kernel of size $r \times r$ with $r < n$ the computational complexity of a convolution in spatial domain measured by the number of multiplications and summations is $\mathcal{O}\left(r^2\right)$ per pixel. With the computationally-efficient FFT algorithm and n being an adequately suitable number such as a power of 2, FFT and its inverse can be done with $\mathcal{O}\left(2n^2 \log_2(n)\right)$ operations on the entire image.[7] Table 3.1 shows the complexity of the different filter methods. Note that the limit complexity of the latter is independent of filter parameters and the point-wise multiplication dominates the calculation cost for large n. Fig. 3.7 shows the computation time for both methods on a 640 × 640 image with a kernel of increasing size. The independence of the calculation time from the size of the kernel for the FFT-method is evident.

Method	**Process**	**Complexity**
Convolution	$k \star g = g_{new}$	$\mathcal{O}\left(n^2 r^2\right)$
FFT	$k \star g \circ\!\!- F \cdot G = G_{new} -\!\!\circ g_{new}$	$\mathcal{O}\left(n^2 + 4n^2 \log_2(n)\right)$

Tab. 3.1. Complexity of different filter methods for $n \times n$ image with kernel of size r. While the computational complexity of a direct calculation depends quadratically on both filter and image size, the cost with the use of a fast Fourier transform is only dependent on the image dimension. This can be a relevant advantage for large kernels.

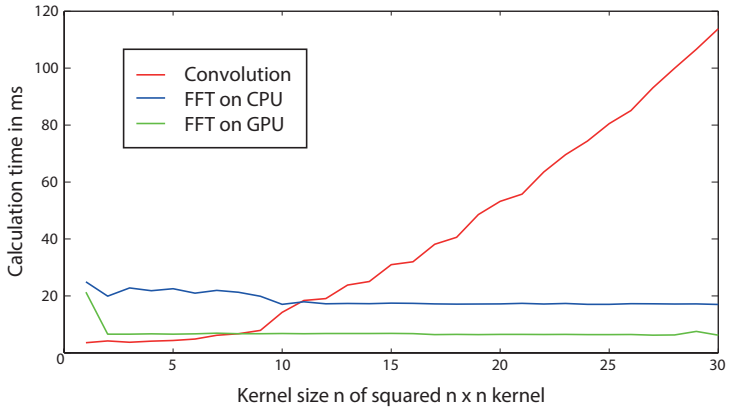


Fig. 3.7. Calculation time for different filter methods and varying kernel size. The calculations are performed using MATLAB R2012b on Windows 7 64bit on a machine with an AMD FX-8120 Eight-Core Processor at 3.10 GHz with 8 GB RAM using an NVIDIA GeForce GTX 550 Ti with 192 CUDA Cores and 1024 MB GDDR5 RAM. The reported time measurements are averaged over 100 runs.
While a direct calculation is faster for small kernels, using FFT speeds up the calculation for larger filter sizes. The independence of the computational complexity from the kernel size is clearly visible.

[7] A 1D FFT requires $\mathcal{O}\left(n \log_2(n)\right)$, see Deuflhard and Hohmann [87, pp. 223–226]. A 2D FFT on an $n \times n$ image requires $2n$ 1D FFTs, one for every column and row, see Bourke [35].

Henceforward, it is also clear that the term filtering refers to the frequency domain of this process where certain frequencies can be accepted or rejected. In order to denoise and blur an image, we introduce a kernel to suppress high frequencies that use the neighbourhood information equally. We do this with an isotropic discrete **Gaussian kernel** whose values are calculated at the pixel centres according to the standard formula

$$K(x,y) = Ae^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.22}$$

with height A and standard deviation σ.
This gives an ideal filter with respect to the artifacts mentioned beforehand, since the Gaussian has an optimal localization in both domains - its Fourier transform is again a Gaussian. Potentially ideal filters in contrast suffer from ringing effects as Fig. 3.8 demonstrates.

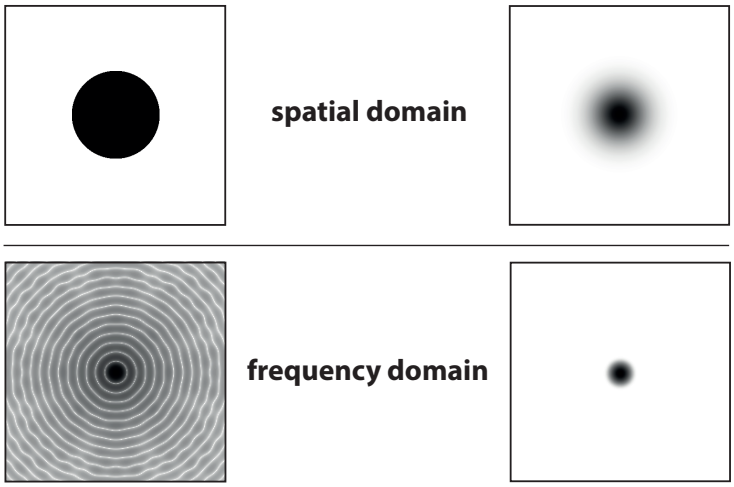


Fig. 3.8. **Idealized filter vs. Gaussian filter.** The circular spatial filter shown on the left causes fringing in frequency domain while the Gaussian filter illustrated on the right has a much better localization in both domains. For visualization purposes, the axes are scaled by the factor 4 and the grey levels are on an inverse logarithmic scale for the frequency domain.

Filtering can be used for a variety of image processing tasks such as local image feature intensification or edge sharpening with numerous discretized operators, and a solid background is fundamental for various machine vision applications. For further application, Gonzalez et al. [151, pp. 152–191, 269–303] give a lucid introduction.
Equipped with denoised and prepared image data, we focus now on the image content and formulate algorithms to localize potentially interesting parts and regions. We begin by classifying image subdomains.

3.4. Segmentation

The important parts of an image are often not displayed on the full screen and further processing is required to find the **regions of interest** (ROIs). The probably most intuitive way of segmenting object and background is by thresholding its grey values within a brightness interval.

3.4.1. Band Thresholding

Definition 3.5
The region of interest R of a ***band thresholding*** *operation on an image with domain D is given by*

$$R = \left\{(x, y) \in D \mid g_{min} < g_{x,y} < g_{max}\right\}. \tag{3.23}$$

This is of great use, especially if the grey values of objects and their background differ considerably. In case of illumination changes during the measurement, a proper contrast normalization as presented in section 3.2 should be taken into account.
Unfortunately, finding proper values for g_{min} and g_{max} can be cumbersome and cause precalculation steps. A way to circumvent this is by finding just one basic threshold parameter to distinguish between object and background if their grey values differ considerably.[8]

3.4.2. Basic Thresholding

Definition 3.6
The region of interest R of a ***basic thresholding*** *operation on an image with domain D is given by*

$$R = \left\{(x, y) \in D \mid g_{x,y} < g_{thresh}\right\} \tag{3.24}$$

or $g_{x,y} > g_{thresh}$ if the object is brighter than the background.
This leaves just one parameter for the operation which separates the grey values. According to our assumption, there should be two maxima within the image histogram, which we might have already calculated for the contrast normalization. The minimum in between separates object and background. Fig. 3.9 shows that this can be problematic since the uniqueness of these extreme values is not guaranteed, but a 1D Gaussian filtering of the histogram solves this problem. The result (with a Gaussian of standard deviation $\sigma = 5$) can be seen in Fig. 3.10 and seems to be a reasonable choice for our example. A lot more advanced classical segmentation techniques for miscellaneous application scenarios can be found in the work of Šonka et al. [389, pp. 175–327]. More recent approaches like the work of He et al. [166] perform learning based object instance segmentation by extending powerful two stage detectors with a parallel second stage prediction of a binary mask at the cost of computational complexity which may not be applicable to high-speed applications or in case of limited computational bandwidth.

Depending on the type of object and particular colour changes in the background region, the result of a segmentation can still be noisy. Fig. 3.11 shows such an example where using a Gaussian filter of high standard deviation σ would result in loosing important image information. To finally get only the sought ROIs (e.g. the discs in the upper part of the image), further processing may be needed. We will solve problems like this with the help of morphological operators which we introduce in the following section.

[8] Cf. Šonka, Hlavac, and Boyle [389, pp. 176–180].

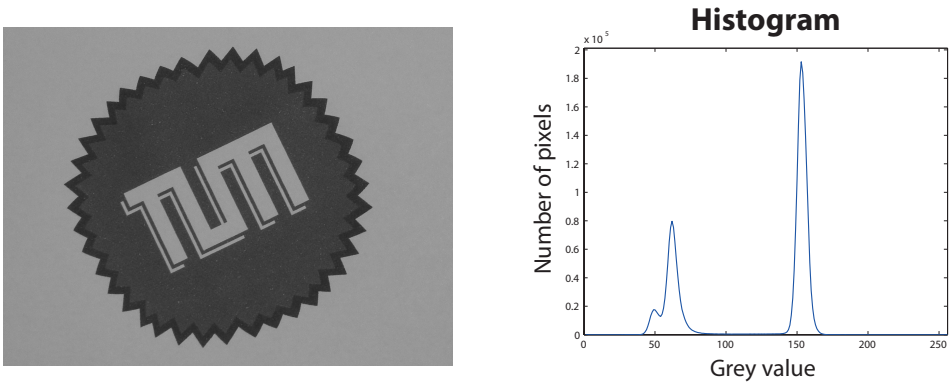


Fig. 3.9. **Image and its histogram.** The source image is shown on the left while it is visible from the histogram on the right that most grey values distribute around the background grey value and the foreground brightness. Multiple local minima are visible.

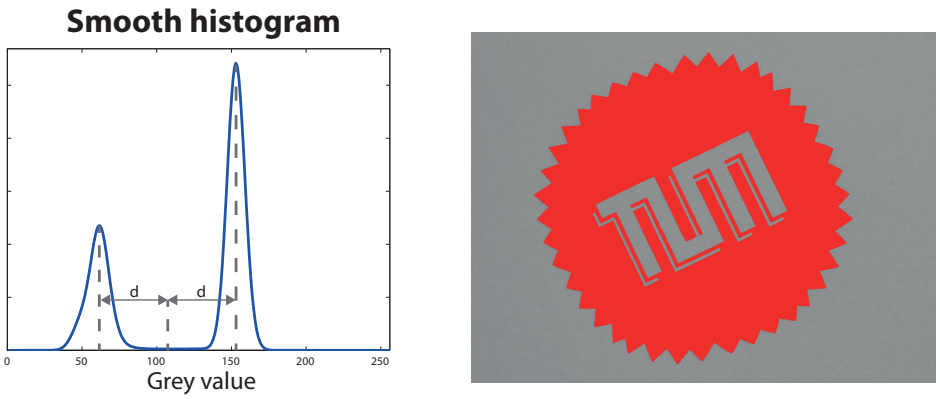


Fig. 3.10. **Smooth histogram and segmented image.** Only two local minima remain in the filtered histogram on the left. A segmentation of the image is possible. The mask is depicted in red on the right.

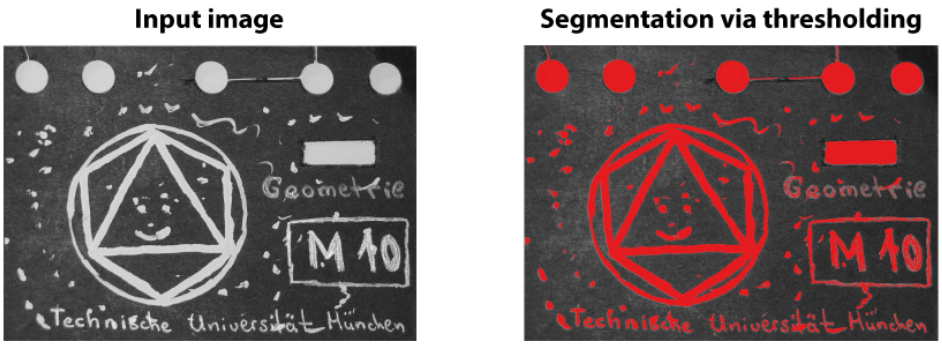


Fig. 3.11. **Segmentation clutter.** The task is to mask the discs in the upper part of the image. The proposed segmentation pipeline alone is incapable of removing the clutter in the background and thus the mask shown in red contains more pixels.

3.5. Morphology

The modification of regions for the selection of important image parts is an indispensable tool for the implementation of effective algorithms. In this section, we deal with the mathematical operations to tackle such tasks. Besides the standard operations from set theory, namely union $A \cup B$, intersection $A \cap B$, difference $A \setminus B$, and complement $\overline{A}$, we therefore introduce a new geometric operation, the translation.[9]

Definition 3.7
*The **translation** of a region A by a vector $t \in \mathbb{Z}^2$ is given by*

$$A_t = \{a + t \mid a \in A\}. \tag{3.25}$$

Fig. 3.12 shows an example of this operation.
With these five elementary set operations, we develop the basic operators of mathematical morphology.

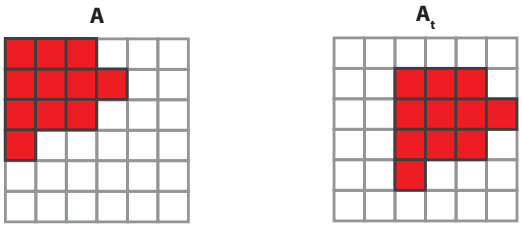


Fig. 3.12. **Translation of region A with $t = (2,1)^T$.** The selected region A in red on the left travels with this operator in the direction of t. The result is shown on the right.

3.5.1. Erosion and Dilation

In typical morphological tasks, there are two sets involved. One is the underlying ROI R of some image and the other one, S, represents the shape we are interested in; it is called the **structuring element** (SE). Mostly, this is a symmetric domain around a special point O (its origin).
The first two morphological operators we define are erosion and dilation.[10]

Definition 3.8
*The **erosion** of a region R by the structuring element S is given by*

$$R \ominus S := \bigcap_{s \in S} R_{-s}. \tag{3.26}$$

[9] Cf. Steger, Ulrich, and Wiedemann [395, p. 126].
[10] Cf. Soille [386, pp. 65, 68].

*The **dilation** of a region R by the structuring element S is given by*

$$R \oplus S := \bigcup_{s \in S} R_{-s}. \tag{3.27}$$

Since

$$R \ominus S = \bigcap_{s \in S} R_{-s} = \left\{ t \in \mathbb{Z}^2 \mid S_t \subseteq R \right\} = \left\{ p \in \mathbb{Z}^2 \mid p + s \in R \ \forall s \in S \right\} \subseteq R, \tag{3.28}$$

an erosion is essentially a shrinking of the region R by putting the origin of S onto an arbitrary pixel p of $R \ominus S$. Thus, S is completely covered by R. See Fig. 3.13 for a simple example.

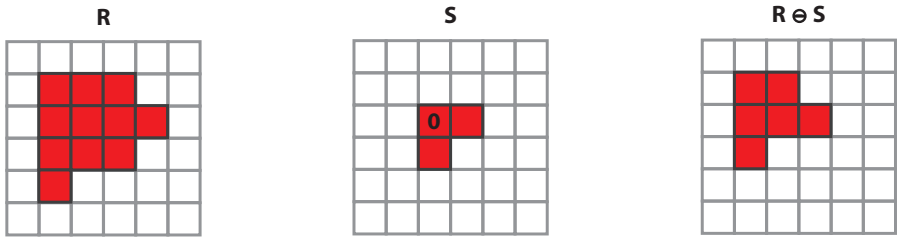


Fig. 3.13. **Erosion of R by S.** The red region R on the left is eroded with the structuring element S in the middle. The reduced region is shown on the right.

In an analogous manner, a dilation can be written as

$$R \oplus S = \bigcup_{s \in S} R_{-s} = \left\{ t \in \mathbb{Z}^2 \mid R \cap S_t \neq \emptyset \right\} = \left\{ p \in \mathbb{Z}^2 \mid \exists s \in S : p - s \in R \right\} \supseteq R. \tag{3.29}$$

This means that a dilation lets the region expand by putting the origin of S onto an arbitrary pixel p of R. Thus, S is completely covered by $R \oplus S$. Fig. 3.14 depicts this fact.

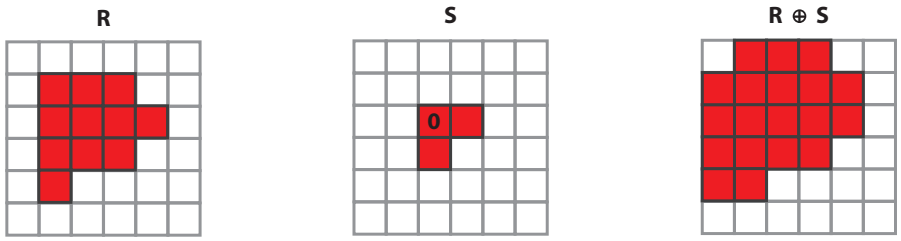


Fig. 3.14. **Dilation of R by S.** The red region R on the left is dilated with the structuring element S in the middle. The expanded region is shown on the right.

As long as not defined specifically, S shall be a centrally symmetric discretization of a two-dimensional disc around O. A first example of the given operations on a more complex region is illustrated in Fig. 3.15. An erosion by S reduces the ROI and deletes isolated small domains, whereas a dilation enlarges the ROI and fills gaps within separated image parts.

From an application viewpoint it would be preferable to have the denoising and cancelling effects of an erosion without a reduction of the main areas. Doing so will help to solve prob-

Fig. 3.15. **Erosion and Dilation of R with the structuring element S.** On the left, both the region R and the structuring element S are shown. The result of an erosion is drawn in the middle in comparison with the region R while the domain after dilation is illustrated on the right.

lems like the one pointed out in Fig. 3.11. This can be done by successive execution of both operations, erosion and dilation.[11]

3.5.2. Image Opening

Definition 3.9
*The **opening** of a region R by the structuring element S is given by*

$$R \circ S := (R \ominus S) \oplus S. \tag{3.30}$$

Although the opening is defined in terms of erosion and dilation we can write

$$R \circ S = (R \ominus S) \oplus S = \bigcup_{S_t \in R} S_t, \tag{3.31}$$

which gives a geometric formulation of the operation, since this is essentially the same as moving the structuring element S along every pixel of R and examining whether it fits the region or not. Schematically this works as shown in Fig. 3.16. We note that by choosing a disc as SE, R is rounded by S from the inside.

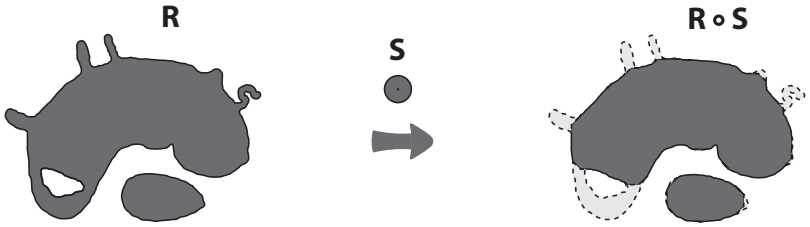


Fig. 3.16. **Opening of R by S.** The successive application of erosion and dilation with the structuring element S transforms the region R on the left into the dark region depicted on the right. The original region is added for comparison in brighter grey.

This indeed solves the issues from Fig. 3.11 which can be seen in Fig. 3.17. Getting rid of the last wrongly selected larger region can then be done by some further parameter filtering to

[11] Cf. Soille [386, pp. 105–106].

which we come in the subsequent sections. At first, however, we focus on the separation of the unconnected parts.
Last but not least, it also has to be mentioned that the defined morphological operations are considerably more costly in terms of their computational complexity compared to the operations and methods presented beforehand. This means, if it is possible we try not to apply the operations on the entire image but rather on some small restricted regions of interest to save computation time and facilitate real-time tasks.

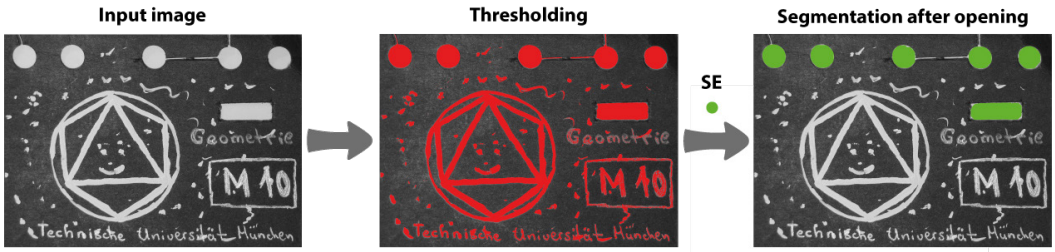


Fig. 3.17. **Selection with opening.** The task is to mask the discs in the upper part of the image. The morphological operation with the structuring element SE shown in green removes the cluttered parts of the red mask in the middle such that the green mask on the right remains.

3.6. Separation

We are now able to detect a set of pixels for a certain ROI. So far, these pixels do not have to be connected, which yields one single set of image coordinates without labelling its non-connected subsets. This works perfectly if we are looking for one connected domain within an image, but this is often not the case. On the contrary, in many tasks we are not only interested in finding or selecting one single object or region within the image. Often several visible spots are relevant for further processing.
We discuss hereafter methods that separate image regions and allow us to label different domains.

3.6.1. Connectivity

If we want to think about **connected components** of a region, we have to determine how it can be defined that two pixels are connected. The natural way of speaking about connection can be formulated in terms of the neighbourhood Definition 2.4, where two pixels are said to be connected if they either share a vertex or an edge.

Definition 3.10
Two pixels p_1 *and* p_2 *are* ***4-connected*** $\iff p_1 \in U_{von\ Neumann}(p_2)$.
Two pixels p_1 *and* p_2 *are* ***8-connected*** $\iff p_1 \in U_{Moore}(p_2)$.

As this definition is obviously symmetric, the ordering of p_1 and p_2 is negligible.
One peculiarity can be noted if we use the same classification for both foreground and background.[12] This issue is illustrated in Fig. 3.18. We would intuitively separate the background into two areas, one enclosed by the circle and one outside of it. However, if we use 8-connectivity for the background, it would consist of only one single connected region. On the other hand, if we use 4-connectivity for the foreground, the circle would not be regarded as a connected region. To avoid this counterintuitive behaviour, we use different connectivity classifications on both sets. **8-connectivity** defines components **in the foreground** whereas **4-connectivity** is used **for the background**.

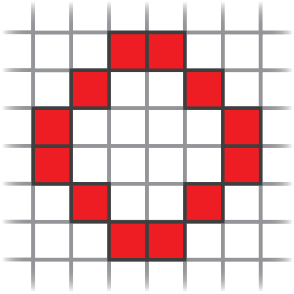


Fig. 3.18. **Thin foreground circle.** Depending on the choice of the neighbourhood, the inner circle is connected with the exterior part or not. Choosing for instance a Moore neighbourhood defines only one connected white region while 4-connectivity yields separate regions. The red circle is 8-connected.

This allows us to finally **separate** the **connected components** of Fig. 3.17. Now that every component has an individual label, we could just look at the number of pixels within every area and get rid of the larger domain which exceeds a specific threshold. Unfortunately this depends on the camera resolution and the camera-object distance. However, in section 3.8 we find ways to describe the geometry of the connected components more specifically; these can be used as distance-invariant filter options.

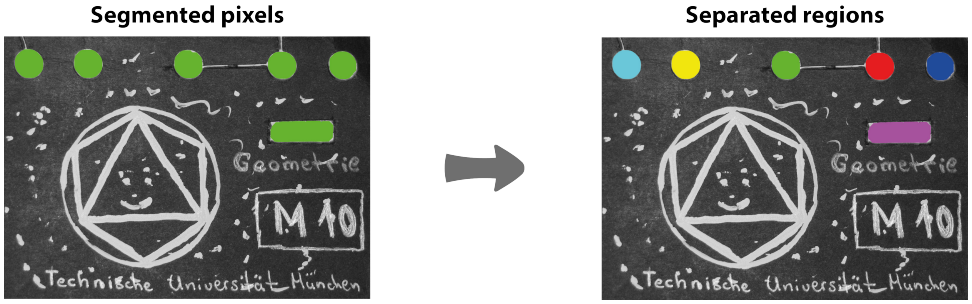


Fig. 3.19. **Separation of regions.** While the green mask after segmentation (shown on the left) is not separated, each individually coloured component (shown on the right) is labeled separately.

The methods presented so far give powerful tools for plenty of applications. However, we still work on the pixel level. In the next sections, we leave this level and formulate solutions to collect more abstract region parameters starting with a definition of an edge in an image.

[12] Cf. Umbaugh [428, pp. 109–110].

3.7. Edge Detection

Catching a glimpse of an image for a short time reveals the main information flow obtained by the human visual system. Remembering only a fraction of the depicted scene demonstrates that the observer often stores only a small amount of the image content at first glance.[13] This can be type and position of the foreground objects and some other high-level image features. To **separate** this **information**, a significant change of the image intensity at particular spots is needed: the edges.
For localization of edges and to detect contours, it is essential to understand, how an edge can be defined. We differentiate between 1D edges along curves and 2D edges.

3.7.1. Edges along Curves

As a start, we think about the problem in a simplified manner and neglect discretization effects. This allows us to use differential operators. For an edge detection along a curve as illustrated in Fig. 3.20, this gives the following definition.

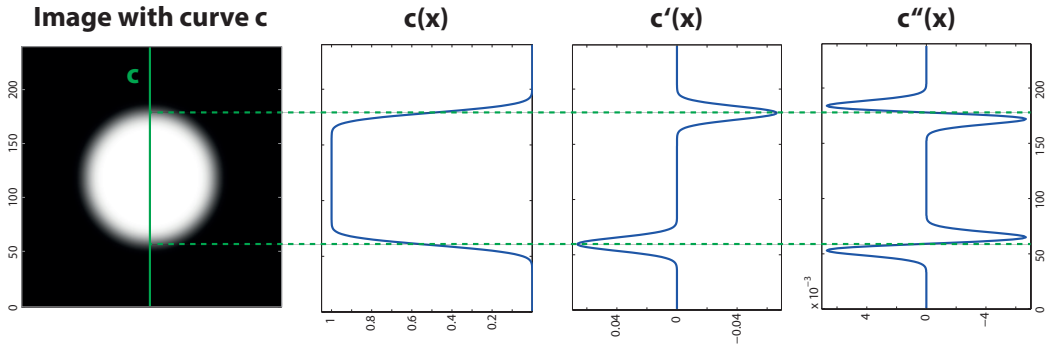


Fig. 3.20. **Idealized sample with derivatives along curve.** The original image is shown on the right together with a curve c depicted in green. The normalized grey values along the curve are given by the plot $c(x)$ while c' and c'' illustrate the derivatives. It is clearly visible that the extreme values of c' (and respectively the zero crossings of c'') localize the boundary of the circle.

Definition 3.11
The set E of ***edges along a curve*** $c : I \to A$ *in an image* $f : D \to \mathbb{G}$ *is given by*

$$E = \left\{ a \in A \mid a = c(e) : \left|c'(e)\right| > C \wedge \left|c'(e)\right| > \left|c'(x)\right| \ \forall x \in B_r(e) \right\}, \tag{3.32}$$

where $I \subset \mathbb{R}$ is an interval, $A \subset \tilde{D} \subset \mathbb{R}^2$ with the continuation $\tilde{D}$ of the discrete image domain D. Moreover, $C \in \mathbb{R}$, $C \gg 0$ is a constant, c is a unit speed curve, and $B_r(p)$ symbolizes a small ball of radius $r > 0$ around $p \in I$.

[13] The storage capacity of the human short term memory for familiar content consists of not more than seven (plus or minus two) storage units, see Microsoft [282]. Catchword: Gedächtnis, 2.2.b. Speicherkapazität für Kurzzeit- und Arbeitsspeicher.

The first condition guarantees that the grey value change is significant since $|c'(e)| \gg 0$. It automatically neglects points that differ just by local lighting variations or by problematic colour fidelity. We call the second condition **non-maximum suppression**. It ensures that the detected edge point is locally unique. We choose c to be of unit speed since this allows us to easily measure distances on the curve later on.

An alternative definition for edges along one-dimensional manifolds in application tasks is often done using the second derivative as Fig. 3.20 suggests. In this case one may call a point an edge point if $c'' = 0$. Two major issues occur in this case: firstly, the existence is not even guaranteed for simple discrete cases. This can be avoided using a soft upper threshold instead of 0. Secondly, however, flat inflection points are wrongly detected as edge points - this would need further processing.[14]

If we want to **apply our definition to real images**, we have to discretize the differential operator and interpolate the grey value on the curve if it does not coincide with a pixel centre. The discretization of the differential operator can be done in a symmetric way as shown in Fig. 3.21, so that

$$\frac{d}{du}(c(u)) \approx \frac{1}{2}(c(u+1) - c(u-1)). \tag{3.33}$$

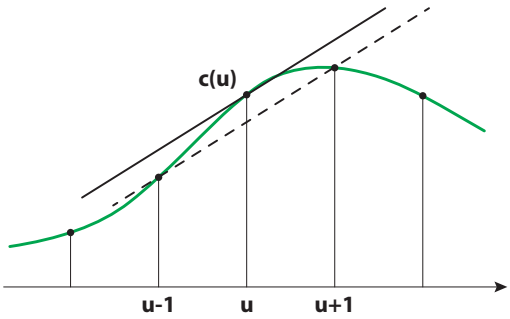


Fig. 3.21. **Discretization of differential operator d/du.** The green continuous curve is evaluated at the discrete points shown in black. The values of $c(u-1)$ and $c(u+1)$ are used for a linear approximation of the derivative at $c(u)$.

With the methods described in section 3.3, we note that this can also be expressed as a 1D convolution (note the mirroring):

$$\frac{d}{du}(c(u)) \approx \left(\frac{1}{2} \cdot \begin{pmatrix} 1 & 0 & -1 \end{pmatrix}\right) \star c(u). \tag{3.34}$$

For convenient reasons and since we are mostly interested in relative comparisons of differential values, we will drop the constant pre-factor and stay with the kernel $\begin{pmatrix} 1 & 0 & -1 \end{pmatrix}$. Our **discrete** formalization of the **differential operator** is therefore given by

$$\mathrm{D}\left[\frac{d}{du}c(u)\right] = \begin{pmatrix} 1 & 0 & -1 \end{pmatrix} \star c(u), \tag{3.35}$$

where D represents the discretization operator.

[14] Cf. Burger and Burge [46, p. 126].

How do we now get the grey value on curve points that lie not directly on the discrete pixel grid? If we have a certain position p for such a point, a virtual pixel centred at p would touch four pixels of the grid underneath as shown in Fig. 3.22. A direct way to a pixel value would be to look for the closest pixel centre and take its grey value: namely **nearest-neighbour interpolation**, which can cause unwanted artifacts.[15] Using the fictive coordinates of p, we can do a linear interpolation in both the vertical and horizontal direction to retrieve its **bilinear interpolation** that gives a scalar at its centre which we assign to the closest value within the pixel depth. With the distances given in Fig. 3.22 we can write this interpolation as

$$g_p = b\,(a\,g_{11} + (1-a)\,g_{01}) + (1-b)\,(a\,g_{10} + (1-a)\,g_{00})\,. \tag{3.36}$$

The effect of this interpolation is visualized in Fig. 3.23 for an edge detection task with a real image. A smoothing of the discrete grey value function would also be done for denoising purposes before calculating the width of such a component part.
In the following, we use these one-dimensional concepts to develop detection methods for two-dimensional edges.

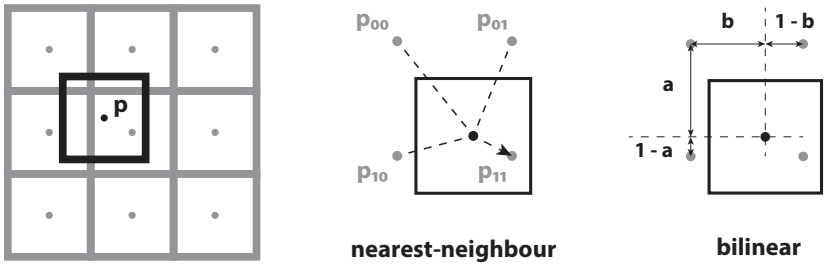


Fig. 3.22. **Parameters for nearest-neighbour and bilinear interpolation.** The sought value at point p not coinciding with the discrete pixel grid shown on the left can be calculated either via nearest-neighbour interpolation, where the value of the closest discrete pixel centre is adopted (in this case p_{11} in the middle) or bilinear interpolation is used with the distances shown on the right.

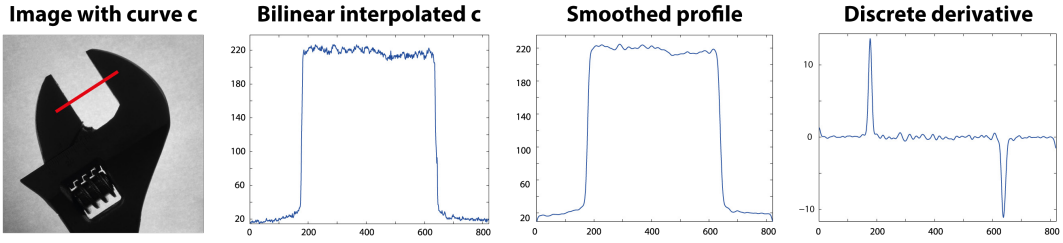


Fig. 3.23. **Edges along curve using bilinear interpolation.** The intensity values of the red linear segment c shown on the left are calculated with bilinear interpolation. The local noise level is reduced by filtering. This can be observed by comparison of the jitter between the two middle images. Calculating the discrete derivative clearly reveals the two edge points.

[15] Cf. Steger, Ulrich, and Wiedemann [395, pp. 96–99].

3.7.2. Image Contours

An image contour or a **two-dimensional edge** is a curve s that has a one-dimensional edge perpendicular to itself. That means if we want to use the concepts of section 3.7.1 to describe contours, we need to look for points whose gradient magnitude is locally maximal in direction of the gradient.[16] In the continuous case, this can be expressed formally by the definition below.

Definition 3.12
*The set R of **contour points** of an image $f : D \to \mathbb{G}$ is given by*

$$R = \{p \in D \mid p \text{ is edge along line } c \wedge c \parallel \nabla f\,(p)\}\,, \tag{3.37}$$

where $D \subset \mathbb{R}^2$ and $\nabla = \left(\partial_x, \partial_y\right)^T$ gives the 2D nabla operator of partial derivatives.

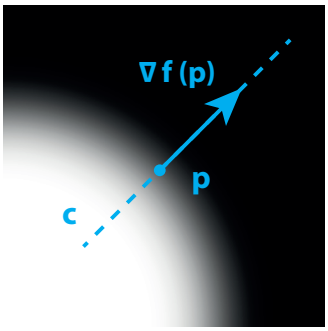


Fig. 3.24. **Contour point with assigned curve.** The curve c is illustrated as a dashed blue line. It can be seen that the derivative $\nabla f\,(p)$ is parallel to the curve at its edge p.

In Fig. 3.24 one such contour point p of the image with an example curve c and its edge p are shown.
We now want to reformulate the definition above for our discrete purposes. A first step therefore is to apply a Gaussian smoothing. This decreases the influence of noise for further processing steps and scatters highly localized image features within kernel-sized areas.
Following the same principles as in the one-dimensional case, we can define the components of the **discrete gradient** on the 2D grid as a convolution with the kernel $\begin{pmatrix} 1 & 0 & -1 \end{pmatrix}$ for the X-direction and with $\begin{pmatrix} 1 & 0 & -1 \end{pmatrix}^{\mathrm{T}}$ for Y respectively. In order to receive reliable values, a smoothing perpendicular to the gradient seems convenient.[17] To do so, we introduce the **Sobel operator** in both directions by its kernels

$$S_x = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \quad \text{and} \quad S_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}. \tag{3.38}$$

[16] Note also that a gradient direction may be noisy within areas of similar grey values.
[17] Cf. Jähne [191, pp. 350–351].

Neglecting constant pre-factors again, this allows us to formulate a **discrete** version of the **image gradient** as

$$\mathrm{D}[\nabla f\,(p)] = \begin{pmatrix} S_x \star f\,(p) \\ S_y \star f\,(p) \end{pmatrix}. \tag{3.39}$$

In Fig. 3.25 the absolute value of this operation can be compared to the original image. What is more, it can be observed that the gradient direction is noisy in areas without contours whereas it is similar along edges.

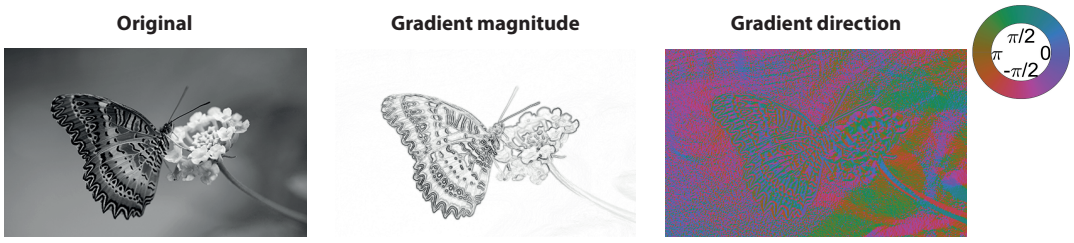


Fig. 3.25. **Grey value image and its gradient image.** On the left is the source image. For the visualization of the gradient magnitude, the vector length is represented by the inverted grey scale. For the image gradient direction, the angle interval $(-\pi, \pi]$ from the positive x-axis is assigned to the colour code shown. While the gradient magnitude in the middle highlights the image contours, it can be observed that the gradient direction on the right is noisy in non-contour areas.

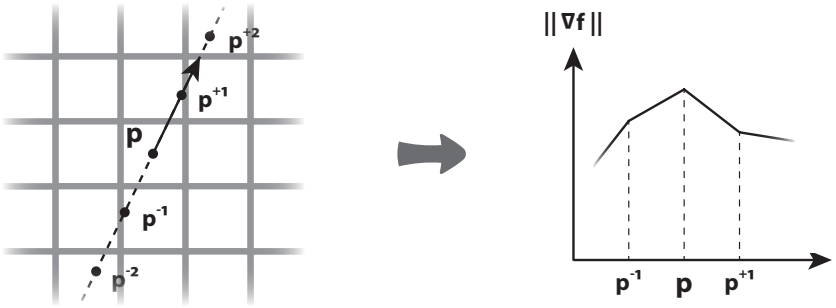


Fig. 3.26. **Edge detection with interpolation along gradient direction.** On the left, the gradient $\nabla f\,(p)$ at p (visualized with an arrow) is used to calculate the direction which is illustrated as a dashed black line. Samples along the line are taken at discrete steps forwards (+) and backwards (-). The right graphic illustrates the interpolated gradient magnitude for some example locations.

If we directly adopt the principles of detecting edges along curves we can now use the gradient $\nabla f\,(p)$ as the direction for a line through p and calculate the local maximum on this line using bilinear interpolation and non-maximum suppression along the gradient direction in order to detect two-dimensional edges as illustrated schematically in Fig. 3.26. The process of edge detection for a certain pixel would then be twofold. Firstly, we calculate the gradient, and secondly, we search for peaks in the direction of the gradient. This would mean that we have to do at least two calculations for fictive pixel centres and the corresponding bilinear interpolations for each considered pixel within the image besides the prior gradient calculation. Since it is essentially a binary decision if either pixel p is an edge point or not given its particular neighbourhood, we instead use a similar, more pixel-based approach to detect edges. After the calculation of the discrete gradient, we **separate the neighbourhood of p into four angle sectors** based on the possible gradient directions according to neighbourhood pixels. The

decision process is then reduced to a comparison of the gradient magnitude of the two relevant pixels within the 8-neighbourhood as shown in Fig. 3.27.

For some tasks, especially if the position of corner points is important, an edge detection with the Laplace operator $\Delta = \nabla \cdot \nabla$ can also be carried out. The condition for corner points would then be $\Delta f\,(p) = 0$. Since this approach is sensitive to noise, further ideas as proposed by Šonka et al. [389, pp. 138–142] have to be taken into account. For our purposes, it is important that the edge detection is robust and accurate. Thus, we keep the discretization simple and use only first derivatives here without specific corner detection.
Fig. 3.28 shows a detailed part of the image gradient whose contour pixels are detected with this method. Specific attention has to be paid to the result on the right which has significantly thinner contours due to the non-maximum suppression.

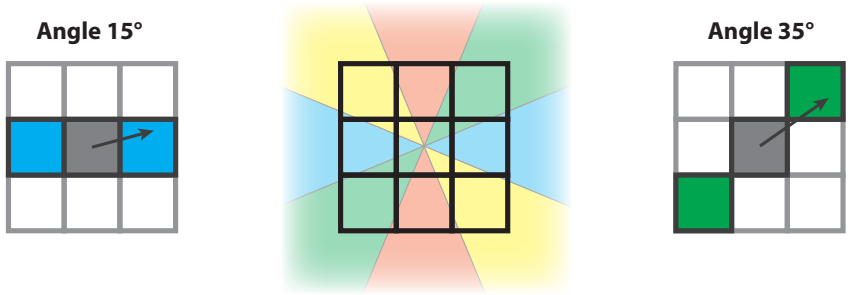


Fig. 3.27. **Pixel-based non-maximum suppression.** In the middle, it is illustrated that the neighbourhood of the centre pixel is separated into four angle sectors which are coloured blue, green, red, yellow. On the left, a query gradient - illustrated by the arrow - with an angle of 15° to the horizontal line falls into the blue sector while the right query point with an angle of 35° is assigned to the green bin. In order to suppress non-maximal values, only a comparison with the gradients in the coloured pixels is necessary.

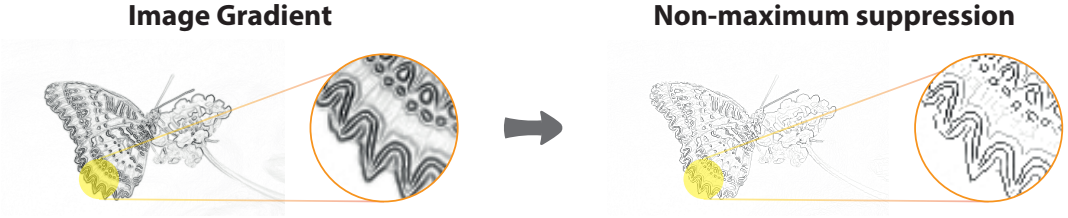


Fig. 3.28. **Image gradient before and after non-maximum suppression.** The left image depicts the image gradient and a magnification. Note the spurious points. After edge thinning with non-maximum suppression, the contours are more clearly defined as shown on the right.

3.7.3. Hysteresis Thresholding

Until now, we have only used one single threshold C for the segmentation of edge points by their gradient magnitude $\|\nabla f\,(p)\|$. This often gives results that are not yet satisfying if the grey level of the object is similar to noisy features or irrelevant background parts. If we use for example a high threshold $C_{up} > C$, we probably get the relevant edge points but it is likely that they stay fragmented. On the other hand, using a low threshold $C_{low} < C$ can give all relevant

edge points but also some irrelevant pixels. This drawback has already been tackled in the early days of computer vision by Canny [60] with the use of two thresholds for **hysteresis thresholding**.[18] For our purposes, we propose Algorithm 3.3.

Algorithm 3.3. Hysteresis Thresholding

Input parameters:

- Image $f : D \to \mathbb{G}$
- Selected pixel set $S \subset D$ by non-maximum suppression
- Lower threshold C_{low}, upper threshold $C_{up} > C_{low}$

Computation steps:

1. Calculate $\|\nabla f\,(p)\| \;\; \forall p \in S$
2. Mark pixels as
   - wrong, if $\|\nabla f\,(p)\| < C_{low}$
   - potential, if $C_{low} < \|\nabla f\,(p)\| < C_{up}$
   - correct, if $\|\nabla f\,(p)\| > C_{up}$
3. Set $P := \{p \in S \mid p \text{ potential}\}$ and $R := \{p \in S \mid p \text{ correct}\}$
4. Detect contour. $\forall p \in R$:
   - Calculate neighbourhood intersection $I = U\,(p) \cap P$
   - If $I \neq \emptyset$ add $p \in P \cap I$ to R

Output:

- Contour pixel set R

Since 8-connectivity defines foreground objects[19] we use $U = U_{Moore}$ for the neighbourhood intersection. This band thresholding process then guarantees that potential pixels are only considered if they are connected to already correct ones. The hereby detected individual pixels still need to get linked to one another in order to form a thin contour line. This is done by repeatedly selecting a first pixel centre of some $p \in R$ and successively looking for adjacent pixels until the end of a contour is hit, the contour closes, an intersection point of two contours is reached or no unprocessed pixels remain in R. An example of this algorithm is shown in Fig. 3.29.

For many application tasks, an algorithm using gradient calculation via Sobel kernels, non-maximum suppression and a hysteresis threshold gives sufficient results and works robust. An example for such a processing chain is illustrated in Fig. 3.30, where the colour image is mapped to grey scale following the recommendation ITU-R BT.601-7 of the International Telecommunication Union BT [43].

$$f\,(x,y) = 0.2989 \cdot \mathrm{R}\,(x,y) + 0.5870 \cdot \mathrm{G}\,(x,y) + 0.1140 \cdot \mathrm{B}\,(x,y), \tag{3.40}$$

[18] Cf. Schaeffel et al. [364, pp. 601–602].
[19] Cf. section 3.6.

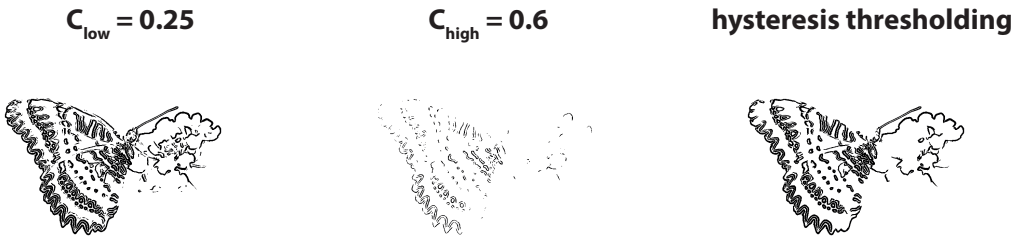


Fig. 3.29. **Hysteresis thresholding on smoothed gradient image scaled to** $[0,1]$**.** The left two images show selected gradient images with different thresholds on the gradient magnitude. While the low threshold C_{low} delivers connected relevant edges, clutter remains. The high threshold C_{up} includes only relevant edge points, but they are fragmented. The hysteresis thresholding on the right involves both thresholding operations and gives a clean and connected result.

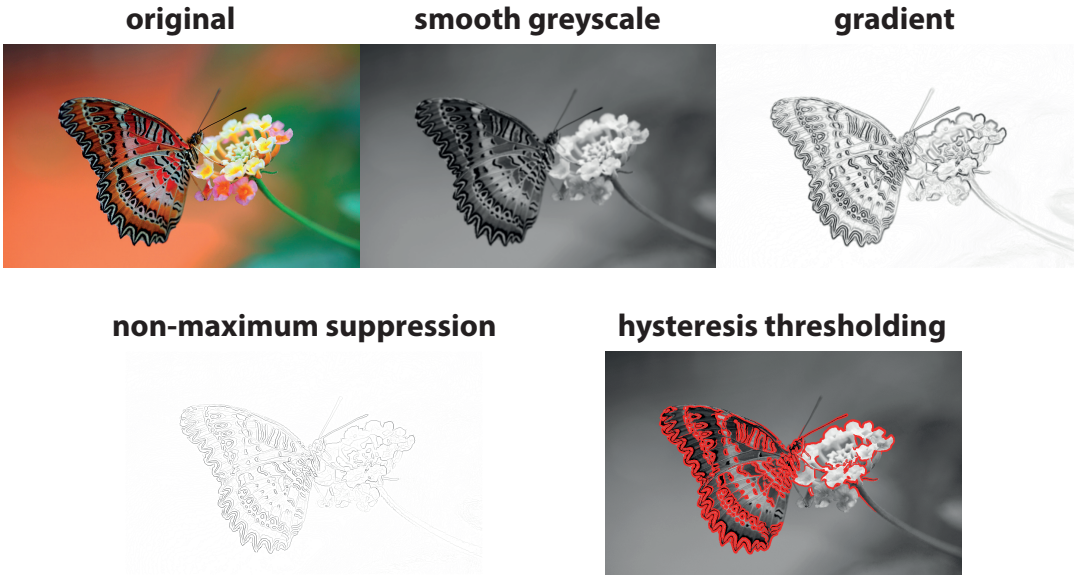


Fig. 3.30. **Edge detection chain.** The original colour image on the upper left is mapped to grey scale according to the standard BT [43] and filtered with a Gaussian kernel of size 5×5 with standard deviation $\sigma = 1.4$. The resulting gradient image on the upper right is processed with a non-maximum suppression for edge thinning (see lower left). A hysteresis thresholding is performed with parameters set to $C_{low} = 0.2 \cdot C_{max}$ and $C_{up} = 0.9 \cdot C_{max}$ relative to the maximum length of the gradient C_{max}. The result is visualized in red on top of the original grey scale image on the lower right.

where the values for R, G, and B are scaled to $[0,1]$ and represent the different colour channels red, green, and blue. The value

$$C_{max} := \max_{p \in S} \{\|\nabla f\,(p)\|\} \tag{3.41}$$

gives the maximum length of the gradient within the relevant image point set. It serves as a parameter for the threshold determination which is calculated as

$$C_{low} = 0.2 \cdot C_{max} \quad \text{and} \quad C_{up} = 0.9 \cdot C_{max}. \tag{3.42}$$

For a variety of reasons including resolution constraints, a better accuracy than a pixel centre can be required. We want to consider a precision enhancement method for edge detection tasks hereafter.

3.7.4. Sub-pixel Precise Contours

If we look for highly accurate edge detection possibilities to calculate the contour lines with a precision above the resolution of our images, we need a more sophisticated method to correct potential sampling errors due to discretization problems of the image data. To achieve such precision, we use an interpolation with a quadratic polynomial as suggested by Steger [392, pp. 9–10] and Steger [393, pp. 116–117] for 2D line detection tasks.
Every edge pixel $\mathbf{p} = (p_x, p_y)$ of a pre-smoothed image f detected by Algorithm 3.3 lies on a ridge line of extreme values within the **gradient image**

$$r\,(x,y) = \sqrt{f_x\,(x,y)^2 + f_y\,(x,y)^2} \tag{3.43}$$

where the subscripts indicate partial derivatives. This gives an 8-neighbourhood in the gradient image whose values we use to calculate the **two-dimensional quadratic Taylor polynomial**

$$\begin{aligned} r\,(x,y) = r\,(\mathbf{x}) &= r\,(\mathbf{x}_0 + \Delta\mathbf{x}) && (3.44)\\ &\approx r\,(\mathbf{x}_0) + \Delta\mathbf{x}^{\mathrm{T}} \nabla r\,(\mathbf{x}_0) + \frac{1}{2}\Delta\mathbf{x}^{\mathrm{T}} H\,(r\,(\mathbf{x}_0))\,\Delta\mathbf{x} && (3.45)\\ &= r\,(\mathbf{x}_0) + \Delta\mathbf{x}^{\mathrm{T}} \begin{pmatrix} r_x \\ r_y \end{pmatrix} + \frac{1}{2}\Delta\mathbf{x}^{\mathrm{T}} \begin{pmatrix} r_{xx} & r_{xy} \\ r_{xy} & r_{yy} \end{pmatrix} \Delta\mathbf{x} && (3.46) \end{aligned}$$

with $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_0$. And $\nabla r\,(\mathbf{x}_0) = \left(r_x, r_y\right)^{\mathrm{T}}$ is the Jacobian of first-order partial derivatives evaluated at $\mathbf{x}_0$, and

$$H\,(r\,(\mathbf{x}_0)) = \begin{pmatrix} r_{xx} & r_{xy} \\ r_{xy} & r_{yy} \end{pmatrix} \tag{3.47}$$

is the Hessian matrix of second-order partial derivatives at $\mathbf{x}_0$.
Searching for the proper sub-pixel precise edge point location, we look for the direction of $\mathbf{p}$ in which the gradient change is maximal in order to detect the position of its maximum. This direction is determined by the eigenvectors of H. The dominant eigenvector of H (i.e. the

eigenvector corresponding to the eigenvalue with maximal absolute value) points perpendicular to the edge line. Let us denote the normalized direction by $\mathbf{n} = (n_x, n_y)$ with $\|(n_x, n_y)\| = 1$. As a next step, we look for the maximum along this line using the Taylor approximation. We therefore set $\mathbf{x}_0 = \mathbf{0}$ as origin and insert $\mathbf{p}_{\text{subpix}} = \lambda(n_x, n_y)$ into equation (3.46).

$$r(\mathbf{p}_{\text{subpix}}) \tag{3.48}$$

$$\approx r(\mathbf{0}) + (\lambda n_x, \lambda n_y) \begin{pmatrix} r_x \\ r_y \end{pmatrix} + \frac{1}{2} (\lambda n_x, \lambda n_y) \begin{pmatrix} r_{xx} & r_{xy} \\ r_{xy} & r_{yy} \end{pmatrix} \begin{pmatrix} \lambda n_x \\ \lambda n_y \end{pmatrix} \tag{3.49}$$

$$= r(\mathbf{0}) + \lambda n_x r_x + \lambda n_y r_y + \frac{1}{2}\lambda^2 n_x^2 r_{xx} + \lambda^2 n_x n_y r_{xy} + \frac{1}{2}\lambda^2 n_y^2 r_{yy}. \tag{3.50}$$

Differentiation with respect to λ gives

$$\partial_\lambda r(\lambda n_x, \lambda n_y) \approx n_x r_x + n_y r_y + \lambda n_x^2 r_{xx} + 2\lambda n_x n_y r_{xy} + \lambda n_y^2 r_{yy}. \tag{3.51}$$

An evaluation of equation (3.51) at 0 yields

$$\lambda = -\frac{n_x r_x + n_y r_y}{n_x^2 r_{xx} + 2 n_x n_y r_{xy} + n_y^2 r_{yy}} \tag{3.52}$$

and our extreme point is given by

$$\mathbf{p}_{\text{subpix}} = \begin{cases} \lambda \mathbf{n}, & \text{if } (\lambda n_x, \lambda n_y) \in [-0.5, 0.5] \times [-0.5, 0.5] \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{3.53}$$

The **sub-pixel precise contour location** $\mathbf{p}_{\text{subpix}}$ therefore is either the corrected extremum within the pixel width and height or its centre point. In Fig. 3.31 the interpolation method is illustrated for one pre-detected pixel $\mathbf{p}$ and its gradient neighbourhood. The slight offset of the contour point in direction of the arrow is noteworthy.

After the calculation of the new contour points, they still need to be connected to each other to form the discrete contour curve. This follows the same principle as for the centre points in the pixel precise case of section 3.7.3 and is illustrated in comparison to the latter in Fig. 3.32. Finally, we have to note that the gradient image r already contains the two partial derivatives f_x and f_y. The partial derivatives of r are given by

$$r_x = \frac{f_x f_{xx} + f_y f_{xy}}{r} \tag{3.54}$$

$$r_y = \frac{f_x f_{xy} + f_y f_{yy}}{r} \tag{3.55}$$

$$r_{xx} = \frac{f_x f_{xxx} + f_y f_{xxy} + f_{xx}^2 + f_{xy}^2 - r_x^2}{r} \tag{3.56}$$

$$r_{xx} = \frac{f_x f_{xxy} + f_y f_{xyy} + f_{xx} f_{yy} + f_{xy} f_{yy} - r_x r_y}{r} \tag{3.57}$$

$$r_{yy} = \frac{f_x f_{xyy} + f_y f_{yyy} + f_{xy}^2 + f_{yy}^2 - r_y^2}{r}. \tag{3.58}$$

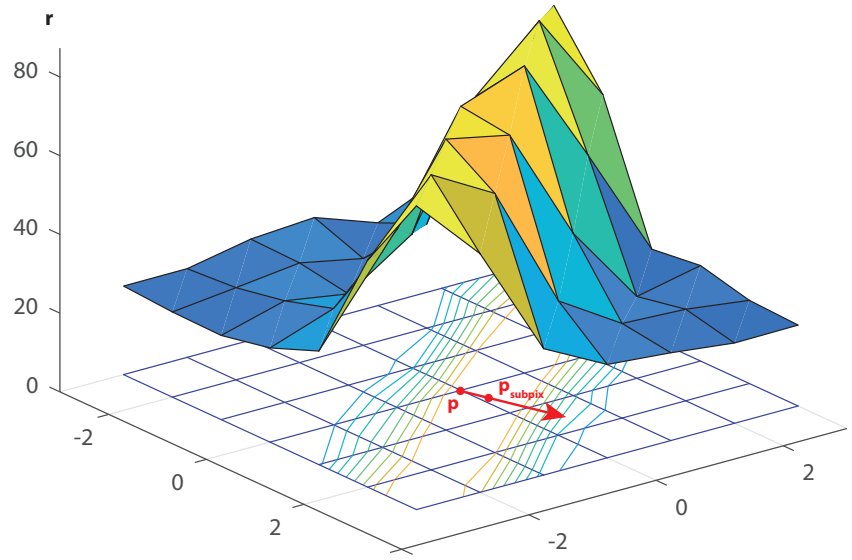


Fig. 3.31. **Sub-pixel precise edge point.** Illustrated are the gradient values $r(x, y)$ above the pixel centre grid. The gradient magnitude is colour coded. The sub-pixel refinement for the pixel $\mathbf{p}$ is given by a step of size λ in the direction of $\mathbf{n}$ which is shown here in red on the grid together with the contour lines. Note the difference between the pixel centre $\mathbf{p}$ and the refined coordinates of $\mathbf{p}_{\text{subpix}}$.

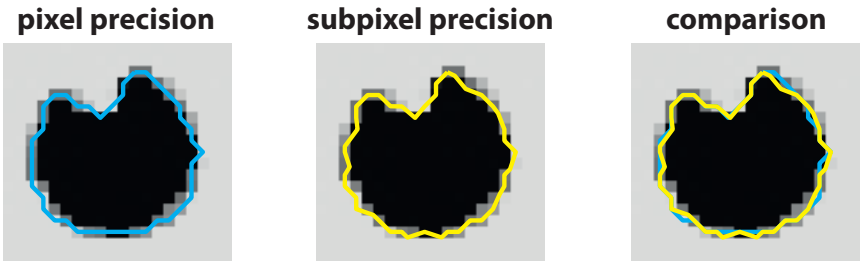


Fig. 3.32. **Detailed sub-pixel contours.** The pixel precise contour detection is drawn in blue on the left. The middle illustrates the sub-pixel refinement. The right image shows the direct comparison of both. Note the difference of the contour lines in particular for points away from the pixel centres.

This causes several computationally costly differentiations including error-prone third-order partial differentiation and can be overcome by applying a convolution method including different 3×3 facet model kernels. Since this is not obligatory for the understanding of the principle for sub-pixel precise contour extraction, we do not go into details of this process and refer to the work of Steger [394, pp. 46–47].

With the methods presented so far it is possible to detect and change special image features such as noisy parts or areas with different light intensities. Furthermore, we can determine image sections according to their grey values or significant shapes. The contour operations for the shape detection can even be performed with an accuracy above the pixel grid resolution. The **drawback** is the **computational complexity**. Those tasks require considerable processing power for large regions and it is our objective to diminish the considered regions as much as possible if we apply sub-pixel precise detection. In the next section we apply the previous work on contours to search the image for particular shapes with given parameters.

3.8. Ellipse Fitting

Are there circles in an image? How can we group ellipses of similar size from pixel data? Which contour looks like the one in my database? We will now discuss possible answers and an efficient algorithm for questions as the ones above.
With the methods presented in section 3.7 it is now not only possible to detect the pixels of image edges, but also to determine the contour line even with sub-pixel precision. The huge amount of information for such contours including the contour points and their connected neighbours are often not of interest for tasks where only the length of a contour line or the diameter of a circular hole are demanded. We need to fit parameters to our measurements to tackle tasks where simple geometric objects, such as lines, circles or ellipses serve as representatives for noisy contours with a few parameters that are often difficult to compare with similar discrete curves. Such objects are called **geometric primitives**.
Simple geometric primitive such as lines can be robustly detected with iterative optimization approaches like the method proposed by Lanser [235, pp. 72–75]. These 2D-forms are relevant for many machine vision tasks which involve camera based measurements, quality inspection and can serve as anchors for classical vision pipelines. For our purposes, however, we focus solely on the detection of ellipses since these are the projections of circles to be detected. Detection of other 3D primitives is briefly discussed in section 8.2.1.

In order to formulate an ellipse fit for a certain pixel set, we start with the representation of a general projectively transformed circle as a quadratic form in $\mathbb{RP}^2$ by its implicit formula with homogeneous coordinates

$$\mathbf{p}^{\mathrm{T}}\mathbf{C}\,\mathbf{p} = (x,y,z)\begin{pmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{3.59}$$

$$= ax^2 + bxy + cy^2 + dxz + eyz + fz^2 = 0. \tag{3.60}$$

Following the idea of Richter-Gebert [347, pp. 148–149], we intersect the conic represented by the homogeneous equation (3.60) with the line at infinity $l_\infty : z = 0$ to classify object properties. This gives

$$ax^2 + bxy + cy^2 = 0. \tag{3.61}$$

Solving equation (3.61) for $x = x(y)$ yields

$$x = \left(\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}\right) y, \tag{3.62}$$

which has up to scalar multiples either none, one or two solutions depending on the discriminant. The three different cases for the transformed circle are illustrated in Fig. 3.33.[20]

[20] Figure based on Richter-Gebert [347, p. 149].

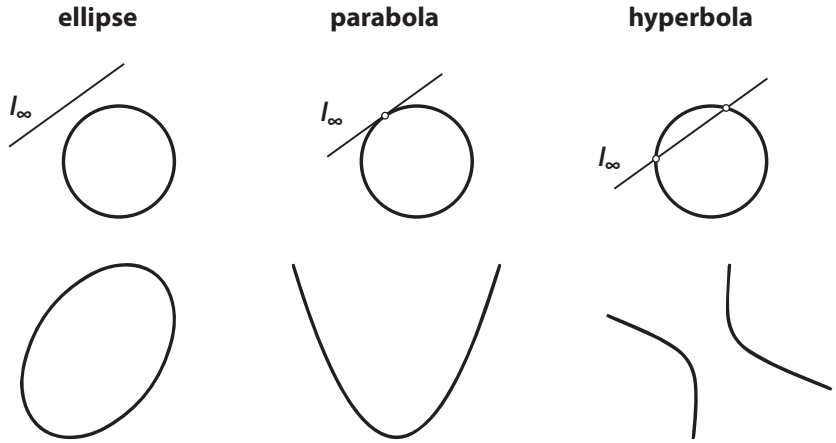


Fig. 3.33. **Different forms of a conic.** The shape of the resulting geometric object depends on the intersection of the conic with the line at infinity l_∞. In the top row, the three possible classes are illustrated schematically. If the line has no common point with the conic (see top left), the result is an ellipse (lower left). Exactly one common point gives a parabola as shown in the middle while two intersections form a hyperbola (lower right).

Since we are only interested in the case of ellipses, we consider the constraint case for the conic being an ellipse given by

$$b^2 - 4ac < 0. \tag{3.63}$$

Dehomogenization of equation (3.60) with $z = 1$ finally gives

$$\mathbf{p}^\mathrm{T}\mathbf{C}\,\mathbf{p} = ax^2 + bxy + cy^2 + dx + ey + f = 0, \tag{3.64}$$

which we can rewrite as

$$\begin{aligned} 0 &= \mathbf{p}^\mathrm{T}\mathbf{C}\,\mathbf{p} && (3.65)\\ &= \underbrace{\begin{pmatrix} a & b & c & d & e & f \end{pmatrix}}_{\mathbf{r}^\mathrm{T}} \underbrace{\begin{pmatrix} x^2 & xy & y^2 & x & y & 1 \end{pmatrix}^\mathrm{T}}_{\mathbf{d}} && (3.66)\\ &= \mathbf{r}^\mathrm{T}\mathbf{d} && (3.67)\\ &=: C(\mathbf{r},\mathbf{d}), && (3.68) \end{aligned}$$

where $\mathbf{r}$ represents the vector of conic parameters that determine the conic type and $\mathbf{d}$ is the designed variable vector that describes the structure of the object. $C(\mathbf{r},\mathbf{d})$ gives the algebraic distance of a point $p = (x, y, 1)$ to the conic $C(\mathbf{r},\mathbf{d}) = 0$. This is the basis for our fitting.
A non-linear approach for minimizing the geometric error for a fit of a 2D point cloud to an ellipse is for example given by Ahn et al. [3]. Even though there exist such ideas, they often use elaborate approximation techniques and thus are computationally complex. Since we want to develop real-time systems, we follow a **linear approach** and **minimize the algebraic error** given by $C(\mathbf{r},\mathbf{d})$ instead, which still gives satisfactory results. However, we keep in mind that for highly accurate fitting procedures without hard runtime requirements, this would not be the method of choice. In this case, choosing a geometric minimization approach would be preferable.

Firstly, we note that, similar to the line fitting, the set of parameters is a homogeneous quantity, since

$$C(\mathbf{r}, \mathbf{d}) = 0 \quad \Leftrightarrow \quad C(\tau\mathbf{r}, \mathbf{d}) = 0 \tag{3.69}$$

$\forall \tau \in \mathbb{R} \setminus \{0\}$. This leaves us the choice of arbitrarily scaling the parameter vector $\mathbf{r}$. We follow the idea of Fitzgibbon et al. [117] and incorporate the choice of τ into the inequality ellipse constraint (3.63) to form the equality constraint

$$4ac - b^2 = 1 \tag{3.70}$$

instead. We can rewrite this in terms of matrix-vector formalism as

$$1 = 4ac - b^2 \tag{3.71}$$

$$= \begin{pmatrix} a & b & c & d & e & f \end{pmatrix} \underbrace{\begin{pmatrix} 0 & 0 & 2 & \\ 0 & -1 & 0 & \mathbf{0}_{[3,3]} \\ 2 & 0 & 0 & \\ & \mathbf{0}_{[3,3]} & & \mathbf{0}_{[3,3]} \end{pmatrix}}_{\mathbf{A}} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} \tag{3.72}$$

$$= \mathbf{r}^{\mathrm{T}} \mathbf{A}\, \mathbf{r}, \tag{3.73}$$

with $\mathbf{0}_{[3,3]}$ representing a 3×3 matrix filled with zeros.
Let us now write all measurements in one matrix. If n measured points are given by $p_i = (x_i, y_i, 1)$, we write them in form of vector $\mathbf{d}$ from equation (3.66) and build

$$\mathbf{D} = \begin{pmatrix} \mathbf{d}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{d}_n^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 & 1 \\ & & \vdots & & & \\ x_n^2 & x_n y_n & y_n^2 & x_n & y_n & 1 \end{pmatrix}. \tag{3.74}$$

Then the minimization problem summing up all the residuals

$$\min \sum_{i=1}^{n} C(\mathbf{r}, \mathbf{d}_i) \tag{3.75}$$

$$\text{subject to } b^2 - 4ac < 0 \tag{3.76}$$

reduces to

$$\min \|\mathbf{D}\, \mathbf{r}\|^2 \tag{3.77}$$

$$\text{subject to } \mathbf{r}^{\mathrm{T}} \mathbf{A}\, \mathbf{r} = 1, \tag{3.78}$$

where $\|\cdot\| = \|\cdot\|_2$ is the Euclidean norm and $\mathbf{D}$ is called the **design matrix**.[21]
We solve this with a Lagrange multiplier and focus now on

$$\min \underbrace{\|\mathbf{D}\,\mathbf{r}\|^2 - \lambda\left(\mathbf{r}^{\mathrm{T}}\mathbf{A}\,\mathbf{r} - 1\right)}_{S} \tag{3.79}$$

instead. Differentiation leads to the necessary condition

$$\begin{aligned}
\mathbf{0} &\overset{!}{=} \nabla S && (3.80)\\
&= \left(\partial_a, \partial_b, \partial_c, \partial_d, \partial_e, \partial_f\right)^{\mathrm{T}} S && (3.81)\\
&= \nabla\left(\|\mathbf{D}\,\mathbf{r}\|^2 - \lambda\left(\mathbf{r}^{\mathrm{T}}\mathbf{A}\,\mathbf{r} - 1\right)\right) && (3.82)\\
&= \nabla\left(\mathbf{r}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathbf{D}\,\mathbf{r}\right) - \nabla\lambda\;\mathbf{r}^{\mathrm{T}}\mathbf{A}\,\mathbf{r} && (3.83)\\
&= 2\;\mathbf{D}^{\mathrm{T}}\mathbf{D}\,\mathbf{r} - 2\lambda\;\mathbf{A}\,\mathbf{r}, && (3.84)
\end{aligned}$$

where the last step works because $\mathbf{A}$ and the scatter matrix $\mathbf{M} := \mathbf{D}^{\mathrm{T}}\mathbf{D}$ are symmetric. This equation can be written as the generalized eigenvalue problem

$$\mathbf{M}\,\mathbf{r} = \lambda\;\mathbf{A}\,\mathbf{r}, \tag{3.85}$$

for which there exist six eigenvalue-eigenvector pairs $(\lambda_k, \mathbf{r}_k)$. Since

$$\|\mathbf{D}\,\mathbf{r}\|^2 = \mathbf{r}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathbf{D}\,\mathbf{r} = \mathbf{r}^{\mathrm{T}}\mathbf{M}\,\mathbf{r} = \lambda\;\mathbf{r}^{\mathrm{T}}\mathbf{A}\,\mathbf{r} = \lambda, \tag{3.86}$$

we are interested in the eigenvector $\mathbf{r}_+$ corresponding to the minimal eigenvalue $\lambda_+ \in \mathbb{R}_0^+$ for which the minimization achieves the best value.[22] Moreover, such an eigenvalue is unique and it always exists.[23]
Let us suppose we already have the pair $(\lambda_+, \mathbf{r}_+)$ that solves equation (3.85). The parameter vector $\mathbf{r}_+$ is still only fixed up to scale, since $(\lambda_+, \eta\;\mathbf{r}_+)$ also satisfies (3.85) $\forall \eta \in \mathbb{R} \setminus \{0\}$. We can find the proper scaling factor by considering condition (3.78), where

$$\eta_+^2\,\mathbf{r}_+^{\mathrm{T}}\mathbf{A}\,\mathbf{r}_+ = 1 \tag{3.87}$$

finally gives

$$\eta_+ = \sqrt{\frac{1}{\mathbf{r}_+^{\mathrm{T}}\mathbf{A}\,\mathbf{r}_+}}. \tag{3.88}$$

[21] Cf. O'Leary and Zsombor-Murray [310, p. 494].
[22] Cf. Halíř and Flusser [159].
[23] It can be shown that the signs of the eigenvalues of the generalized eigenvalue problem $\mathbf{M}\,\mathbf{r} = \lambda\;\mathbf{A}\,\mathbf{r}$ with positive definite $\mathbf{M}$ and symmetric $\mathbf{A}$ are the same as the signs of the eigenvalues of $\mathbf{A}$ up to permutation (see Lemma 1, Fitzgibbon et al. [117, p. 478]). Since the different eigenvalues of $\mathbf{A}$ in our case are $\{-2, -1, 0, 2\}$, this leaves exactly one positive eigenvalue.

We conclude this line of thoughts by putting the ideas together in Algorithm 3.4.

Algorithm 3.4. Ellipse Fitting

Input parameters:

- Contour points $p_i = (x_i, y_i)$ with $i \in \{1, \ldots, n\}$

Computation steps:

1. Set up design matrix $\mathbf{D} = \begin{pmatrix} \mathbf{d}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{d}_n^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 & 1 \\ & & \vdots & & & \\ x_n^2 & x_n y_n & y_n^2 & x_n & y_n & 1 \end{pmatrix}$
2. Calculate scatter matrix $\mathbf{M} := \mathbf{D}^{\mathrm{T}}\mathbf{D}$
3. Solve generalized eigenvalue problem $\mathbf{M}\,\mathbf{r} = \lambda\,\mathbf{A}\,\mathbf{r}$
4. Get $(\lambda_+, \mathbf{r}_+)$ with $\lambda_+ \in \mathbb{R}_0^+$
5. Calculate scaling factor $\eta_+ = \sqrt{\dfrac{1}{\mathbf{r}_+^{\mathrm{T}}\mathbf{A}\,\mathbf{r}_+}}$
6. Scale parameter vector $\mathbf{r} = \eta_+ \mathbf{r}_+$

Output:

- Ellipse parameters $\mathbf{r}^{\mathrm{T}} = \begin{pmatrix} a & b & c & d & e & f \end{pmatrix}$

This procedure is robust against occlusion and small outliers, as you can see in Fig. 3.34. If large deviations from the ellipse contour occur, weighting coefficients for the algebraic distances should be introduced.[24] The two figures Fig. 3.34 and Fig. 3.35 shows that this method can work in presence of noise and on real data. Thus, we do not consider any further weighting here.

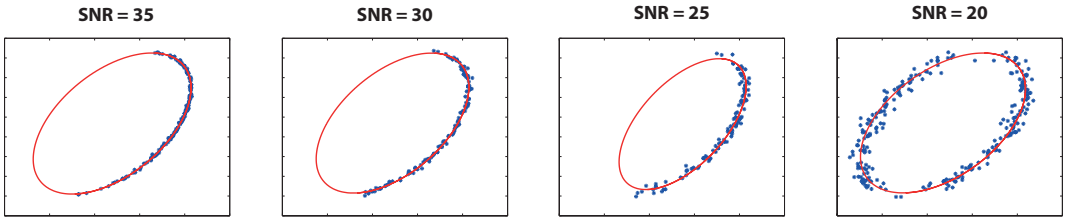


Fig. 3.34. **Fitted ellipses of artificial point sets with decreasing signal to noise (SNR) ratio.** The illustrations show a synthetically created set of contour points on an ellipse with different noise levels in blue. The results of Algorithm 3.4 on these point sets is depicted in red. Note that the algorithm is robust to both a severe occlusion of contour points (see the left three illustrations) and various levels of noise.

3.8.1. Ellipse Properties

At last, we want to state some useful ellipse properties, especially how we derive **geometric quantities** that describe an elliptic shape given its algebraic parameters. We thereby focus on

[24] One such concept is given by Lanser [235, pp. 76–77].

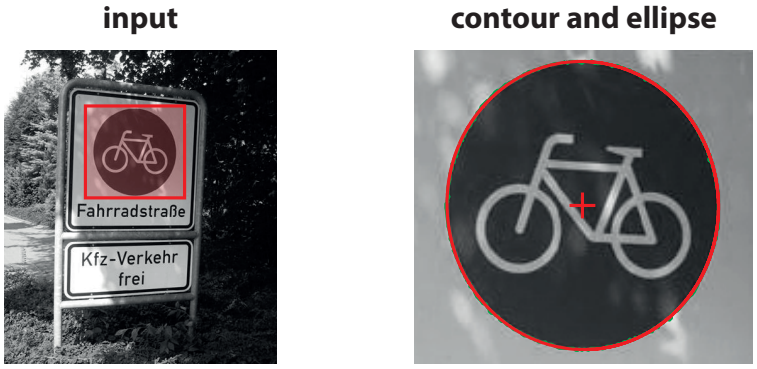


Fig. 3.35. Ellipse fitted on contour. For the image shown on the left, a contour detection delivers the contour drawn in green on the right. Algorithm 3.4 detects the ellipse with the centre point shown in red on the right as an overlay.

the centre point $q = (q_x, q_y)$ and the semi axes as illustrated in Fig. 3.36. In order to do this, we remember the dehomogenized algebraic equation (3.64) for an ellipse given by

$$ax^2 + bxy + cy^2 + dx + ey + f = 0. \tag{3.89}$$

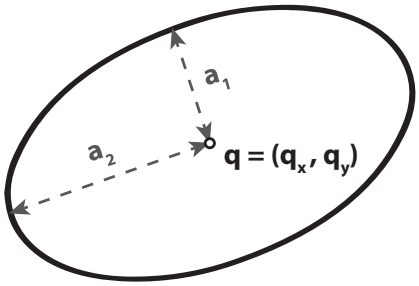


Fig. 3.36. Ellipse with centre point and semi-axes. The semi-axes lengths are shown with dashed arrows. The major axis joins the centre point $q = (q_x, q_y)$ and the ellipse points of maximal distance a_2 away from q while the minor axis joins the centre and its closest points on the ellipse contour at distance a_1.

Thus we get

$$q_x = \frac{be - 2cd}{4ac - b^2} \quad \text{and} \quad q_y = \frac{bd - 2ae}{4ac - b^2}, \tag{3.90}$$

as centre point coordinates in the non-degenerate case, and

$$a_{1,2} = \frac{2ae + 2cd + 8f\,b^2 - 2bde - 8ace}{(4ac - b^2)\left((a + c) \pm \sqrt{(a - c)^2 + b^2}\right)} \tag{3.91}$$

for the two semi-axes lengths.[25]

As a last step in this image processing study, we combine the designed ideas and processes to form a robust algorithm for detecting centres of circles within arbitrary images.

[25] Cf. Weisstein [447].

3.9. Marker Detection Chain

Imagine an object with uni-coloured circular stickers on its surface that vary in their colour significantly from the rest of the object - or a manufactured planar board with circles. We now want to collect all the ideas from the previous sections to form an algorithm that is able to detect the position of the centre points of these circles within an arbitrary image of such a scene. We therefore note that depending on the camera position, this task is identical to finding a circular shaped structure under some projection. In other words, we look for the **centre points of ellipse contours** within the entire image.
Let us assume that all the hardware is set up properly and the camera is already calibrated as proposed in section 4.2 where the intrinsic camera parameters are satisfyingly estimated.[26] The circular regions reflect or absorb our illumination in a way that foreground and background can be clearly distinguished. If a new image is acquired, Algorithm 3.5 gives the coordinates of the centre points of the ellipse structures within this image, where pixel size units are used. Fig. 3.37 shows a step-by-step example of the subroutines of Algorithm 3.5.

This algorithm is the essence of this chapter and we use it as a basis for further tasks and problem solutions. From now on, we can look at it pragmatically as in most cases, it is sufficient to know what it does whereas the technical details are not crucial and we therefore mostly refer to it as a black box which translates an input image into a coordinate set of centre points.
Before we look at further image processing steps and analyze various geometries, we focus on the reliability of our image data by looking at the geometry of a single camera setup and discuss ways to correct already mentioned hardware errors with mathematical models based on the extraction of image content.

[26] The methods proposed in section 4.2 – and the intrinsic camera parameters in particular – guarantee the correction of hardware errors for real applications. For this reason, we can suppose an idealized theoretical setup here and deal with the necessary requirements to approximate such a setup in the mentioned section.

Algorithm 3.5. Extraction of Ellipse Centre Coordinates

Input parameters:

- Image f with grey values $g_{x,y}$, width w, height h, and pixel depth d

Preprocessing:

1. **Image Enhancement** ($f \rightarrow f_v, f_{enh}$)
   a) Set truncation parameters p_{low} and p_{up}
   b) Run robust contrast normalization (Algorithm 3.2)
   c) Save image histogram f_v
2. **Filtering** ($f_{enh} \rightarrow f_{fil}$)
   a) Prepare 5×5 Gaussian kernel k (Equation (3.22))
   b) Convolve $f_{fil} := (k \star g_{enh})_{x,y}$ (Definition 3.3)

Computation steps:

1. **Segmentation** ($f_{fil}, f_v \rightarrow R$)
   a) 1D Gaussian filtering of f_v ($f_v \rightarrow f_{v_{fil}}$)
   b) Get 2 largest local maxima $f_{v_{low}}$, $f_{v_{high}}$ of $f_{v_{fil}}$ with $v_{low} < v_{high}$
   c) $g_{thresh} := \dfrac{v_{high} - v_{low}}{2}$
   d) Perform basic thresholding on f_{fil} to retrieve R (Definition 3.6)
2. **Morphology** ($R \rightarrow R_{mor}$)
   a) Prepare disc-shaped structuring element E
   b) Do opening $R_{mor} := R \circ E$ (Definition 3.9)
3. **Separation** ($R_{mor} \rightarrow S = \{S_1, \ldots, S_n\}$)
   a) Set $n = 0$, $S_0 = \emptyset$
   b) Get connected components $\forall p \in R_{mor}$:
      if $\exists\ i \in \{0, \ldots, n\}:$ p 8-connected to S_i (Definition 3.10)
      $S_i = S_i \cup p$
      else
      $n = n + 1$ and $S_n := \{p\}$
4. **Contour Detection** ($S \rightarrow C = \{C_1, \ldots, C_n\}$)
   a) Prepare disc-shaped structuring element L
   b) Get contour of RoI $\forall\ i \in \{0, \ldots, n\}$:
      - Extend RoI $C_i := S_i \oplus L$ with dilation (Definition 3.8)
      - Calculate image gradient $\mathrm{D}[\nabla f\,(p)]$ $\forall p \in C_i$ (Equation (3.39))
      - Do pixel-based non-maximum suppression $\forall p \in C_i$ (Fig. 3.27)
      - Calculate C_{max} (Equation (3.41)). $C_{low} = c^- \cdot C_{max}$, $C_{up} = c^+ \cdot C_{max}$
      - Perform hysteresis thresholding (Algorithm 3.3)
      - Get sub-pixel precise contour points p_{subpix} $\forall p \in C_i$ (Equation (3.53))
      - Save coordinates $p = p_{\text{subpix}}$ $\forall p \in C_i$
5. **Ellipse Fitting** ($C \rightarrow Q = \{q_1, \ldots, q_m\}$)
   a) Set $Q := \emptyset$ and analyze contour $\forall\ i \in \{0, \ldots, n\}$
      - Calculate ellipse parameters $\mathbf{r}_i$ from C_i (Algorithm 3.4)
      - Filter ellipse centres in coordinate set Q
        if ellipse shape tolerable with $\frac{\max\{a_1,a_2\}}{\min\{a_1,a_2\}} < a_{thresh}$ (Equation (3.91))
        Detect centre points $q = \left(q_x, q_y\right)$ (Equation (3.90))
        $Q = Q \cup q$

Output:

- Coordinate set with centres $Q = \{q_1, \ldots, q_m\}$

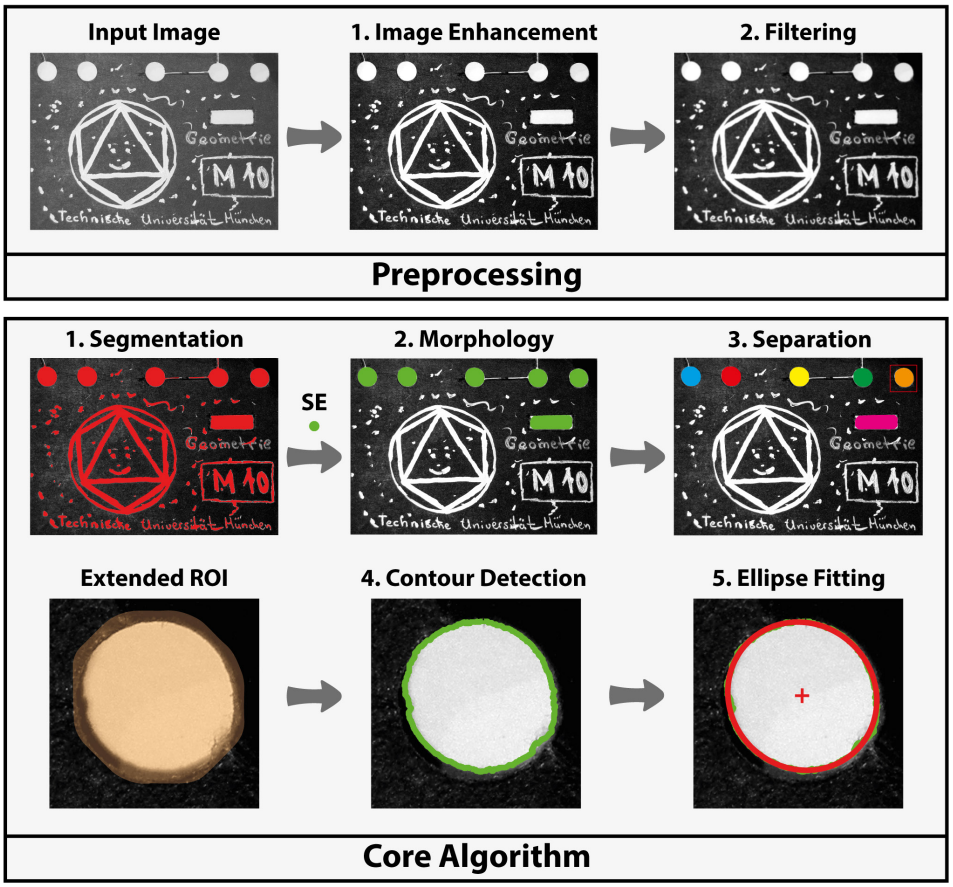


Fig. 3.37. **Example with subroutines of Algorithm 3.5.** In order to extract ellipse centre points a three stage algorithm is performed. In a first preprocessing stage, the input image (top left) is first processed with a robust contrast normalization (top middle) and then smoothed with a Gaussian kernel (top right). The second stage detection pipeline commences with a basic thresholding (middle left) to obtain a first segmentation mask. The mask is reduced with a morphological opening (centre image) and connected components are separated (middle right). In a third stage, each individual region of interest is then extended (bottom left) and fed into a contour detector (bottom middle). An ellipse fitting method delivers ellipse parameters to filter relevant centre points (bottom right).

4 Camera Geometry

> ΑΓΕΩΜΕΤΡΗΤΟΣ
> ΜΗΔΕΙΣ ΕΙΣΙΤΩ.
>
> – **Plato's Academy, Athens**
> (Engraved at the Door)[1]

Once the data is acquired, transferred and present as a pixel-array in the memory of our processing unit (see Fig. 2.1), we can describe it with the presented mathematical formulation of 2D image and video.
In this chapter, we detail the parametric camera model we want to use and investigate how to calibrate the camera with the concepts and algorithms from chapter 3. We start to investigate the camera component of the vision pipeline more closely in order to bridge the gap between the physical exposure of cells on the sensor and the availability of an image as a pixel grid with intensity values. To do so, we provide the required mathematical background to describe the camera and its surroundings with the help of projective geometry and formulate a model that describes the projection of world points onto images. We then focus on the estimation of the model parameters.
In order to fit the model parameters to our hardware, we use a camera calibration algorithm including the pinhole camera model as used by Zhang [481] and combine it with lens distortion correction from Heikkila and Silven [168].

4.1. Camera Model

A camera maps a world point $\mathbf{x}_W = (x, y, z)$ with its three coordinates onto a two-dimensional image. Thus, geometrically speaking, an idealized standard camera is a mapping.
We use projective geometry to investigate the properties of the standard **pinhole camera**. The model is based on the optical phenomenon which can be observed when light travels through a small hole and gets projected onto a flat surface. In a dark room one can observe a reversed an inverted image of the scene on the other side, which is why this apparatus is also called a "camera obscura"[2] and was used together with a lens in the opening already in the 16th century by drawers and painters.[3] The first published picture of a camera obscura dates back to Frisius [124, p. 61] from 1545 where the author describes an installation to observe the solar

[1] "No one ignorant of geometry may enter.", as detailed in D. H. Fowler. The Mathematics of Plato's Academy: A New Reconstruction [pp. 200-201]. Oxford University Press, 1987.
[2] "Camera obscura" is Latin for "dark room".
[3] Cf. Gage and Gage [130].

eclipse from January 24, 1544. It is illustrated in Fig. 4.1 together with a schematic drawing. Most of the modern cameras on the market can be describe with this model.[4]

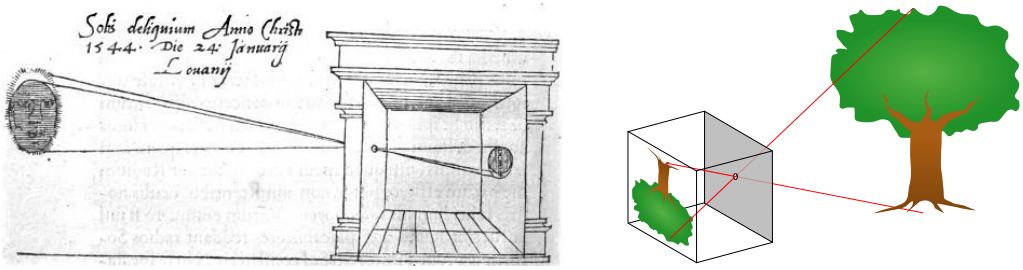


Fig. 4.1. **Camera obscura.** On the left, the first published picture of a camera obscura from Frisius [124, p. 61] is shown. Note how the occluded solar region is upside down on its projection plane. On the right, a schematic drawing shows the principle of such installations.

4.1.1. Pinhole Camera

The geometry of the model we want to use is illustrated in Fig. 4.2[5]. The world point $\mathbf{x}_C$ is projected through the pinhole to the point $\mathbf{x}$ on the **image plane** (or principal plane) which is located at a distance f from the projection centre. We call f the principle distance or **focal length.**[6] The **principle point** $C = (c_x, c_y)$ is the base point for the perpendicular principle ray through the **camera centre** O. The coordinate system (x_C, y_C, z_C) located at O is the **camera coordinate system** and we call a system (x_{img}, y_{img}) the **image plane coordinate system** and the shifted, scaled version (x, y) the **image coordinate system** where the pixel width s_x and height s_y in image plane coordinates determine the scaling of the axes.
We keep in mind that the origin of the latter is on the top left corner of the image and consistent with our Definition 2.2, since the projection flips the actual orientation of the scene. Therefore it can sometimes be easier to work with a **virtual image plane** in front of the camera centre at $z_c = f$ with aligned axes to the camera coordinate system as illustrated in Fig. 4.3. We come back to the method of virtual image planes in chapter 6.3.

4.1.2. World to Image

At first, we look a bit closer to the given geometry and formulate the mapping from an arbitrary world point $\mathbf{x}_C$ to the point $\mathbf{x}$ in pixel coordinates. As shown in Fig. 4.3, $\mathbf{x}_C = (x, y, z)$ maps to $\left(\frac{fx}{z}, \frac{fy}{z}, -f\right)$ and the projection is given in image plane coordinates by

$$\begin{pmatrix} x & y & z \end{pmatrix}^{\mathrm{T}} \mapsto \begin{pmatrix} \frac{fx}{z} & \frac{fy}{z} \end{pmatrix}^{\mathrm{T}}. \tag{4.1}$$

[4] Cf. Xu and Zhang [456, p. 8].
[5] Figure inspired by Steger, Ulrich, and Wiedemann [395, p. 182].
[6] Cf. Tipler and Mosca [411, p. 1038].

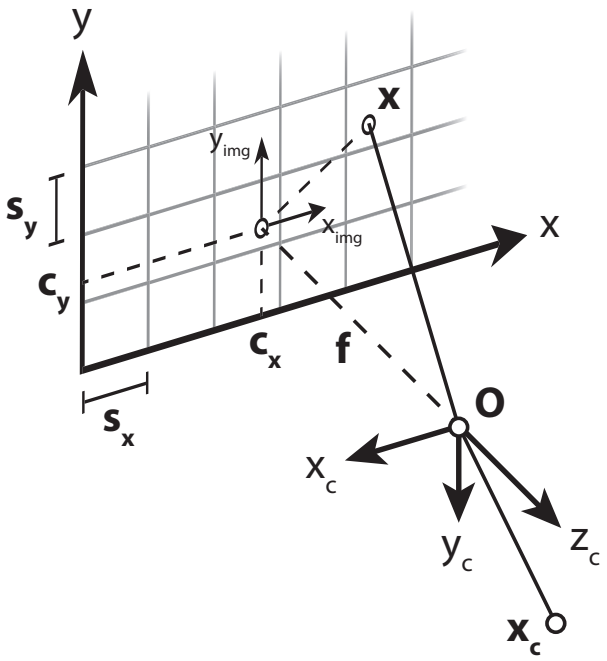


Fig. 4.2. **Pinhole camera geometry.** The world point $\mathbf{x}_C$ is projected to the image plane. The focal length f between the principle point $C = (c_x, c_y)$ and the optical centre O together with the pixel width s_x and height s_y determine the image coordinate (x, y) of $\mathbf{x}$.

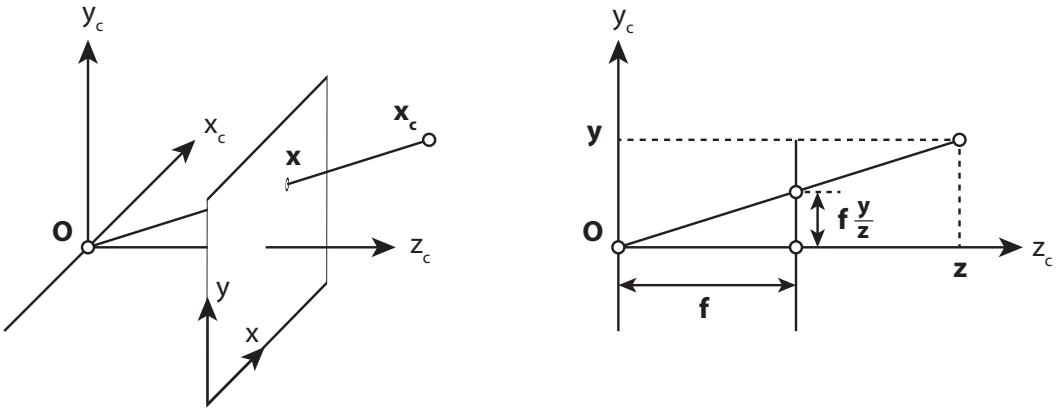


Fig. 4.3. **Virtual image plane.** The actual projection of the scene lies behind the optical centre at $z = -f$ to the left of these axes and the orientation is flipped. For convenience, a virtual image plane in front of the camera coordinate system at $z = f$ is introduced to simplify the orientation and align the axes. The left image shows the virtual image plane in a generic projection while the right image illustrates the y_C-z_C-plane in side view.

Describing the projection in terms of homogeneous coordinates leads to

$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fx \\ fy \\ z \end{pmatrix} = \underbrace{\begin{pmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{pmatrix}}_{\mathbf{M}} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{4.2}$$

where the projection matrix $\mathbf{M}$ can be expressed with the notation of Šonka et al. [389, pp. 564–565] as

$$\mathbf{M} = \begin{pmatrix} f & & \\ & f & \\ & & 1 \end{pmatrix} \begin{pmatrix} \mathbf{I} \mid \mathbf{0} \end{pmatrix} \tag{4.3}$$

$$= \mathbf{Q}_1 \begin{pmatrix} \mathbf{I} \mid \mathbf{0} \end{pmatrix} \tag{4.4}$$

with the identity matrix $\mathbf{I} \in \mathbb{R}^{3\times3}$ and a column vector $\mathbf{0} \in \mathbb{R}^3$ appended to the right.
To transform the origin of the coordinate system to the origin of the image coordinate system, a further translation is needed, so that

$$\begin{pmatrix} x & y & z \end{pmatrix}^\mathrm{T} \mapsto \begin{pmatrix} \frac{fx}{z} + \tilde{c}_x & \frac{fy}{z} + \tilde{c}_y \end{pmatrix}^\mathrm{T} \tag{4.5}$$

with $\tilde{c}_x = c_x s_x$ and $\tilde{c}_y = c_y s_y$ in image plane coordinates. This gives

$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fx + z\tilde{c}_x \\ fy + z\tilde{c}_y \\ z \end{pmatrix} = \begin{pmatrix} 1 & & \tilde{c}_x \\ & 1 & \tilde{c}_y \\ & & 1 \end{pmatrix} \mathbf{Q}_1 \begin{pmatrix} \mathbf{I} \mid \mathbf{0} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{4.6}$$

$$= \mathbf{Q}_2 \mathbf{Q}_1 \begin{pmatrix} \mathbf{I} \mid \mathbf{0} \end{pmatrix} \mathbf{x}_C \tag{4.7}$$

$$= \begin{pmatrix} f & & \tilde{c}_x & 0 \\ & f & \tilde{c}_y & 0 \\ & & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}. \tag{4.8}$$

Since CCD cameras can have a non-squared element alignment,[7] a different scaling according to the two axes of the image coordinate system is introduced. Therefore we finally scale the

[7] Cf. Hartley and Zisserman [165, p. 156].

x and y-coordinate properly according to the pixel width s_x and the pixel height s_y, which yields

$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} s_x^{-1}(fx + z\tilde{c}_x) \\ s_y^{-1}(fy + z\tilde{c}_y) \\ z \end{pmatrix} = \begin{pmatrix} s_x^{-1} & & \\ & s_y^{-1} & \\ & & 1 \end{pmatrix} \mathbf{Q}_2\mathbf{Q}_1 \Big(\mathbf{I} \mid \mathbf{0} \Big) \tag{4.9}$$

$$= \mathbf{Q}_3\mathbf{Q}_2\mathbf{Q}_1 \Big(\mathbf{I} \mid \mathbf{0} \Big) \mathbf{x}_C \tag{4.10}$$

$$= \begin{pmatrix} fs_x^{-1} & & c_x & 0 \\ & fs_y^{-1} & c_y & 0 \\ & & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{4.11}$$

with the camera calibration matrix

$$\mathbf{K} = \mathbf{Q}_3\mathbf{Q}_2\mathbf{Q}_1 \tag{4.12}$$

containing the **intrinsic camera parameters**. To sum up, this gives the transformation

$$\mathbf{x} = \mathbf{K} \Big(\mathbf{I} \mid \mathbf{0} \Big) \mathbf{x}_C \tag{4.13}$$

from camera coordinates $\mathbf{x}_C$ to image coordinates $\mathbf{x}$.
Note that because of the structure of $\mathbf{K}$ not all parameters can be determined by a set of coordinates and their projections since the system is underdetermined. A change of f for example could be counterbalanced by a change of the pixel sizes s_x and s_y as illustrated in Fig. 4.4[8] where two possible camera realizations for the same scenario are shown.
We solve this issue by fixing the pixel size namely s_y with the knowledge of the sensor specifications by the manufacturer.[9]

In general, an arbitrary point can be given in some world coordinate system as $\mathbf{x}_W = (x_W, y_W, z_W)$ which is not necessarily aligned to the camera coordinates $\mathbf{x}_C = (x_C, y_C, z_C)$. In fact, the first is related with the latter through a translation $\mathbf{t}$ and a rotation $\mathbf{R}$ as shown in Fig. 4.5[10].

[8] Figure inspired by Steger, Ulrich, and Wiedemann [395, p. 191].
[9] Cf. Steger, Ulrich, and Wiedemann [395, p. 191].
[10] Figure inspired by Steger, Ulrich, and Wiedemann [395, p. 182].

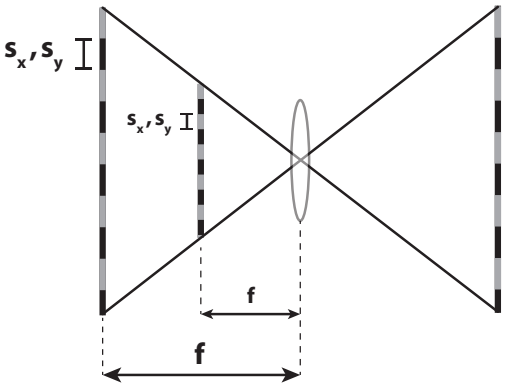


Fig. 4.4. **Underdetermination of camera parameters.** The system is underdetermined for the parameters f, s_x, and s_y. Two possible realizations (left) for the projection of the right are shown.

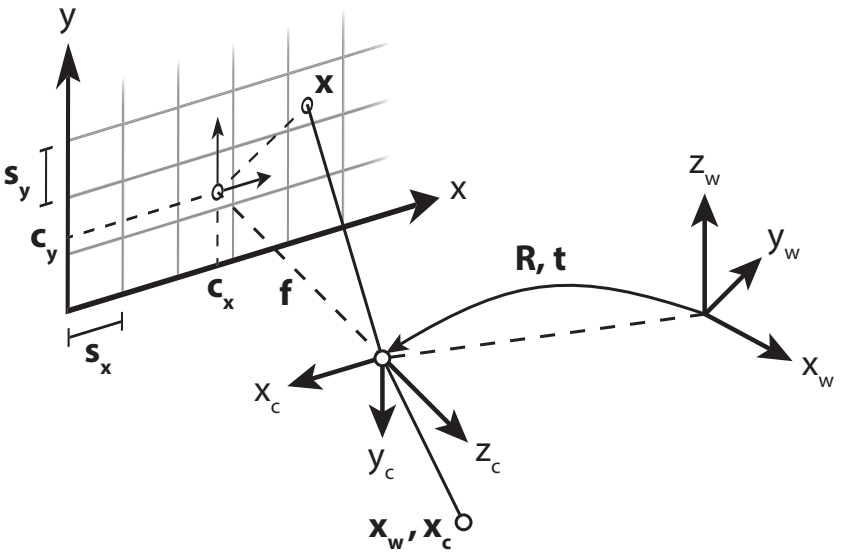


Fig. 4.5. **Pinhole camera with different coordinate systems.** The two coordinate systems are connected with a rigid transformation involving the rotation $\mathbf{R}$ and the translation $\mathbf{t}$. This transformation aligns the world coordinate system $\mathbf{x}_W = (x_W, y_W, z_W)$ to the camera coordinates $\mathbf{x}_C = (x_C, y_C, z_C)$.

If the origin of the camera coordinate system is given by C_W in world coordinates, we can write $\mathbf{x}_C = \mathbf{R}(\mathbf{x}_W - C_W)$. A coordinate transformation from world coordinates to camera coordinates can then be written as[11]

$$\mathbf{W} = \begin{pmatrix} \boxed{\mathbf{R}} & \begin{matrix} | \\ \mathbf{t} \\ | \end{matrix} \\ - \; \mathbf{0} \; - & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & -\mathbf{R}C_W \\ \mathbf{0} & 1 \end{pmatrix}. \tag{4.14}$$

[11] Cf. Richter-Gebert and Orendt [348, pp. 19–20] where this is proposed for 2D translations and rotations.

with the **extrinsic camera parameters**, namely the translation vector $\mathbf{t} = -\mathbf{R}C_W \in \mathbb{R}^3$ and the rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$.
Together with equation (4.13) for the intrinsic camera parameters we can now formulate the complete transformation of a world point to an image point following the chain shown in Fig. 4.6 as

$$\mathbf{x} = \mathbf{K}\mathbf{R}\left(\mathbf{I} \mid -C_W\right)\mathbf{x}_W \tag{4.15}$$

$$= \mathbf{K}\left(\mathbf{R} \mid \mathbf{t}\right)\mathbf{x}_W. \tag{4.16}$$

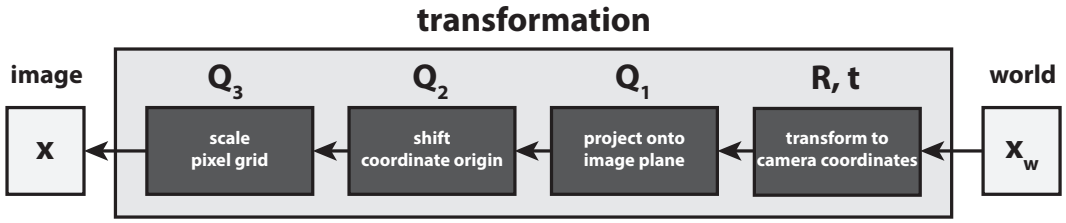


Fig. 4.6. **Projection chain of world point x_W.** The world point is first referenced to the camera coordinates with a rigid transformation by $\mathbf{R}$ and $\mathbf{t}$ before it is projected onto the image plane ($\mathbf{Q}_1$). The transformation $\mathbf{Q}_2$ shifts the coordinate origin and $\mathbf{Q}_3$ takes care for scaling.

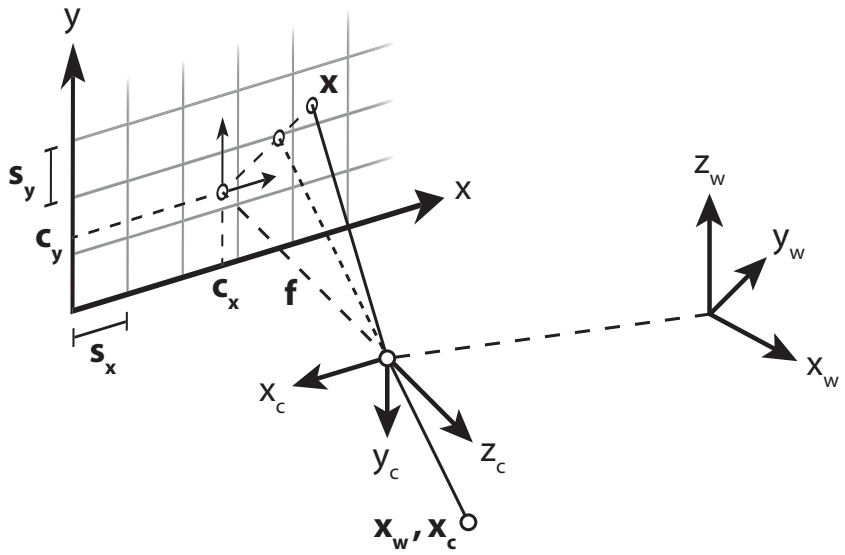


Fig. 4.7. **Pinhole camera with lens distortions.** The projection is affected by the non-linear distortion of the lens. While the dotted line shows the direct projection of the point through the pinhole, the lens causes a distortion which forces the light to be slightly deviated. The difference of the dots on the sensor shows the effect on pixel level.

Our supposed model has one disadvantage, though: It does not take the lens of the camera system into account, yet. The spherical shape of the lens causes **radial distortions** which modify our actual calculated coordinates $\mathbf{x}$ as illustrated in Fig. 4.7[12], where the dotted line indicates the point without distortion.
This is a non-linear effect and can be approximated for most lenses by the model of Lanser

[12] Figure inspired by Steger, Ulrich, and Wiedemann [395, p. 182].

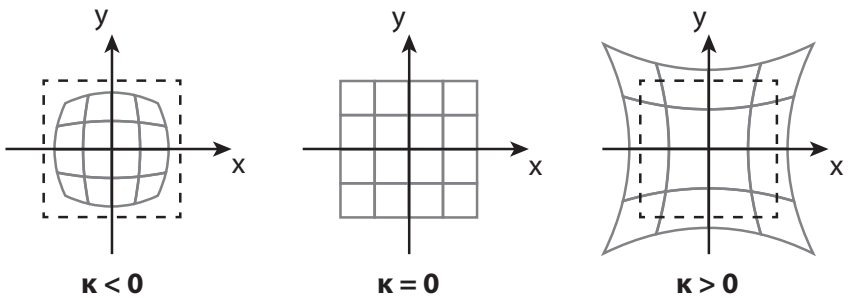


Fig. 4.8. **Effects of radial distortion.** Lens composition is responsible for the parameter κ to take different values. Three main effects can be observed. For $\kappa < 0$ a barrel distortion is visible as shown on the left while $\kappa > 0$ causes a pincushion distortion as illustrated on the right. A value of $\kappa = 0$ prevents distortion.

[235, pp. 45–46]. The following function describes this phenomenon in terms of image coordinates:

$$D_1 : \mathbb{R}^2 \to \mathbb{R}^2 \tag{4.17}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \frac{2}{1 + \sqrt{1 - 4\kappa\,(x^2 + y^2)}} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{4.18}$$

with dimensionless normalized image coordinates x and y. Fig. 4.8 illustrates this effect with different values for the parameter κ. It comes to a barrel-like look of the grid for $\kappa < 0$, and for $\kappa > 0$ to pincushion distortion. If we want to solve this, we have to invert the function D. This is analytically possible and gives

$$D_1^{-1} : \mathbb{R}^2 \to \mathbb{R}^2 \tag{4.19}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \frac{2}{1 + \kappa\,(x^2 + y^2)} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{4.20}$$

Empirical tests show, however, that a more elaborate model better reflects the physical lens-camera system at the cost of losing the ability for an analytic inverse mapping.[13] We consequently integrate a higher order distortion model into our formulation and adopt the ideas of Heikkila et al. [168]. The new distortion model involves a radial distortion component

$$D_2 : \mathbb{R}^2 \to \mathbb{R}^2 \tag{4.21}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \left(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6\right) \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.22}$$

[13] Cf. Heikkila and Silven [168].

with $r^2 = \left(x^2 + y^2\right)$ and the radial distortion coefficients κ_1, κ_2, κ_3 of the lens. In order to address the effect when the image plane and the lens are not parallel, tangential distortion can also be modeled with

$$D_\tau : \mathbb{R}^2 \to \mathbb{R}^2 \tag{4.23}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x + 2\tau_1 xy + \tau_2\left(r^2 + 2x^2\right) \\ y + \tau_1\left(r^2 + 2y^2\right) + 2\tau_2 xy \end{pmatrix} \tag{4.24}$$

where τ_1 and τ_2 depict the tangential distortion coefficients.
Even though this description of the hardware components is more accurate, the issue with the more complex model becomes evident if we look for example at two radial distortion coefficients. The mappings become fifth order polynomials so that an analytic solution for the inverse mapping does not exist anymore. However, Heikkila et al. [168] propose an empirical inverse model which compensates for distortions caused by the model above.
For all our experiments, we utilize the more complex model and decide to work with a radial distortion model with two coefficients, thus choosing $\tau_1 = \tau_2 = \kappa_3 = 0$.

If we write $\mathbf{D}^{-1}$ for a function that acts on homogeneous coordinates of $\mathbb{RP}^2$ by using an inverse mapping D^{-1} for the first two coordinates and keeping the third one fixed, we can now rewrite (4.15) as

$$\mathbf{x} = P\left(\mathbf{x}_W\right) = \underbrace{\mathbf{Q}_3\mathbf{Q}_2\mathbf{D}^{-1} \circ \mathbf{Q}_1}_{\mathbf{K}}\mathbf{R}\left(\mathbf{I} \mid -C_W\right)\mathbf{x}_W \tag{4.25}$$

$$= \underbrace{\mathbf{K}\left(\mathbf{R} \mid \mathbf{t}\right)}_{\mathbf{P}}\mathbf{x}_W \tag{4.26}$$

$$= \mathbf{P}\,\mathbf{x}_W. \tag{4.27}$$

Since the correction of the distortion must happen before the shift of the image coordinate system via the principle point C, we have to apply $\mathbf{D}^{-1}$ after $\mathbf{Q}_1$. As denoted above, we define the whole process also as $\mathbf{K}$ for a simpler notation even though the non-linear operation $\mathbf{D}^{-1}$ is involved. Likewise we treat the whole function P in further sections as a projection matrix $\mathbf{P}$ to ease the description.

4.1.3. Image to World

The general triangulation of an arbitrary 3D world point is considered in chapter 6.3. Later we argue that in favour of reconstructing its coordinates, it is helpful to utilize two acquired images of the same scene from different perspectives. In this section though, we concentrate on establishing 2D-3D correspondences of known real-world points and their projections in order to determine the camera parameters. Hence, we focus on the special case of **triangulation of co-planar points** with only one acquired image and construct a method for intersecting the visual ray of an image point through the camera centre with this plane.
In terms of the different coordinate systems of our camera model, we can for example choose

the plane $z_W = 0$. For application reasons, we model the coordinate points $\mathbf{x}_W^i$, $i \in \{1, \ldots, n\}$ as centroids of circular markers on some planar target within this plane as illustrated in Fig. 4.9[14].

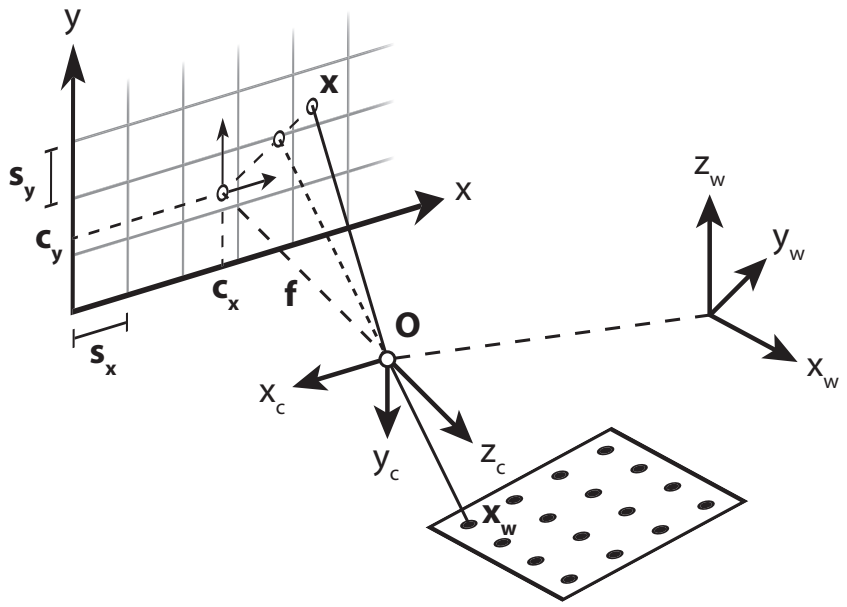


Fig. 4.9. **Pinhole camera and co-planar world points.** The individual centroids of circular markers on a planar target are projected onto the image plane to provide 2D-3D correspondences. After correction for the distortion effects, the pixel location of $\mathbf{x}$ changes and it is possible to shoot a ray through the camera centre O (densely dashed line). This ray intersects with the target in the circle centre $\mathbf{x}_W$.

Circular markers are projected as ellipses onto the image plane. In section 3.9, we established Algorithm 3.5 to calculate the pixel locations of the projected points $\mathbf{x}^i$ in image coordinates with sub-pixel precision. For now, we can think of an image processing pipeline as a black box which can determine the marker centres and investigate the geometric problem first.
Thus, inversion of the steps from equation (4.25) gives a mapping for the homogenized image point

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{pmatrix} = \tilde{\mathbf{x}} = \mathbf{D} \circ \mathbf{Q}_2^{-1} \mathbf{Q}_3^{-1} \mathbf{x}. \tag{4.28}$$

This introduces a point $\tilde{\mathbf{x}}_C = (\tilde{x}, \tilde{y}, f)$ to the virtual image plane. With $\tilde{\mathbf{x}}_C$ and the origin of the camera coordinate system $\mathbf{O}_C$, we can form the visual ray through the camera centre as a line

$$l_C = \{\mathbf{O}_C + \eta \tilde{\mathbf{x}}_C \mid \eta \in \mathbb{R}\}. \tag{4.29}$$

And hence we have the desired line which we want to intersect with the target plane. Since $z_W = 0$, this plane is given by the exterior camera parameters $\mathbf{R}$ and $\mathbf{t}$. It could be transformed

[14] Figure inspired by Steger, Ulrich, and Wiedemann [395, p. 182].

into the camera system, too. We, however, transform the visual line into world coordinates. A point in camera coordinates $\mathbf{x}_C$ transforms into world coordinates $\mathbf{x}_W$ by

$$\mathbf{x}_W = \mathbf{R}^{-1}(\mathbf{x}_C - \mathbf{t}) = \mathbf{R}^{\mathrm{T}}(\mathbf{x}_C - \mathbf{t}). \tag{4.30}$$

This brings two points on the transformed line, the camera centre $\mathbf{O}_W = -\mathbf{R}^{\mathrm{T}}\mathbf{t}$ and the point

$$\tilde{\mathbf{x}}_W = \mathbf{R}^{\mathrm{T}}(\tilde{\mathbf{x}}_C - \mathbf{t}). \tag{4.31}$$

Joining them yields the desired line

$$l_W = \{\mathbf{O}_W + \eta(\tilde{\mathbf{x}}_W - \mathbf{O}_W) \mid \eta \in \mathbb{R}\} \tag{4.32}$$

$$= \{\mathbf{O}_W + \eta\mathbf{d}_W \mid \eta \in \mathbb{R}\} \tag{4.33}$$

in world coordinates. Writing $\mathbf{O}_W = \left(O_W^x, O_W^y, O_W^z\right)$ and $\mathbf{d}_W = \left(d_W^x, d_W^y, d_W^z\right)$, we can intersect this line with $z_W = 0$ and get the 3D world coordinate vector

$$\mathbf{x}_W = \begin{pmatrix} O_W^x - O_W^z d_W^x \left(d_W^z\right)^{-1} \\ O_W^y - O_W^z d_W^y \left(d_W^z\right)^{-1} \\ 0 \end{pmatrix} \tag{4.34}$$

on the target plane.

We now use the two methods from the sections on ***world to image*** (4.1.2) and ***image to world*** (4.1.3) to formulate a procedure to adjust the parameters of our pinhole camera model.

4.2. Camera Calibration

The model introduced previously has six degrees of freedom for the extrinsic camera parameters, three for both the translation $\mathbf{t} = \left(t_x, t_y, t_z\right)$, and the angles α, β, and γ of the rotation $\mathbf{R}$. With $C = \left(c_x, c_y\right)$, f, and s_x, s_y, there are five degrees of freedom for the intrinsic parameters from which we keep the pixel size s_y fixed. The modelled distortion parameters κ_1, κ_2 add another two degrees of freedom for a total of **13** (12 with fixed s_y) **degrees of freedom**.

Now imagine an image acquisition with a **calibration target** as shown in Fig. 4.10 where the centroids of the dots of the planar target code the position of the n world points $\mathbf{x}_W^i$, $i \in \{1, \ldots, n\}$.

As a first step, we detect the calibration target itself.[15] Given its characteristic structure, what we do is to look for a large and bright connected region with at least n dark holes inside of it. This can be done using the principles from section 3.4 and section 3.6. Since circles transform to ellipses via projection, we can then approximate the centroids within the separated area in terms of image coordinates with Algorithm 3.5. The whole process is illustrated in Fig. 4.10.

[15] Cf. Lanser [235, p. 51].

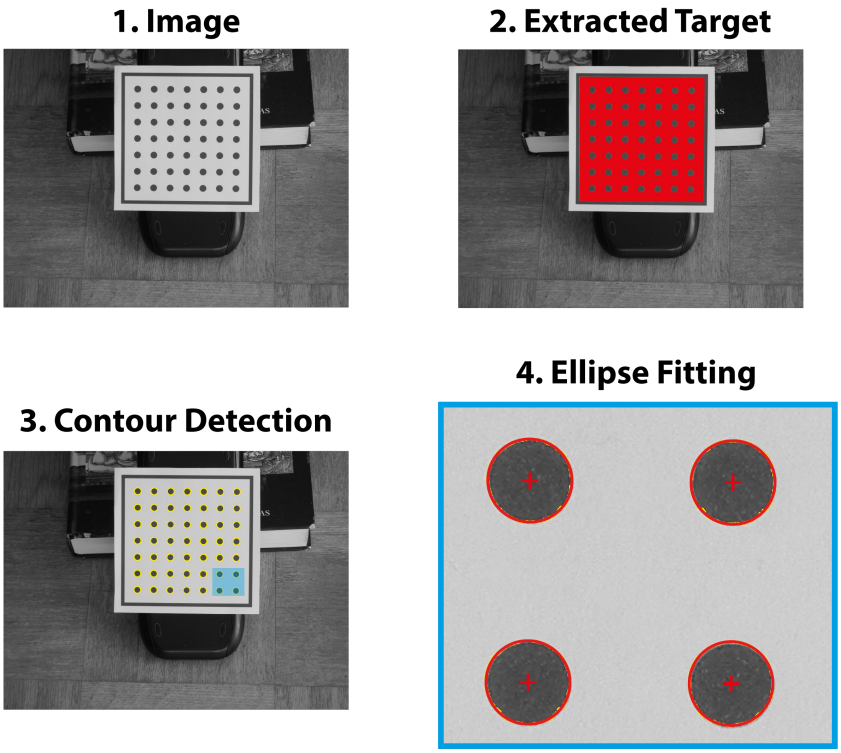


Fig. 4.10. **Extracting coordinates from calibration target.** After image acquisition, the full calibration target is detected (2.). Separated into regions with ellipses (3.). For each individual region, a sub-pixel precise contour detection is performed and an ellipse fit generates the sub-pixel precise centroid coordinates shown in red (4.).

Suppose the centroids of the projections are given by $\mathbf{x}_i$ with $i \in \{1,\ldots,n\}$. A parameter estimation for the parameters

$$\mathbf{r} = \left(f, s_x, s_y, c_x, c_y, \kappa_1, \kappa_2, \alpha, \beta, \gamma, t_x, t_y, t_z\right) \tag{4.35}$$

from n points can be formulated as the **non-linear optimization problem**

$$\min \sum_{i=1}^{n} \left\| \mathbf{x}_i - \mathbf{P}\left(\mathbf{x}_W^i, \mathbf{r}\right) \right\|^2 \tag{4.36}$$

which minimizes the squared geometric error of a point projected by our model and the actual point measured in the image.
For a whole set of m images where the calibration target determines the exterior orientations and is moved around in between two shots as shown in Fig. 4.11, we get

$$\mathbf{r} = \left(f, s_x, s_y, c_x, c_y, \kappa_1, \kappa_2, \mathbf{e}_1, \ldots, \mathbf{e}_m\right), \tag{4.37}$$

$$\mathbf{e}_j = \left(\alpha_j, \beta_j, \gamma_j, t_x^j, t_y^j, t_z^j\right) \tag{4.38}$$

with the **multi-image minimization**

$$\min \sum_{j=1}^{m} \sum_{i=1}^{n} \left\| \mathbf{x}_{i,j} - \mathbf{P}\left(\mathbf{x}_W^{i,j}, \mathbf{r}\right) \right\|^2. \tag{4.39}$$

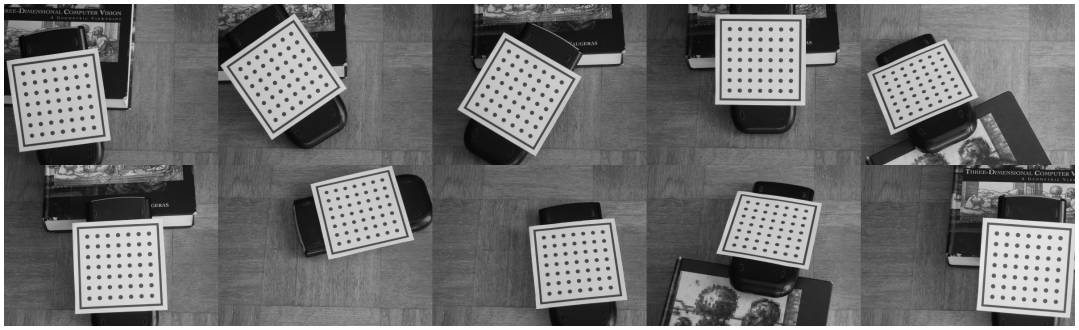


Fig. 4.11. **Multi-image calibration.** Each acquisition gives a set of 49 co-planar points together with 6 extrinsic parameters. Multiple images increases the amount of data points for our optimization problem. The intrinsic parameters remains the same while the extrinsic values vary between different acquisitions.

To gain a solution to this non-linear minimization problem, we use the iterative Levenberg-Marquardt algorithm.[16] Since the radial correction of the image depends on the minimization itself, we iteratively calculate the minimum and use the calculated radial correction for a new loop until the parameter change is negligible.
Besides that, we require several reliable initial parameters for this minimization to converge. For the intrinsic parameters, the actual manufacturer information shall suffice. However, we also need an estimation for the extrinsic parameters given by $\mathbf{e}_1, \ldots, \mathbf{e}_m$. This is indeed a non-trivial task and, following the ideas presented by Lanser [235, pp. 52–53], we can detect the coordinates of the markers in terms of the camera coordinate system at first by using the circular shape of our markers. That is to say, the 3D coordinates of the centroid of a circular marker for the point with the virtual image coordinates $\tilde{\mathbf{x}}_C = (\tilde{x}, \tilde{y}, f)$ can be estimated by

$$\mathbf{m} = \begin{pmatrix} m_x \\ m_y \\ m_z \end{pmatrix} = \begin{pmatrix} rfa^{-1} \\ m_z f^{-1} \tilde{x} \\ m_z f^{-1} \tilde{y} \end{pmatrix}. \tag{4.40}$$

The parameters for this are the radius r of the dot and the major axis a of the corresponding ellipse. We then calculate a regression plane through the point cloud of n markers and determine the normalized normal vector $\mathbf{n}_1$ of it. $\mathbf{n}_1$ can then be used to calculate the rotation $\mathbf{R}$ which is given by a transformation that maps the vector $\mathbf{n}_1$ perpendicular to the plane $z_W = 0$ to the vector $\mathbf{n}_2 = (0, 0, 1)$ perpendicular to the plane $z_C = 0$.
As the origin of the world coordinate system, we use the 3D centroid C^{avg} of all the n markers given by equation (4.40) so that we get the translation vector $\mathbf{t} = \mathbf{R}^{\mathrm{T}} C^{avg}$.
Summing up all the results in this chapter, we formulate Algorithm 4.1 for the estimation of the camera parameters $\mathbf{r}$.

The camera calibration is essential for accurate measurements. Stability and reliability of the algorithms in all consecutive chapters depend on this procedure. Instead of planar boards with circular markers other commonly used calibration targets include checkerboards as shown in Fig. 4.12 or fiducial markers as discussed in chapter 7.2.

[16] Cf. Hartley and Zisserman [165, pp. 600–602].

Algorithm 4.1. Camera Calibration

Input parameters:

- Calibration target with n circular dots of radius r
- m images $\mathbf{I}^j$, $j \in \{1, \ldots, m\}$ of the target

Preprocessing:

1. Initialize intrinsic parameters $\mathbf{r} = \left(f, s_x, s_y, c_x, c_y, \kappa_1, \kappa_2\right)$
2. Further parameters: ε. Initialize: $\tau = \infty$

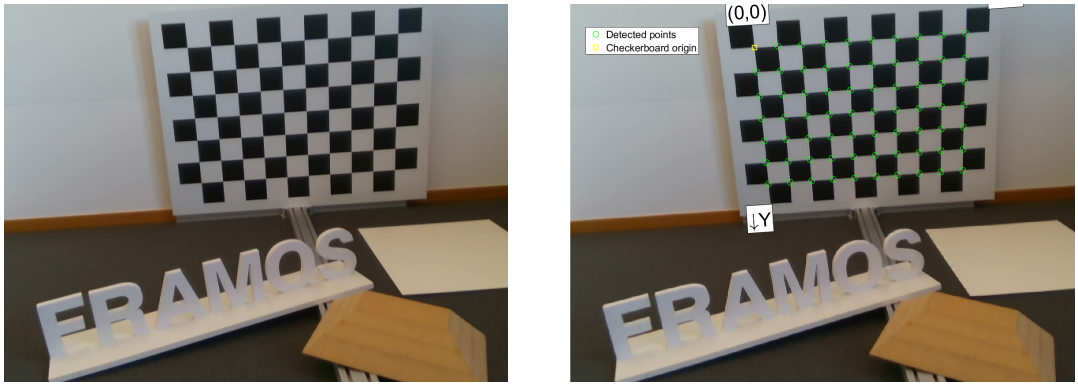
Computation steps:

while $|\tau^{prev} - \tau| < \varepsilon$ ***or*** *first iteration* **do**

- Save previous residual: $\tau^{prev} = \tau$
- // Collect 2D image coordinates of markers
- **for** $j = 1$ **to** m **do**
  - Extract region R_j of calibration target
  - Perform coordinate detection on R_j to get set Q_j (Algorithm 3.5)
    - Save major axis $a_{i,j}$
  - Reset counter: i = 0
  - // Estimate exterior parameters of image $\mathbf{I}^j$ (Section 4.2)
  - **for** $\boldsymbol{x}_j \in Q_j$ **do**
    - Increment counter $i = i + 1$ and label points $\mathbf{x}_{i,j} = x_j$
    - Get 3D marker coordinates $\mathbf{m}_{i,j}$ of $\mathbf{x}_{i,j}$ with $a_{i,j}$ (Equation (4.40))
  - // Check number of markers on calibration target
  - **if** $i < n$ **then**
    - Skip image $\mathbf{I}^j$ and start loop again with next $j = j + 1$
  - **else**
    - Calculate regression plane E_j and average marker centroids: C_j^{avg}
    - Estimate exterior parameters with C_j^{avg} and norm. normal vector of E_j:
      $\mathbf{e}_j = \left(\alpha_j, \beta_j, \gamma_j, t_x^j, t_y^j, t_z^j\right)$
    - // Calculate world points and image points by modelled projection
    - **for** $\boldsymbol{x}_{i,j} \in Q_j$ **do**
      - Transform image coordinates to world: $\mathbf{x}_{i,j} \mapsto \mathbf{x}_W^{i,j}$ (Section 4.1.3)
- Estimate parameter by minimization (Levenberg-Marquardt):
  $\min \sum_j \sum_i \left\| \mathbf{x}_{i,j} - \mathbf{P}\left(\mathbf{x}_W^{i,j}, \mathbf{r}, \mathbf{e}\right) \right\|^2$ (Equations (4.25), (4.39))
- Update initialization parameters: $\mathbf{r}$, $\mathbf{e}$, and current residual τ

Output:

- Intrinsic camera parameters: $\mathbf{r} = \left(f, s_x, s_y, c_x, c_y, \kappa_1, \kappa_2\right)$
- Exterior camera parameters: $\mathbf{e} = (\mathbf{e}_1, \ldots, \mathbf{e}_m)$ with $\mathbf{e}_j = \left(\alpha_j, \beta_j, \gamma_j, t_x^j, t_y^j, t_z^j\right)$



RGB Image **Checkerboard Detection**

Fig. 4.12. **Calibration board with checkerboard pattern.** A common alternative to circular markers on a planar target is a planar checkerboard with squares of known size. An RGB or greyscale image (left) is then used with a detector for the checkerboard corners that are illustrated in green on the right. The coordinate origin (yellow) is chosen as the corner of the detected grid and determines the x- and y-coordinate origin. The sub-pixel precise calculations of corner coordinates are fed into Algorithm 4.1 to determine the calibration parameters identical to the ellipse coordinate centres.

After all, the camera calibration can be done offline, which makes the computation time non-critical. During the process of image acquisition it is helpful to pay attention that the calibration target is moved to different viewing angles at various distances in order to reduce the error rate of the estimated parameters.

Until now, we presented methods with **classical image processing** techniques and algorithms which are used in this thesis to realize modern high performance 3D vision systems. As a result of this, the outcome algorithms and pipelines are used throughout this document.

All previously discussed methods have in common that they target a highly specific problem and solve it automatically with a handcrafted pipeline for the task. The design of a solution for a computer vision problem, however, is not always evident or reliably possible in general. This holds true in particular if the input structure is complex or the sample distribution is difficult to model. **Data-driven methods** are often powerful tools to abstract image data and solve these specific task by generalizing underlying patterns automatically. In the following, we study statistical methods from machine learning that help to meet our objectives. Artificial neural networks formed by optimizing parameters of highly nonlinear functions in order to generalize from a training dataset to further samples are therefore considered next.

5 Neural Networks

> *A deep-learning system*
> *doesn't have any explanatory power.*
>
> **– Geoffrey Hinton**

Powerful machine learning algorithms benefit from the large amount of data present in many parts of our life to extract patterns and predict trends. Deep-learning systems can leverage this information and find structure in features extracted from it. Neural networks are one way to design such systems that learn from our data. More complex nets can thereby provide more accurate predictions but extracted features may be harder to interpret at the same time. They are powerful tools that provide high-parametric models able to mimic nonlinear processes by means of data analysis.
We want to touch here the basic concepts, designs and training mechanisms of neural nets and state their potential such that we can use them for image processing afterwards.

As the name suggests, an **artificial neural network** (artificial NN) is an artificially-designed ensemble of connected neurons which have vague resemblance to biological neural networks in the brain where interconnected synapses react on certain stimuli.[1] We specifically focus on multi layer networks with the capability of representing hierarchical structured abstraction levels to enable content specific machine learning. Such deep neural networks can help to generalize a task from a set of training examples to previously unseen data with similar distribution or patterns by optimization the parameters of the architecture. In vision, such deep learning approaches significantly enhanced the performance of a wide variety of tasks involving recognition, classification, segmentation, regression and many more.
The aim of this chapter is by no means to give a complete overview over the advances of this briskly progressing and remarkable research field, but rather to provide the general ideas for the concepts used in this thesis. There are exceptional works from very experienced researcher in this domain that conduct a more in-depth analysis of neural networks, their applications and the theoretical foundation which go way beyond the scope possible in this chapter.
For a sound theoretical background and mathematically rigorous presentation, the authoritative work of Goodfellow et al. [152] is an excellent resource and a brief introduction is given by LeCun, Bengio, and Hinton [239] as a starting point to study this field. A more applied and practical access is given by Rosebrock [352], who explains the concepts hands-on with many code examples.
In the following, we recap the idea of artificial neural networks as a potent tool in machine learning. We briefly justify their use and explain both a general vanilla network structure and the involved components. Finally we conclude with an overview of parameter optimization.

[1] Cf. Purves et al. [335].

5.1. Multilayer Perceptron

Suppose you have to solve the assignment task F where you assign a value $y \in Y$ to a given value $x \in X$ with some criteria.[2] If you want to formalize the ideal way to solve this job, you can imagine the function $F : X \to Y$ that describes this assignment. A neural network can be seen as a parametric approximation of F that can learn to solve such a task by optimization of certain intrinsic parameters with the help of data. More specifically, a neural network can be viewed as a function $f : X \to Y$ where an input $x \in X$ is fed into the network which ideally produces the output $y \in Y$. A classical use case is a non-linear separation of the input space according to a set of labeled training data points $x_i \in X$ in order to classify images for example. The function f has in general a multitude of parameters and can be decomposed into sub-functions that are connected through a graph which defines the network **architecture**.
In order to identify the components of such an architecture and to clarify concepts, we take a look at a specific class of neural networks and analyze the so called **multilayer perceptron** shown in Fig. 5.1 in more detail.

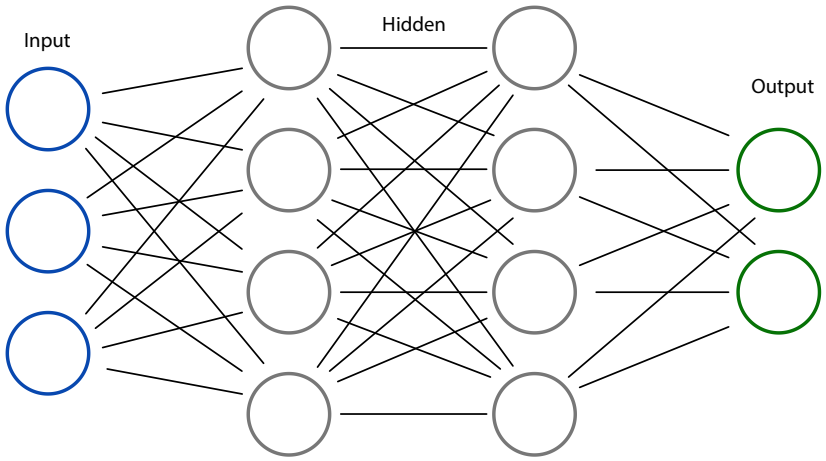


Fig. 5.1. **Multilayer perceptron.** The architecture shows a 4-layer neural network whose information flow is from left to right. The circular nodes represent the neurons and the directed connections are illustrated with lines. The input neurons are connected to a first hidden layer which is fully connected with a second hidden layer before feeding the two output neurons.

A multilayer perceptron (MLP) is a specific class of feedforward neural net well suited to explain naming conventions. The term feedforward refers to the fact that the architecture can be represented as a directed acyclic graph as illustrated in Fig. 5.1.[3] The network consists of hierarchically ordered neurons, illustrated as circles and connections between them which are marked as lines. Let us imagine first, that the neural network is already **trained**. This means, that the network parameters have been optimized in a way that the function f represented by the network approximates the assignment task F sufficiently well.[4]
Imagine each neuron in the network as a unit holding a real number $a \in \mathbb{R}$ which we call the **activation**. High numbers correspond to activated neurons while low numbers describe inactive neurons at a specific point. A high number in a neuron $\bar{y} \in Y$ in the output layer, for

[2] For our purposes, we can think of X and Y as subsets of $\mathbb{R}^d$ with $d \in \mathbb{N}$.
[3] Networks with cyclic graphs are usually called Recurrent Neural Networks (RNNs).
[4] We will look into the optimization part (i.e. training) of the neural network in section 5.4.

instance, represents the response of the system to a certain input stimulus $\bar{x} \in X$ suggesting a correspondence of $\bar{x} \longleftrightarrow \bar{y}$ for the assignment F in our example.
The way information is processed in the network is determined by the way activations of neurons in layer l influence the activations of neurons in layer $l+1$. The idea behind it is that early layers pick up information very close to the data input while deeper layers respond on higher level information from the earlier extracted activations. Speaking of images, the flow could be from input pixel values over edge detection to image patterns ending with image concepts or classes represented by a certain output probability. However, feeding images motivated by this idea is discussed with the concepts of filters from part 3.3 in section 5.3.
Let us focus on an input $\bar{x} \in \mathbb{R}^d$ which can be imagined also as grey values for an image of resolution $d = w \times h$ with width w and height h. The model used to calculate the activation of the next column follows the idea illustrated in Fig. 5.2.

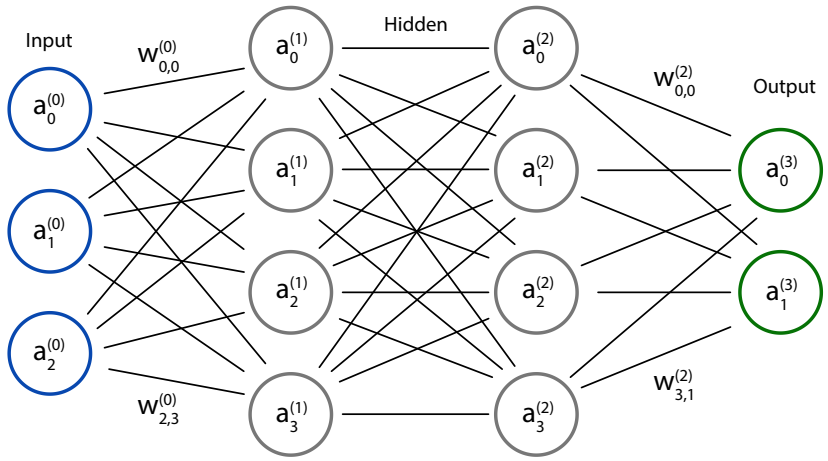


Fig. 5.2. **Weights and activations of a multilayer perceptron.** Each neuron i at layer l is assigned to an activation function $a_i^{(l)}$. The edges in the graph represent the weights of the activation, where one weight $w_{i,j}^{(l)}$ is the amount of influence of the neuron i of layer l on the activation of neuron j in layer $l+1$. For illustration purposes not all associated weights are drawn here.

Each neuron can hold an activation $a_i^{(l)}$ where l is the layer number an i represents the neuron number within this layer. For the input layer this could be the grey value of a certain pixel. Every edge going out of layer l in the graph is assigned a weight $w_{i,j}^{(l)}$ where i is the neuron number in layer l and j represents the neuron number in layer $l+1$.
In order to calculate the activation in a certain neuron j of layer $l+1$, a weighted sum of the connected neurons from the previous layer is chosen such that

$$a_j^{(l+1)} = \sigma\left(\sum_{i=0}^{k}\left(w_{i,j}^{(l)} a_i^{(l)}\right) + b_j^{(l+1)}\right) \tag{5.1}$$

together with a bias $b_j^{(l+1)}$ for inactivity and a differentiable nonlinearity $\sigma : \mathbb{R} \to \mathbb{R}$ that guarantees that a change in the input causes some change in its output which can be amplified by the nonlinearity around a certain threshold. Commonly used **activation functions** include the sigmoid function

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{5.2}$$

and a rectifier to employ a rectified linear unit (ReLu)

$$\sigma(x) = \max(0, x) \tag{5.3}$$

as introduced by Glorot, Bordes, and Bengio [142].[5] This forces the network to fire if and only if the common activations are bigger than a bias of $-b_j^{(l+1)}$.
The concept of equation (5.1) can be summarized for a whole layer in matrix-vector notation as

$$\mathbf{a^{(l+1)}} := \left(a_0^{(l+1)}, a_1^{(l+1)}, \dots, a_n^{(l+1)}\right)^{\mathrm{T}} \tag{5.4}$$

$$= \sigma\left(\begin{pmatrix} w_{0,0}^{(l)} & w_{1,0}^{(l)} & \cdots & w_{k,0}^{(l)} \\ w_{0,1}^{(l)} & w_{1,1}^{(l)} & \cdots & w_{k,1}^{(l)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{0,n}^{(l)} & w_{1,n}^{(l)} & \cdots & w_{k,n}^{(l)} \end{pmatrix} \begin{pmatrix} a_0^{(l)} \\ a_1^{(l)} \\ \vdots \\ a_k^{(l)} \end{pmatrix} + \begin{pmatrix} b_0^{(l+1)} \\ b_1^{(l+1)} \\ \vdots \\ b_n^{(l+1)} \end{pmatrix}\right) \tag{5.5}$$

$$=: \sigma\left(\mathbf{W^{(l)}}\mathbf{a^{(l)}} + \mathbf{b^{(l+1)}}\right) \tag{5.6}$$

where σ acts on each vector component individually.
With this model in mind we can look at each neuron as a function that responds with a value $a \in \mathbb{R}$ to a given set of stimuli from connected neurons in the previous layer. Thus the entire network is a function composed of these individual neural sub-functions.

5.2. Universal Approximation Theorem

In general, a feedforward multilayer perceptron with a single hidden layer and nonconstant, bounded, and continuous activation function can serve as a **universal function approximator** as shown by the theoretical result of Cybenko [76] for sigmoid activation functions as in equation (5.2). Hornik [183] generalizes the result to more generic activation functions. They prove that any continuous function F on compact subsets of $\mathbb{R}^k$ can be approximated by a feedforward network with one hidden layer and weights according to equation (5.6). Rephrasing the theorem for a single output in our notation with the definitions in equation (5.1), we can state

Theorem 5.1 (Universal approximation)
Let the activation function $\sigma : \mathbb{R} \to \mathbb{R}$ be nonconstant, bounded, and continuous. Let I_k with $\boldsymbol{a^{(0)}} \in I_k$ denote a compact subset of $\mathbb{R}^{k+1}$ and $C(I_k)$ the space of continuous functions $F : I_k \to \mathbb{R}$. Then it holds:

$$\forall\, F \in C(I_k), \varepsilon > 0\ \exists\, N \in \mathbb{N}, v_j, b_j^{(1)} \in \mathbb{R}, w_j^{(0)} = \left(w_{0,j}^{(0)}, \dots, w_{k,j}^{(0)}\right)^T \in \mathbb{R}^{k+1} \text{ with } j \in \{0, 1, \dots, N\}:$$

$$f\left(\boldsymbol{a^{(0)}}\right) = f\left(\left(a_0^{(0)}, a_1^{(0)}, \dots, a_k^{(0)}\right)^T\right) := \sum_{j=0}^{N} v_j\, \sigma\left(\sum_{i=0}^{k}\left(w_{i,j}^{(0)} a_i^{(0)}\right) + b_j^{(1)}\right) \tag{5.7}$$

[5] Other activation functions include hyperbolic tangent, softmax, leaky ReLu and many more that cannot be addressed in the scope of this thesis.

approximates the function F which is independent of σ in a way that it holds

$$\left|f\left(\boldsymbol{a}^{(0)}\right)-F\left(\boldsymbol{a}^{(0)}\right)\right|<\varepsilon \quad \forall\, \boldsymbol{a}^{(0)}\in I_k. \tag{5.8}$$

This means, that the functions f represented by the single layer neural network are dense in $C(I_k)$ which can be directly deduced to the general case of multiple output neurons. As the proofs are not constructive regarding the network architecture and the training, the amount of neurons for a specific approximation accuracy and the optimization complexity remain unclear. The tremendous success of neural networks in various scientific fields in the last decade, however, indubitably demonstrates the practical performance of these pipelines. We focus our attention now more towards vision-specific aspects of neural networks and combine the architectural ideas presented in section 5.1 with the classical vision concepts of chapter 3.3.

5.3. Convolutional Neural Networks

The biological brain studies of Hubel and Wiesel [189] and Hubel et al. [190] show the reaction of neurons in the visual cortex in certain mammals is restricted to a specific stimulus in the visual field. Their study shows that a simple local visual stimulus such as a straight edge of a specific orientation causes the activation of certain neurons while a change in location induces other cells of the brain region to react. Moreover, cells that are close to each other react on similar regions. The space causing a response in a specific cell is called its **receptive field**. A second category of more complex cells is less sensitive to the location of the input visual signal, however, they react on specific visual patterns.
The idea of **Convolutional Neural Networks** (CNNs) is somewhat similar and combines low level cues such as edge information with more abstract visual concepts in hierarchical order on top of them. Image data is fed into a specific variation of a shift invariant multilayer perceptron where a convolutional layer (i.e. a filter as described in section 3.3) is applied to the data of its receptive field without being fully connected to every pixel in the image. According to Definition 2.2 and Definition 3.3, we formulate

Definition 5.1
*For an image $f: D \to \mathbb{G}_d$ with $(x,y)\mapsto g_{x,y}$ and domain $D=\{0,1,\dots,w-1\}\times\{0,1,\dots,h-1\}$, the output of a **convolutional layer** with a filter bank of size n formed by kernel functions k_i, $i\in\{0,\dots,n-1\}$ is a **feature map** given by*

$$\boldsymbol{M}(x,y,i)=(k_i\star g)_{x,y}\,. \tag{5.9}$$

A convolutional layer has thus parameters for the amount of kernels and their dimension, stride and padding such that the filter moves according to the stride over the image while usually a zero-padding guarantees the correct input image crop for the convolution especially at image

borders.[6] Deeper layers can convolve feature maps to new feature maps by summation over the third dimension accordingly or are fully connected such that the receptive field at deeper levels increases.

CNNs are widely used for deep learning in vision tasks where the use of $k \times k$ filters with **shared weights** requires significantly fewer parameters than a fully connected layer that connects all pixels. A filter stack with n kernels, for instance, requires $n \cdot k^2$ parameters, while a fully connected layer has $w \cdot h$ weights for an image of size $w \times h$. This is a relevant difference in most common cases where $k \ll w$ and $k \ll h$.

Apart from the weight sharing, a frequently used concept is **pooling** where for instance the maximum value (max pooling) or the average (average pooling) of a neuron cluster after a nonlinearity is condensed with this operation to one single neuron in order to progressively reduce the size of the feature map. This also reduces the amount of output parameters. Fig. 5.3 shows a classical model proposed by LeCun et al. [241] which involves the presented concepts to recognize hand-written digits.

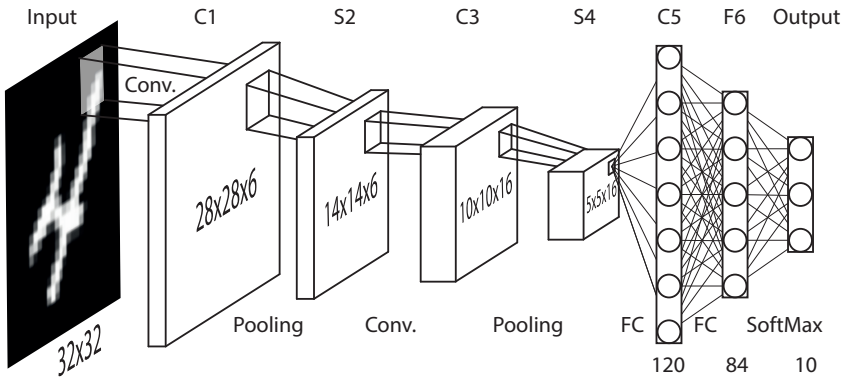


Fig. 5.3. **LeNet-5.** The graphic illustrates the pioneering model of LeCun et al. [241] for handwritten digit recognition. The 7-layer CNN consists of a hierarchical chain of convolutional layers, subsampling layers using average pooling and fully-connected layers. Extracted feature maps feed a fully connected layer. The nonlinearity is a hyperbolic tangent and the final layer uses a softmax function in order to obtain probabilities for each digit. The architecture has a total amount of 60k parameters.

In order to efficiently learn from image data for large scale tasks with CNNs, it took over a decade from these early approaches until Krizhevsky, Sutskever, and Hinton [228] published an architecture with over 62M parameters which significantly improved image based object recognition with convolutional neural nets. The authors make efficient use of modern GPUs, perform **data augmentation** and use **dropout**[7] for regularization to prevent over-fitting while optimizing the weights.

In contrast to early CNN scholars such as Denker et al. [83] who use manually designed kernels for image recognition, we want to automatically optimize the weights in a training stage which we specify hereafter.

[6] Strictly speaking, a convolutional layer uses a cross-correlation filter rather than a convolution according to Definition 3.2, however, the term "convolution" is most commonly used and equal up to kernel indexing which is why we do not differentiate here.

[7] Cf. Hinton et al. [175].

5.4. Training

Until now, we analyzed different network architectures and discussed specific layers. However, we did not speak about the actual learning process, yet. In this section, we focus on the parameter training and go back to the example of a multilayer perceptron for the purpose of notation as presented in section 5.1.
At first let us assume that all weights are initialized with random numbers. To this end, if we feed a particular input $x_i \in X$ to the network, it produces an output $f(x_i)$ which in general is far away from solving the assignment task $x_i \hookleftarrow y_i$ for the correct response $y_i \in Y$. We can now define a cost for this output

$$C(x_i, y_i) : X \times Y \to \mathbb{R}. \tag{5.10}$$

One such example could be the use of an L^p-norm in the real vector space $\mathbb{R}^d$ with

$$C(x_i, y_i) := \|f(x_i) - y_i\|_p = \left\|\mathbf{a}^{(\mathbf{L})} - y_i\right\|_p \tag{5.11}$$

where $\mathbf{a}^{(\mathbf{L})}$ defines the activations in the output layer L and y_i can either be given directly (**supervised learning**) or can be a function of the input data (**self-supervised learning**); in the latter, the data provides the supervision.
As C depends on all weights and biases of the network and our goal is to change them such that the average cost for our training examples decreases, we take a look at the derivatives of C in order to **optimize** the **weights and biases**.
A naive way to solve this task would be to minimize the average cost of all training data such that our objective becomes

$$\min \frac{1}{\mathrm{M}} \sum_{i \in I} C(x_i, y_i, \mathbf{W}), \tag{5.12}$$

where $I = \{1, \ldots, M\}$ indicates the labeled input data points and $\mathbf{W}$ represents all weights and biases of the network. Thus in order to minimize the objective, we can use gradient descent with stepsize η and calculate

$$-\eta \nabla C(\mathbf{W}), \ \eta \in \mathbb{R}. \tag{5.13}$$

Due to the large number of parameters and the nonlinear activation functions, the loss function embedded in the multi-dimensional parameter space is highly non-convex. For such problems it can be very difficult to find a global minimum and gradient descent brings us only iteratively to a local minimum. However, Choromanska et al. [69] show that most local minima are of similar quality in large-size networks and the search for the global minimum is prone to overfitting in practice. Thus, we seek an efficient way to optimize the network parameters using gradient descent.
Let us take one step back and analyze the two consecutive last layers of the network conceptually. Neurons that react on a specific pattern have high activation if the specific stimulus occurs in the input. On the other side, if a high activation is desired, the neurons reacting on this pattern should be strongly linked to it. This concept can also be found in the biological behaviour of the visual cortex as observed by Hebb [167] and Lowel and Singer [265] who note that “neurons wire together if they fire together”. For artificial neural networks this holds true

not only for the second last layers but even before. However, as the cost function is evaluated with the output of the last layer whose neurons are connected to entries in the second last layer, we want to first update their weights and propagate the error backwards.

5.4.1. Backpropagation

Backpropagation[8] is one mechanism commonly used to train convolutional neural networks since its first use for CNNs in the early 1990s.[9] To calculate the derivative with respect to the network parameters, we introduce the shorthand for a part of equation (5.1) such that

$$z_j^{(l+1)} := \sum_{i=0}^{k} \left(w_{i,j}^{(l)} a_i^{(l)} \right) + b_j^{(l+1)} \tag{5.14}$$

for the weighted sum of activations from layer l to $l+1$. Thus the activations in layer $l+1$ become

$$a_j^{(l+1)} = \sigma\left(z_j^{(l+1)} \right) \tag{5.15}$$

with the nonlinearity σ. If we now focus on the second last layers L and $L-1$, we can conceptualize the information flow as shown in Fig. 5.4 where the activations $a_i^{(L-1)}$ from layer $L-1$ together with their weights $w_{i,j}^{(L-1)}$ and biases $b_j^{(L)}$ form the auxiliary variables $z_j^{(L)}$. These in turn feed nonlinearities σ to calculate the activations $a_j^{(L)}$ which can be compared to $y_j \in Y$ via the cost function C.

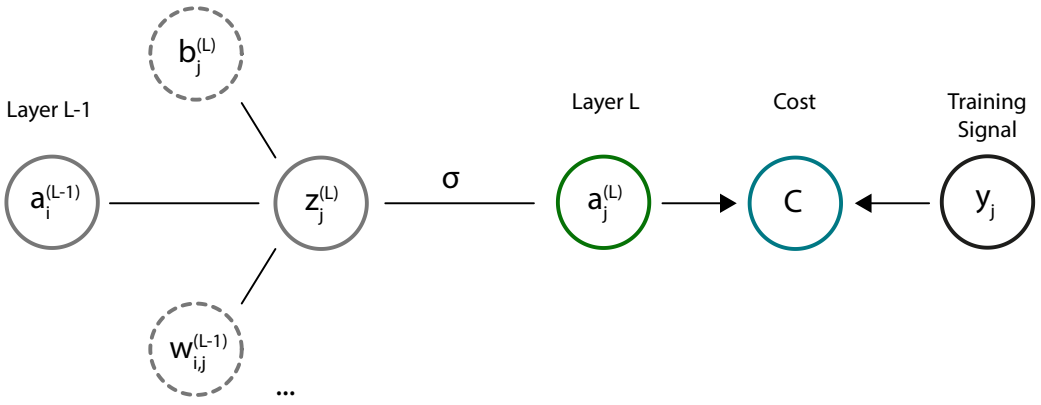


Fig. 5.4. Information flow in neural network. Illustrated is a neuron from the second last layer $L-1$ with activation $a_i^{(L-1)}$ which feeds together with $k+1$ neurons and weights $w_{i,j}^{(L-1)}$ of the same layer and a bias $b_j^{(L)}$ the function $z_j^{(L)}$ which is mapped by the nonlinearity σ to the activation $a_j^{(L)}$ of the neuron j in layer L. Together with the training signal $y_j \in Y$, the cost C can be evaluated.

[8] Cf. Werbos [450].
[9] Cf. LeCun et al. [240].

Applying the chain rule, the derivatives of the cost function with respect to weights, biases and activations as illustrated in Fig. 5.4 become

$$\frac{\partial C}{\partial w_{i,j}^{(L-1)}} = \frac{\partial z_j^{(L)}}{\partial w_{i,j}^{(L-1)}} \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \frac{\partial C}{\partial a_j^{(L)}} \tag{5.16}$$

$$\frac{\partial C}{\partial b_j^{(L)}} = \frac{\partial z_j^{(L)}}{\partial b_j^{(L)}} \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \frac{\partial C}{\partial a_j^{(L)}} \tag{5.17}$$

$$\frac{\partial C}{\partial a_i^{(L-1)}} = \sum_{j=0}^{k} \frac{\partial z_j^{(L)}}{\partial a_i^{(L-1)}} \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \frac{\partial C}{\partial a_j^{(L)}} \tag{5.18}$$

where the last summation combines all $k+1$ neurons from layer $L-1$. Together with

$$\frac{\partial z_j^{(L)}}{\partial w_{i,j}^{(L-1)}} = a_i^{(L-1)}, \quad \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} = \sigma'\left(z_j^{(L)}\right), \quad \frac{\partial z_j^{(L)}}{\partial b_j^{(L)}} = 1, \quad \frac{\partial z_j^{(L)}}{\partial a_i^{(L-1)}} = w_{i,j}^{(L-1)}, \tag{5.19}$$

we are able to calculate the gradient from equation (5.13) as

$$\frac{\partial C}{\partial w_{i,j}^{(L-1)}} = a_i^{(L-1)} \sigma'\left(z_j^{(L)}\right) \frac{\partial C}{\partial a_j^{(L)}} \tag{5.20}$$

$$\frac{\partial C}{\partial b_j^{(L)}} = \sigma'\left(z_j^{(L)}\right) \frac{\partial C}{\partial a_j^{(L)}} \tag{5.21}$$

$$\frac{\partial C}{\partial a_i^{(L-1)}} = \sum_{i=0}^{k} w_{i,j}^{(L-1)} \sigma'\left(z_j^{(L)}\right) \frac{\partial C}{\partial a_j^{(L)}}. \tag{5.22}$$

This holds true not only for the second last layers, but for two common consecutive layers $l-1$ and l. Thus, we can propagate back the error of the network prediction in the final layer L through the entire network up to the input layer 0 and optimize the network parameters.
Due to the iterative use of the chain rule, however, the gradients for activation functions in a small range may vanish. To diminish the risk of **vanishing gradients** and to realize efficient training, modern architectures rely on rectified linear units as proposed by Glorot, Bordes, and Bengio [142].
In practice averaging over all training samples is very time consuming for large training sets. Thus, we use **stochastic gradient descent** by randomly shuffling the training data and dividing it into **mini-batches** which can be used to calculate the gradient. Equation (5.12) then effectively becomes

$$\min \frac{1}{|I_n|} \sum_{i \in I_n} C\left(x_i, y_i, \mathbf{W}\right), \tag{5.23}$$

where $I_n \subset I$ defines a mini-batch with $n \in \{1, \ldots, N\}$. While a gradient calculated from a batch is not the correct one, it still serves as a good approximation to provide a significant speed up. Adaptive optimization variants such as **AdaGrad**, **RMSProp** and its variants[10] as well as **ADAM**[11] are also commonly used to train neural networks.

[10] Cf. Mukkamala and Hein [293].
[11] Cf. Kingma and Ba [219].

Subsequently, we go one step further, leave the two-dimensional world of single images, and discuss the special geometry of binocular computer vision systems with calibrated cameras where we can extract depth information by looking at the same scene from two different viewpoints. The discussed deep learning foundation thereby serves as a tool to enable robust and accurate 3D measurements using convolutional neural networks. Beyond that, the invariance of our centre coordinate extraction approach from section 3.8 helps us to compare and receive information from multiple views when images are acquired together.

Part III

Optical Pose Computation

6 3D Sensing

> *Change the way you look at things*
> *and the things you look at change.*
>
> **– Wayne W. Dyer**

How can we make a computer see in three dimensions? Where do we get the depth information from? There are a lot of different possibilities to tackle this problem. One can either use time-of-flight cameras, interferometry, work with coherence-based methods and with structured light or even detect the shape of a structure by its shading.[1]
The focus of this chapter is on the extraction of spatial information from different sources. We initially discuss different 3D sensing system groups and lay the mathematical foundation to describe motion in space before explaining how multiple sensors can be referenced to each other. For this purpose, we introduce a second camera to imitate the human visual system. After an in-depth study of this two-view setup, we formulate a way to triangulate world points from camera images and analyze their properties according to the acquisition structure. A variety of approaches to extract depth information from images enabled by this knowledge is then developed which we consecutively utilize for marker-based tracking.[2]

6.1. 3D Sensors

Single and multi channel images as introduced in Definition 2.2 can be used to collect 3D measurements of their surroundings using several cameras to observe the scene from different angles as shown in Fig. 6.1. Scene points are viewed from different perspectives and their distance to the cameras can thus be triangulated.

Apart from passive **stereo vision** with cameras, a multitude of other sensing systems for 3D measurements exist and we introduce commonly used principles before diving into the details of geometric methods and applications of these sensing concepts.

[1] For more information on these methods, see Jähne [191, pp. 217–242].
[2] Some parts about epipolar geometry and marker-based pose tracking are improved reprints of Busam [47] to provide a detailed explanation.

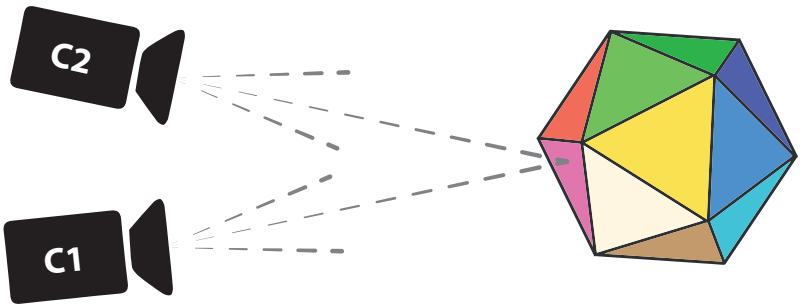


Fig. 6.1. **Passive stereo vision.** The two cameras C1 and C2 observe the scene from two different viewpoints. It is possible to triangulate the coordinates of a 3D point in the overlapping field of view if its location in both images is known.

6.1.1. Structured Light

Passive stereo vision where a scene is viewed by multiple cameras comes with certain problems. In particular areas of low texture are problematic as the correspondence of a scene point in multiple images is difficult to determine without characteristic structure. **Structured light** tackles this problem by adding a known texture to the visible scene via active illumination of the surroundings with a designed intensity pattern.
In general, an active illuminator emits light in the form of a 2D data array, usually an image projected onto the surface geometry. A projector represents an inverse camera and while the chip of a camera is sensitive to electromagnetic waves of a specific spectrum, a projector emits light of these wavelengths or multiple colours as illustrated for a stripe pattern in Fig. 6.2.

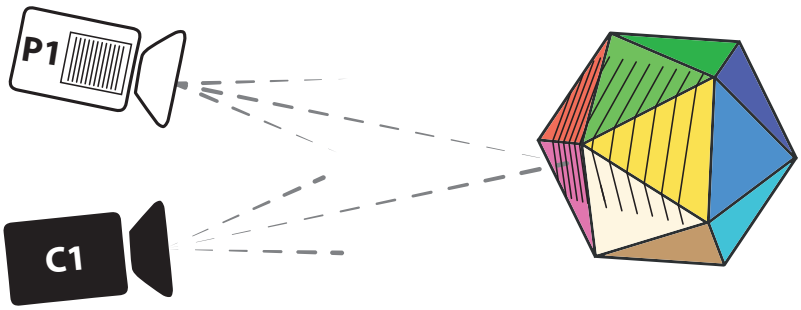


Fig. 6.2. **Active stereo vision with structured light.** The projector P1 acts as an inverse camera and projects a stripe pattern into the scene which is deformed by the object's geometry. A passive camera C1 observes the deformation. A reconstruction of the object surface is possible even in non-textured and unicoloured areas.

The deformation of the pattern enables the reconstruction of the surface even in low-textured and unicoloured areas if the relative position and orientation of camera and projector with respect to each other are known. It is possible to refine the reconstruction quality by sequential projections with varying pattern which is shown in an industrial application in chapter 10. A variety of approaches to encode pattern position and to refine realizations of this idea exist. Geng [140] provides an overview of different structured-light approaches for 3D surface reconstruction.

Other active systems to retrieve depth information from the scene include **time-of-flight cameras** and **LiDAR** sensors that use the known speed of light and measure the round trip time needed for an electromagnetic pulse of a specified wavelength in order to estimate the distance of objects. These systems are also used for autonomous driving and to evaluate stereo vision systems.[3]

6.1.2. Volumetric Approaches

In medical applications, image sensing systems such as **Magnetic Resonance Imaging (MRI)** and **computed tomography (CT)** provide the user with 3D volumetric data. For other sensing modalities such as **diagnostic sonography** with ultrasound, a tracking of position and orientation of the sensing transducer enables the possibility to reconstruct the imaged volume with spatial fusion of the acquisitions. This will be discussed in more detail in chapter 8.1.

In order to realize spatial alignments of observed 2D image slices and to reference system components as a backbone to properly describe the 3D pipelines hereafter, we discuss different mathematical concepts that describe rigid body motions in space as a next step.

6.2. Poses

A crucial element of 3D computer vision is the relative displacement between different system components of a pipeline. Thus, pose parametrization is ubiquitous in vision and robotics application as developed further throughout this thesis.
In this part, we take a close look on its mathematical description and discuss algebraic and practical properties of various forms to establish a strong backbone for applications such as pose estimation, tracking and co-calibration which follow afterwards.
The most widely used pose parametrizations involve rotation matrices, quaternions as well as twist-coordinates. We commence with a general introduction of poses and look at the commonly used matrix-vector representations.

6.2.1. Rigid Displacements

Definition 6.1
*A **rigid displacement** is a transformation*

$$P : \mathbb{R}^3 \to \mathbb{R}^3 \tag{6.1}$$

$$p \mapsto \boldsymbol{R}p + \boldsymbol{t} \tag{6.2}$$

with a translation vector $\boldsymbol{t} \in \mathbb{R}^3$ *and a rotation matrix* $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$ *where* $\boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}$ *and* $\det(\boldsymbol{R}) = 1$.

[3] Cf. Geiger, Lenz, and Urtasun [139], and Geiger et al. [138].

Fig. 6.3 illustrates this with an object coordinate system x-y-z which moves to x'-y'-z'. The displacement rotates and translates a set of points $p_i \in \mathbb{R}^3$ to its new locations $P(p_i) \in \mathbb{R}^3$. The position and orientation of the object is called its **pose**. A pose is characterized by six degrees of freedom: three for the translation in space and another three for the rotation about each spatial axis. This is why we sometimes also speak about **6D poses**.

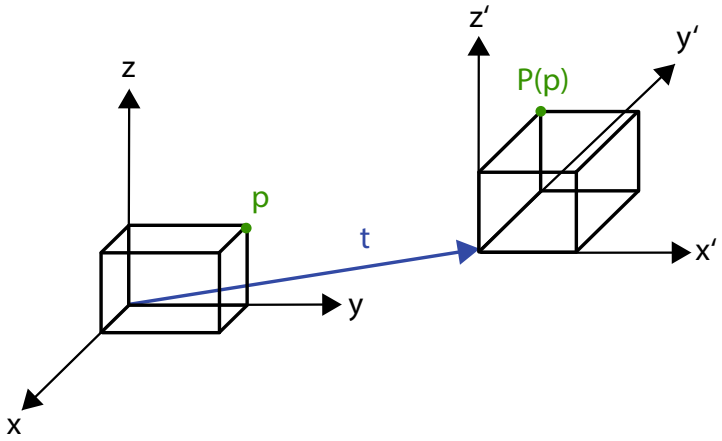


Fig. 6.3. **Rigid displacement.** The object is transformed with a rigid transformation P that involves the translation $\mathbf{t}$ illustrated by the blue vector and a rotation $\mathbf{R}$ that rotates the coordinate frame x-y-z to x'-y'-z'. The point p on the object is moved to $P(p)$.

Using homogeneous coordinates, the point

$$p = (x, y, z)^{\mathrm{T}} \tag{6.3}$$

$$p \mapsto (x, y, z, 1)^{\mathrm{T}} =: \mathbf{p} \tag{6.4}$$

can be transformed directly by

$$\mathbf{p} \mapsto \mathbf{P}\,\mathbf{p} \tag{6.5}$$

$$:= \begin{pmatrix} \boxed{\mathbf{R}} & \mathbf{t} \\ -0- & 1 \end{pmatrix} \mathbf{p} = \begin{pmatrix} \mathbf{R}p + \mathbf{t} \\ 1 \end{pmatrix}. \tag{6.6}$$

A rotation around an arbitrary axis can be decomposed into two translations and a rotation around the origin as shown in Fig. 6.4. We therefore focus on translations in Euclidean space $\mathbb{R}^3$ and rotations about the origin such that we treat rigid transformations in $\mathrm{SO}(3) \times \mathbb{R}^3$.

We discuss the parametrization of the elements of the 3D rotation group $\mathrm{SO}(3)$ in more detail.

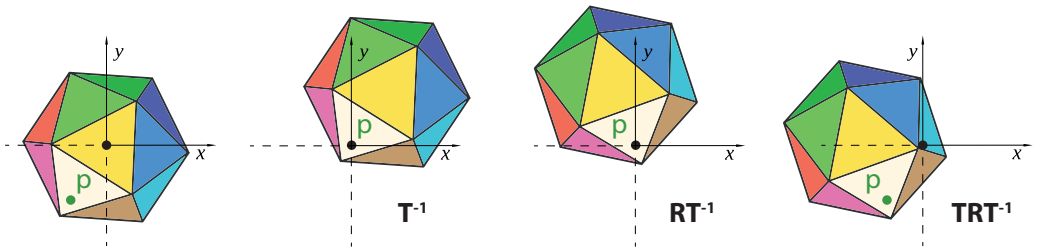


Fig. 6.4. **Rotation around arbitrary axis.** The object is rotated around the axis perpendicular to the image plane through the point p depicted in green. This can be done by consecutive execution of the three illustrated task. First, translate the local coordinates to the world reference by $\mathbf{T}^{-1}$, then rotate around the origin ($\mathbf{RT}^{-1}$). The sought rotation is given by translation of the result back ($\mathbf{TRT}^{-1}$).

6.2.2. Rotation Matrices & Euler Angles

A rotation about the origin in 2D with the angle φ can be described by

$$\mathbf{R}(\varphi) = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}. \tag{6.7}$$

Fig. 6.5 illustrates the rotation of the point $(r,0)^{\mathrm{T}}$ with the angle φ.

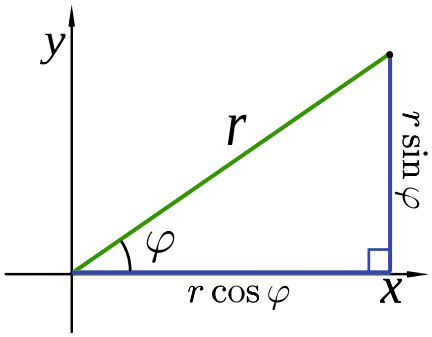


Fig. 6.5. **2D Rotation.** The vector $(r,0)^{\mathrm{T}}$ is rotated about the origin with the angle φ counterclockwise. The new position is illustrated with the green line. the new coordinates are given by $(r\cos\varphi, r\sin\varphi)^{\mathrm{T}}$ as shown in blue.

Applying equation (6.7) to the 3D planes with normals in x-, y-, and z-direction, we get

$$\mathbf{R}_{\mathrm{x}}(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}, \quad \mathbf{R}_{\mathrm{y}}(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}, \quad \mathbf{R}_{\mathrm{z}}(\gamma) = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

where the rotation angles α, β, and γ are known as yaw, pitch and roll. Altogether, we can write a representation for the 3D rotation as

$$\mathbf{R} = \mathbf{R}_z(\gamma)\,\mathbf{R}_y(\beta)\,\mathbf{R}_x(\alpha) \tag{6.8}$$

with three parameters and nine entries. This representation has the minimum of three degrees of freedom to describe a rotation in space. However, it is not unique.

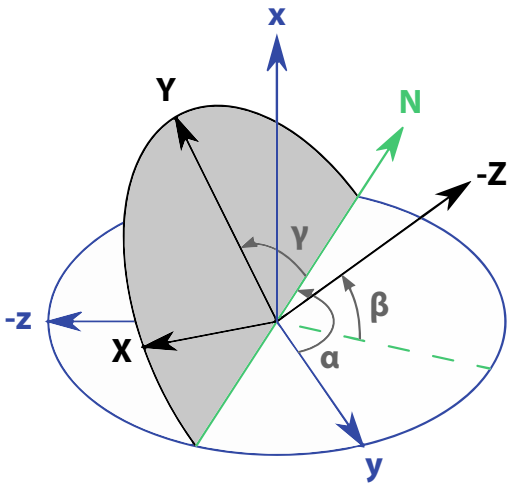


Fig. 6.6. **3D Rotation and Euler angles.** Illustrated is a rotation of the blue x-y-z reference frame about the origin with angles α, β, and γ into the black X-Y-Z coordinate frame. For visualization purposes, the negative z- and Z-axis are drawn. The rotations are indicated in grey and the line of nodes N (i.e. the XY-yz intersection and the y-axis after the first rotation) is highlighted in full green while the -z-axis is shown with a dotted green line before the second rotation.

Looking at the 3D scenario in Fig. 6.6, for example, reveals a problem if x-y and X-y planes become incident. It comes to the **Gimbal lock** phenomenon where – depending on whether z- and Z-axis point in the same or opposite direction – only $\alpha + \gamma$ or $\alpha - \gamma$ are uniquely defined and not the individual values α and γ. This becomes problematic in particular when an interpolation between certain rotations is needed. We study this problem more closely in chapter 9.1.
One way to circumvent this issue is the use of quaternions.

6.2.3. Quaternions

Another way to look at equation (6.7) is the use of a complex numbers $q \in \mathbb{C}$. While a matrix-vector multiplication with $\mathbf{R}(\varphi)$ rotates a vector $(a, b)^{\mathrm{T}}$ about the origin with the angle φ, the rotor $q = \cos\varphi + \mathbf{i} \sin\varphi$ rotates the complex number representation $p = a + \mathbf{i}\, b$ in the same way.
The goal of this section is to use an extension of the complex numbers, the **quaternions** $\mathbb{H}$ to perform a similar task in 3D and to define a rotation quaternion $\mathbf{q} \in \mathbb{H}$. We follow the terminology of Busam et al. [50] and Busam et al. [48].

6.2.3.1. Introduction to Quaternions

Hamilton [160] is the first to describes a non-commutative division algebra of hypercomplex numbers in his early work which is why we attribute the letter $\mathbb{H}$ to this concept. Besides their essential role in pure and applied geometry,[4] quaternions are frequently used in computer vision.[5]
The concatenation of rotations becomes more efficient with quaternions and singularities are avoided.[6] As an effect of this, quaternions are applied to various 3D processing pipelines that require robust real-time functionality.[7] The particular fact that elements of $\mathbb{H}$ can be identified with points on the 3-dimensional hypersphere S^3 is the source for their success in animation and rendering where Shoemake [374] proposes an efficient method for keyframe interpolation. The abbreviation of his method, **SLERP** stands for **S**pherical **L**in**E**ar inte**RP**olation and uses geodesic curves on S^3 to continuously blend from one quaternion to another.
The extension of the complex numbers with quaternions is realized in general by using three imaginary units **i**, **j**, **k**.

Definition 6.2
*A **quaternion q** as an element of the algebra $\mathbb{H}$ has the form*

$$\boldsymbol{q} = q_1\boldsymbol{1} + q_2\boldsymbol{i} + q_3\boldsymbol{j} + q_4\boldsymbol{k} \tag{6.9}$$

$$= \left(q_1, q_2, q_3, q_4\right)^T, \tag{6.10}$$

with $\left(q_1, q_2, q_3, q_4\right)^T \in \mathbb{R}^4$ *and*

$$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -\boldsymbol{1}. \tag{6.11}$$

Another way of writing a quaternion is $\mathbf{q} := [a, \mathbf{v}]$, where $\mathbf{v} = \left(q_2, q_3, q_4\right)^{\mathrm{T}} \in \mathbb{R}^3$ is called the vector part and $a = q_1 \in \mathbb{R}$ is the scalar part. The multiplication Table 6.1 for the imaginary units shows that quaternion composition is not commutative in general.
Similar to the complex numbers, a conjugation operator for the quaternion $\mathbf{q} \in \mathbb{H}$ is defined as

$$\bar{\mathbf{q}} := q_1 - q_2\mathbf{i} - q_3\mathbf{j} - q_4\mathbf{k}. \tag{6.12}$$

Especially the **unit quaternions** (or versors) $\mathbf{q} \in \mathbb{H}_1$ with

$$1 \stackrel{!}{=} \|\mathbf{q}\| := \mathbf{q} \cdot \bar{\mathbf{q}} \tag{6.13}$$

[4] E.g. Arnol'd [7] uses quaternions for geometrical purposes and Richter-Gebert and Orendt [348] apply them to different geometric problems.
[5] Cf. Pervin and Webb [329].
[6] The problem of Gimbal lock as explained in section 6.2.2 can be avoided. Cf. Lepetit and Fua [244].
[7] Cf. Mukundan [294].

$\cdot$	$\mathbf{1}$	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$
$\mathbf{1}$	$\mathbf{1}$	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$
$\mathbf{i}$	$\mathbf{i}$	$-\mathbf{1}$	$\mathbf{k}$	$-\mathbf{j}$
$\mathbf{j}$	$\mathbf{j}$	$-\mathbf{k}$	$-\mathbf{1}$	$\mathbf{i}$
$\mathbf{k}$	$\mathbf{k}$	$\mathbf{j}$	$-\mathbf{i}$	$-\mathbf{1}$

Tab. 6.1. **Multiplication table for the imaginary units of the quaternion algebra $\mathbb{H}$.** The missing symmetry in the table demonstrates that the multiplication is non-commutative. Note that extending the table along the diagonal reveals the real number $\mathbb{R}$ and the complex numbers $\mathbb{C}$ as part of the quaternions.

are of particular interest in computer vision due to their strong connection with spatial rotations[8] since they give a compact and numerically stable parametrization for orientation and rotation of objects in $\mathbb{R}^3$.

6.2.3.2. Rotations with Quaternions

Quaternions can be used to uniquely define spatial rotations. A rotation about the unit axis $\mathbf{v} = (v_1, v_2, v_3)^\mathrm{T} \in \mathbb{R}^3$ with angle θ is thereby given by the **rotor**

$$\mathbf{r} = [\cos(\theta/2), \sin(\theta/2)\mathbf{v}] \tag{6.14}$$

and antipodal points $\mathbf{q}$ and $-\mathbf{q} \in \mathbb{H}_1$ are identified with the same element in $\mathrm{SO}(3)$. Altogether, the unit quaternions form a double covering group of the 3D rotations about the origin and any **point quaternion** or pure quaternion

$$\mathbf{p} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}. \tag{6.15}$$

of a point $\mathbf{u} = (x, y, z)^\mathrm{T} \in \mathbb{R}^3$ can be rotated by the versor $\mathbf{r}$ via the sandwiching product map

$$\mathbf{p} \mapsto \mathbf{r} \cdot \mathbf{p} \cdot \bar{\mathbf{r}}. \tag{6.16}$$

The radius for the hypersphere $\mathbb{H}_1$ with $r = 1$ is arbitrary and done to simplify the notation. In fact, any other sphere in $\mathbb{R}^4$ with radius $r \neq 0$ would also work. The inverse rotation is given by

$$\mathbf{q}^{-1} := \frac{\bar{\mathbf{q}}}{\|\mathbf{q}\|^2} \tag{6.17}$$

such that $\mathbf{q}\mathbf{q}^{-1} = \mathbf{1}$ which simplifies to $\mathbf{q}^{-1} = \bar{\mathbf{q}}$ in $\mathbb{H}_1$. If we compare quaternions on the same line in $\mathbb{H}$ through the origin, we see that they describe the same rotation as

$$(\nu\mathbf{q}) \cdot \mathbf{p} \cdot (\nu\mathbf{q})^{-1} = \nu \cdot \mathbf{q} \cdot \mathbf{p} \cdot \mathbf{q}^{-1} \cdot \nu^{-1} \tag{6.18}$$

$$= \mathbf{q} \cdot \mathbf{p} \cdot \mathbf{q}^{-1} \cdot \nu\nu^{-1} \tag{6.19}$$

$$= \mathbf{q} \cdot \mathbf{p} \cdot \mathbf{q}^{-1} \tag{6.20}$$

[8] Cf. Faugeras [105].

holds true for an arbitrary quaternions $\mathbf{q} \in \mathbb{H} \setminus \{0\}$ and all $\nu \in \mathbb{R} \setminus \{0\}$.

6.2.4. Comparing Parametrizations

To compare different parametrizations for 6D poses in $\mathrm{SO}(3) \times \mathbb{R}^3$, we take a special look at rotations and summarize the findings in Table 6.2 for Euler angles, rotation matrices and quaternions.

Euler Angles	Rotation Matrices	Quaternions
✓ Easy for simple rot.	✓ Point transf. cheap	✓ Ref. system independent
✓ Most compact 3 DoF		✓ Isotropic (no order issues)
✗ Transf. via matrices	∘ Same form as other	✓ Concat. cheap (16×)
✗ Order dependent	linear transformations	✓ Smooth interpolation
✗ No direct composition		✓ No Gimbal Lock / Flipping
✗ Gimbal Lock / Flipping	✗ Concat. expensive (27×)	✓ Re-Normalization
✗ Difficult to predict	✗ Re-Orthogonalization	∘ Compact storage (4 coeff.)
✗ Interpr. counterintuitive	✗ Redundancy (9 coeff.)	✗ Difficult to visualize
✗ Ambiguities	✗ $\mathbb{R}^{3\times3} \supset \mathrm{SO}(3)$	✗ Rotations only

Tab. 6.2. **Comparison of different parametrizations for 3D rotations.** While the Euler angle representation has the most compact form it suffers from a series of drawbacks such as Gimbal Lock and the fact that the values cannot serve as transformation directly. Rotation matrices are well studied algebraic objects similar to other representations which come with the drawback of redundant representation that make an adjustment due to numerical inaccuracies with a re-orthogonalization very costly. Concatenating them requires 27 multiplications. Quaternions on the other side suffer from the lack of direct visualization of quaternionic curves while being very efficient and flexible in practice. The aspect of efficient interpolation can be a further advantage which we address in detail in chapter 9.1.

Further parametrizations for rotations such as Euler–Rodrigues parameters which are closely related to quaternions do also exist. The details would go beyond the scope of this thesis and we point the interested reader to Goldstein, Poole, and Safko [150].
This concludes our investigations on separate parametrization of rotation and translation and we research the concept of dual quaternions where these two entities are jointly fused.

6.2.5. Dual Quaternions

Rotations as presented so far, either use the group $\mathrm{SO}(3)$ of rotation matrices or the hypersphere $\mathbb{H}_1$ of quaternions. These representations are sufficient to handle orientation and the translation component is usually treated separately.[9] In this part, we treat translation and orientation jointly on the the dual quaternion quadric in 7-dimensional real projective space $\mathbb{RP}^7$ and

[9] Cf. Farenzena, Bartoli, and Mezouar [104] as well as Jia and Evans [195].

investigate the Riemannian manifold of unit **dual quaternions** $\mathbb{DH}_1$.
While the matrix representation of SE(3) with homogeneous matrices suffers from intrinsic singularities of this representation, dual quaternions can be a solution. Using a parametrization with dual quaternions (DQ), we can form a common space for the entire 6D pose that jointly describes both components the rotation and the translation.

6.2.5.1. Introduction to Dual Quaternions

Kavan et al. [207] already use a dual quaternion formulation successfully for interpolation and Xu et al. [457] apply the idea to rigid body dynamics. However, the applications of dual quaternions are not numerous which is partially due to the more complex underlying manifold that lacks immediate geometric intuition in contrast to non-dual quaternions.
The non-commutative division algebra of hypercomplex numbers $\mathbb{H}$ invented by Hamilton [160] are extended by Clifford [71] to the **Clifford algebra** of dual quaternions. In spite of the usefulness of dual approaches for many real-time tasks, they do not receive the same attention as their non-dual counterpart.
A rigid body displacement can be fully described by a dual quaternion $\mathbf{Q} \in \mathbb{DH}_1$ of unit length. Similar to the quaternion rotation from section 6.2.3, we use unit length dual quaternions to represent spatial displacements. For this, we define an ordered pair of quaternions with dual number coefficients as a dual quaternion. Following the terminology of Kenwrigth [215], we can write a **dual number** Z as an element of the algebra $\mathbb{D}$ in the form

$$Z = r + \varepsilon s, \tag{6.21}$$

where $r, s \in \mathbb{R}$ and $\varepsilon^2 = 0$.[10] The term $\varepsilon \neq 0$ is called the dual operator, r is the real-part, and s is the dual part. A dual conjugate similar to the complex conjugate in $\mathbb{C} = \mathbb{R} + i\mathbb{R}$ is defined as

$$\hat{Z} := r - \varepsilon s. \tag{6.22}$$

If we extend this concept to dual quaternions, we can write

Definition 6.3
*A **dual quaternion** $\boldsymbol{Q} \in \mathbb{DH}$ is an ordered set of quaternions $\boldsymbol{r}, \boldsymbol{s} \in \mathbb{H}$ with*

$$\boldsymbol{Q} = \boldsymbol{r} + \varepsilon \boldsymbol{s} = \left(q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8\right)^T, \tag{6.23}$$

where $\left(q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8\right)^T \in \mathbb{R}^8$ and $\varepsilon \neq 0$ with

$$\varepsilon^2 = 0, \quad \varepsilon \boldsymbol{i} = \boldsymbol{i}\varepsilon, \quad \varepsilon \boldsymbol{j} = \boldsymbol{j}\varepsilon, \quad \varepsilon \boldsymbol{k} = \boldsymbol{k}\varepsilon. \tag{6.24}$$

Again, this leads to non-commutative multiplications as shown in Table 6.3 where we notice that the Clifford algebra of dual quaternions contains the real numbers $\mathbb{R}$, the complex numbers $\mathbb{C}$, the dual numbers $\mathbb{D}$, and the quaternions $\mathbb{H}$.

[10] Cf. Ercan and Yüce [100].

$\cdot$	$\mathbf{1}$	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$	ε	$\varepsilon\mathbf{i}$	$\varepsilon\mathbf{j}$	$\varepsilon\mathbf{k}$
$\mathbf{1}$	$\mathbf{1}$	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$	ε	$\varepsilon\mathbf{i}$	$\varepsilon\mathbf{j}$	$\varepsilon\mathbf{k}$
$\mathbf{i}$	$\mathbf{i}$	$-\mathbf{1}$	$\mathbf{k}$	$-\mathbf{j}$	$\varepsilon\mathbf{i}$	$-\varepsilon$	$\varepsilon\mathbf{k}$	$-\varepsilon\mathbf{j}$
$\mathbf{j}$	$\mathbf{j}$	$-\mathbf{k}$	$-\mathbf{1}$	$\mathbf{i}$	$\varepsilon\mathbf{j}$	$-\varepsilon\mathbf{k}$	$-\varepsilon$	$\varepsilon\mathbf{i}$
$\mathbf{k}$	$\mathbf{k}$	$\mathbf{j}$	$-\mathbf{i}$	$-\mathbf{1}$	$\varepsilon\mathbf{k}$	$\varepsilon\mathbf{j}$	$-\varepsilon\mathbf{i}$	$-\varepsilon$
ε	ε	$\varepsilon\mathbf{i}$	$\varepsilon\mathbf{j}$	$\varepsilon\mathbf{k}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$
$\varepsilon\mathbf{i}$	$\varepsilon\mathbf{i}$	$-\varepsilon$	$\varepsilon\mathbf{k}$	$-\varepsilon\mathbf{j}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$
$\varepsilon\mathbf{j}$	$\varepsilon\mathbf{j}$	$-\varepsilon\mathbf{k}$	$-\varepsilon$	$\varepsilon\mathbf{i}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$
$\varepsilon\mathbf{k}$	$\varepsilon\mathbf{k}$	$\varepsilon\mathbf{j}$	$-\varepsilon\mathbf{i}$	$-\varepsilon$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$

Tab. 6.3. **Multiplication table for the imaginary units of the dual quaternion algebra $\mathbb{DH}$.** The missing symmetry in the table demonstrates that the multiplication is non-commutative. Note that extending the table along the diagonal reveals the real number $\mathbb{R}$, the complex numbers $\mathbb{C}$, and the quaternions $\mathbb{H}$ as part of the dual quaternions.

Defining the conjugate $\bar{\mathbf{Q}}$ of the dual quaternion $\mathbf{Q} = \mathbf{r} + \varepsilon\mathbf{s}$ as

$$\bar{\mathbf{Q}} := \bar{\mathbf{r}} + \varepsilon\bar{\mathbf{s}}, \tag{6.25}$$

we can investigate the constraints given for a **unit dual quaternion** $\mathbf{Q} \in \mathbb{DH}_1$ of length 1. Restricting the dual quaternion $\mathbf{Q}$ to unit length gives

$$\begin{aligned} 1 \stackrel{!}{=} \|\mathbf{Q}\| &:= \mathbf{Q} \cdot \bar{\mathbf{Q}} && (6.26)\\ &= (\mathbf{r} + \varepsilon\mathbf{s}) \cdot (\bar{\mathbf{r}} + \varepsilon\bar{\mathbf{s}}) && (6.27)\\ &= \mathbf{r}\bar{\mathbf{r}} + \varepsilon(\mathbf{r}\bar{\mathbf{s}} + \mathbf{s}\bar{\mathbf{r}}), && (6.28) \end{aligned}$$

which decomposes into the two distinct constraints

$$\begin{aligned} \mathbf{r}\bar{\mathbf{r}} &= 1 \quad \text{and} && (6.29)\\ \mathbf{r}\bar{\mathbf{s}} + \mathbf{s}\bar{\mathbf{r}} &= 0. && (6.30) \end{aligned}$$

6.2.5.2. Displacements with Dual Quaternions

The group of rigid body displacement $\mathrm{SE}(3)$ and the unit dual quaternions are isomorphic[11] and the two constraints from equations (6.29) and (6.30) reduce the eight parameters of a dual quaternion to the six degrees of freedom for a rigid motion in space. Let us construct a unit dual quaternion by writing the translation as a point quaternion $\mathbf{t}$ (cf. (6.15)) and the rotation $\mathbf{r}$ as a unit quaternion (cf. (6.14)) such that

$$\mathbb{DH}_1 \ni \mathbf{Q} = \mathbf{r} + \varepsilon\frac{1}{2}\mathbf{t}\mathbf{r}. \tag{6.31}$$

[11] Cf. Ablamowicz and Sobczyk [1].

Analogously to the rotor in non-dual quaternions, we name $\mathbf{Q} \in \mathbb{DH}_1$ a **displacor** which spatially moves a **dual point quaternion** $\mathbf{P} = \mathbf{1} + \varepsilon\mathbf{u}$ with the point quaternion $\mathbf{u} = [0, \mathbf{p}]$ via a sandwiching product map.
Direct calculation shows that the spatial displacement in terms of dual point quaternions becomes

$$\begin{aligned}
\mathbf{P} \mapsto & \mathbf{Q} \cdot \mathbf{P} \cdot \hat{\bar{\mathbf{Q}}} && (6.32)\\
&= \left(\mathbf{r} + \frac{\varepsilon}{2}\mathbf{t}\mathbf{r}\right)(\mathbf{1} + \varepsilon\mathbf{u})\left(\bar{\mathbf{r}} - \frac{\varepsilon}{2}\bar{\mathbf{t}}\bar{\mathbf{r}}\right) && (6.33)\\
&= \left(\mathbf{r} + \frac{\varepsilon}{2}\mathbf{t}\mathbf{r} + \varepsilon\mathbf{r}\mathbf{u}\right)\left(\bar{\mathbf{r}} - \frac{\varepsilon}{2}\bar{\mathbf{r}}\bar{\mathbf{t}}\right) && (6.34)\\
&= \mathbf{r}\bar{\mathbf{r}} + \varepsilon\left(\frac{1}{2}\mathbf{t}\mathbf{r}\bar{\mathbf{r}} + \mathbf{r}\mathbf{u}\bar{\mathbf{r}} - \frac{1}{2}\mathbf{r}\bar{\mathbf{r}}\bar{\mathbf{t}}\right) && (6.35)\\
&= 1 + \varepsilon\left(\mathbf{r}\mathbf{u}\bar{\mathbf{r}} + \mathbf{t}\right), && (6.36)
\end{aligned}$$

where the two conjugates for the dual quaternion and the dual are calculated consecutively. The last step holds since $\bar{\mathbf{t}} = -\mathbf{t}$ for the point quaternion $\mathbf{t}$ and the term $\mathbf{r}\mathbf{u}\bar{\mathbf{r}}$ matches exactly the known quaternion rotation from equation (6.16). As a result, the last line (6.36) represents a spatial displacement in dual quaternion notation. In compliance with quaternions, a recalculation with the displacor $-\mathbf{Q}$ gives the same result.

6.2.6. Riemannian Geometry

The spaces $\mathbb{H}_1$ and $\mathbb{DH}_1$ can also be studied from the perspective of differential geometry. In fact, both the unit quaternion and unit dual quaternion space are non-Euclidean and form differentiable **Riemannian manifolds**.[12] We analyze these spaces locally and calculate exponential and logarithm maps explicitly in quaternion representation to enable implementation of algorithms acting directly on the manifold afterwards.
For a Riemannian manifold $\mathbb{G}$, a continuous collection of inner products on the tangent space of $\mathbb{G}$ at $\mathbf{x} \in \mathbb{G}$ defines a Riemannian metric. The shortest path defined by such a metric on the manifold is called the **geodesic**. We analyze the geometric structure of (dual) quaternion space and calculate mappings into the tangent space and back using parallel transport. This paves the way also for the pose interpolation discussed in chapter 9.1 and pose filtering methods with local geodesic regressors discussed in part 9.2.

6.2.6.1. Geometry of $\mathbb{H}_1$ and $\mathbb{DH}_1$

The unit quaternions are constrained by equation (6.13) and cover the three dimensional hypersphere $S^3 \in \mathbb{R}^4$. Thus, $\mathbb{H}_1$ and the real projective space $\mathbb{RP}^3$ are isomorphic.
Similarly, the two constraints (6.29) and (6.30) help to analyze the structure of unit dual quaternion space. Looking at the first equation $\|\mathbf{r}\| = 1$, we observe that the real part $\mathbf{r}$ of $\mathbf{Q}$ is forced to be of unit length, hence $\mathbf{r} \in \mathbb{H}_1$. This constrains the projection to the first four parameters to form a hypersphere. Thinking of all parameters in homogeneous coordinates

[12] Cf. Tron, Vidal, and Terzis [420].

and identifying equivalence classes again, we can investigate the unit dual quaternion space on the 7-dimensional hypersphere $S^7 \in \mathbb{R}^8$. The identification of antipodal points on the hypersphere forms the seven dimensional real projective space $\mathbb{RP}^7$. A closer look at the second constraint $\mathbf{r}\bar{\mathbf{s}} = -\mathbf{s}\bar{\mathbf{r}}$ reveals that this is a definition of a 6-dimensional quadric in $\mathbb{RP}^7$. Thus, equation (6.29) is redundant and the space constrained even more by equation (6.30). As a result, $\mathbb{DH}_1$ is not a hypersphere, but a quadric[13] which needs to be considered for any operation on the manifold.

6.2.6.2. Lie Groups and Parallel Transport

Differentiable manifolds which are also groups with smooth operations are called **Lie groups**. This is the case for both non-dual and dual quaternions and we take a closer look at some aspects from this differential geometry perspective. Introduced to study infinitesimal transformations,[14] the tangent space at the identity of the group is commonly described as the **Lie algebra** to the Lie group. It gives a local linearization to the Lie group near the identity. The map from the tangent space $T_{\mathbf{x}}\mathbb{G}$ at $\mathbf{x}$ to the Lie group $\mathbb{G}$ is called the exponential map

$$\exp_{\mathbf{x}} : T_{\mathbf{x}}\mathbb{G} \to \mathbb{G}. \tag{6.37}$$

It is locally defined and maps a vector in the tangent space to a point on the manifold. The mapping follows the geodesic on $\mathbb{G}$ through $\mathbf{x}$. The inverse to the exponential map is called the logarithm map

$$\log_{\mathbf{x}} : \mathbb{G} \to T_{\mathbf{x}}\mathbb{G}. \tag{6.38}$$

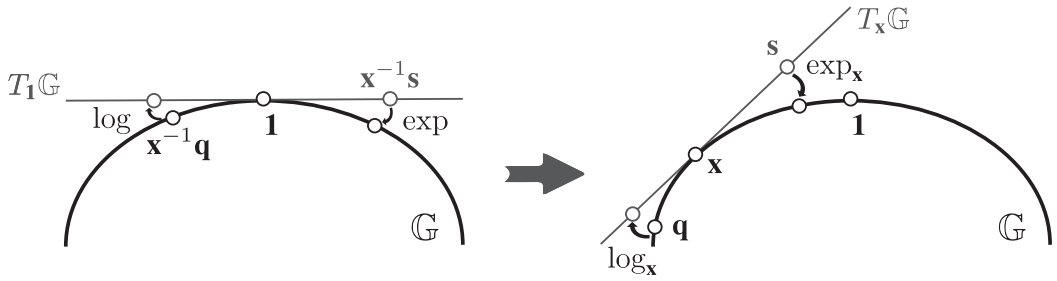


Fig. 6.7. **Parallel transport for the calculation of Lie operators $\exp_{\mathbf{x}}$ and $\log_{\mathbf{x}}$.** The intermediate calculations of $\mathbf{x}^{-1}\mathbf{s}$ and $\mathbf{x}^{-1}\mathbf{q}$ are used to apply the operators to the Lie algebra $T_1\mathbb{G}$ (left) at the identity of the Lie group $\mathbb{G}$. The information is transported along the manifold to point $\mathbf{x} \in \mathbb{G}$ (right).

Explicit formulas for the general mappings $\exp_{\mathbf{x}}$ and $\log_{\mathbf{x}}$ may be cumbersome to derive. Thus, the underlying Lie group is often studied through an investigation of the according Lie algebra and by utilizing parallel transport. We follow the same approach and derive numerically stable Lie algebras for the unit and the dual quaternions. Parallel transport as illustrated in Fig. 6.7

[13] The quadric is called Study quadric as detailed in Study [397].
[14] Cf. O'Connor and Robertson [311].

can be used to deduce the general case to calculate $\exp_{\mathbf{x}}$ at point $\mathbf{x} \in \mathbb{G}$.[15] Defining $\log := \log_{\mathbf{1}}$ and $\exp := \exp_{\mathbf{1}}$ for the logarithm and exponential maps at the identity $\mathbf{1} \in \mathbb{G}$ it holds

$$\exp_{\mathbf{x}}(\mathbf{s}) = \mathbf{x} \exp\left(\mathbf{x}^{-1}\mathbf{s}\right), \tag{6.39}$$

$$\log_{\mathbf{x}}(\mathbf{q}) = \mathbf{x} \log\left(\mathbf{x}^{-1}\mathbf{q}\right). \tag{6.40}$$

We now look at the groups $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$ and deduce formulae for these maps in quaternion notation. For matrices Murray et al. [300] and Visser et al. [436] are one of several sources that study these maps. As shown in section 6.2.4, there are several advantages if quaternions are used, such as their small memory footprint and efficiency for consecutive transformations. Moreover, numerical stability is advantageous compared to the higher dimensional matrix space where a re-orthogonalization is more expensive to calculate than normalizing the quaternion vector. These are advantages that are of particular interest in regression tasks with neural networks for instance. We study the exponential maps therefore directly by their Maclaurin series definition in (dual) quaternion space.

6.2.6.3. Exponential and Logarithm Map in $\mathbb{H}$

The quaternion vector $\mathbf{1} = (1,0,0,0)^{\mathrm{T}}$ defines the identity in $\mathbb{H}_1$. Its tangent space $T_{\mathbf{1}}\mathbb{H}_1$ is the hyperplane parallel to the plane defined by x_2, x_3, x_4 axes that touches the hypersphere $S^3 \in \mathbb{R}^4$ in $\mathbf{1}$. Thus, any element in $T_{\mathbf{1}}\mathbb{H}_1$ can be written in the form $\mathbf{q} \in \mathbb{H}$ with

$$\mathbf{q} = [0, \phi\mathbf{v}] \tag{6.41}$$

with the unit vector $\mathbf{v} \in \mathbb{R}^3$, $\|\mathbf{v}\| = 1$ and $\phi \in \mathbb{R}$. Thus it holds

$$\mathbf{q}^k = \begin{cases} (-1)^{\frac{k}{2}}\ \phi^k & \forall k \in \{0,2,4,\ldots\} \\ (-1)^{\frac{k-1}{2}}\ \phi^k \mathbf{q} & \forall k \in \{1,3,5,\ldots\}. \end{cases} \tag{6.42}$$

Writing the series expansion for the exponential map gives

$$\exp : T_{\mathbf{1}}\mathbb{H}_1 \to \mathbb{H}_1 \tag{6.43}$$

$$\mathbf{q} \mapsto \exp(\mathbf{q}) := \sum_{k=0}^{\infty} \frac{\mathbf{q}^k}{k!} \tag{6.44}$$

$$\overset{(6.42)}{=} \left(1 - \frac{\phi^2}{2!} + \frac{\phi^4}{4!} - \ldots\right) + \left(\phi\mathbf{q} - \frac{\phi^3\mathbf{q}}{3!} + \frac{\phi^5\mathbf{q}}{5!} - \ldots\right) \tag{6.45}$$

$$= \cos(\phi) + \frac{\sin(\phi)}{\phi}\mathbf{q} \tag{6.46}$$

$$= [\cos(\phi), \sin(\phi)\mathbf{v}] \tag{6.47}$$

$$=: \mathbf{r}. \tag{6.48}$$

[15] Cf. Gallier [133].

Where (6.46) recognizes the Taylor series of the trigonometric functions about 0. This relationship aligns with the notation discussed in (6.14) with $\phi = \theta/2$. The inverse reads as

$$\log : \mathbb{H}_1 \to T_{\mathbf{1}}\mathbb{H}_1 \tag{6.49}$$

$$\mathbf{r} \mapsto [0, \phi\mathbf{v}]. \tag{6.50}$$

6.2.6.4. Exponential and Logarithm Map in $\mathbb{DH}$

For the dual quaternion mappings, let $\mathbf{Q}$ be a pure dual quaternion in $\mathbb{DH}$ with

$$\mathbf{Q} = \omega\mathbf{q} + \varepsilon\psi\mathbf{q}_\varepsilon. \tag{6.51}$$

where $\mathbf{q}, \mathbf{q}_\varepsilon \in \mathbb{H}_1$ are two pure quaternions. One can simplify[16] the Maclaurin series for the exponential operator such that

$$\exp : T_{\mathbf{1}}\mathbb{DH}_1 \to \mathbb{DH}_1 \tag{6.52}$$

$$\mathbf{Q} \mapsto \sum_{k=0}^{\infty} \frac{\mathbf{Q}^k}{k!} \tag{6.53}$$

$$= \frac{1}{2}(2\cos(\omega) + \omega\sin(\omega)) \tag{6.54}$$

$$- \frac{1}{2\omega}(\omega\cos(\omega) - 3\sin(\omega))\mathbf{Q} \tag{6.55}$$

$$+ \frac{1}{2\omega}(\sin(\omega))\mathbf{Q}^2 \tag{6.56}$$

$$- \frac{1}{2\omega^3}(\omega\cos(\omega) - \sin(\omega))\mathbf{Q}^3. \tag{6.57}$$

In order to formulate the inverse function we take a closer look at the unit dual quaternion

$$\mathbf{Q} = [\phi, \mathbf{v}] + \varepsilon[\phi_\varepsilon, \mathbf{v}_\varepsilon] \tag{6.58}$$

$$:= [\Phi, \mathbf{V}] \tag{6.59}$$

with the dual entities

$$\Phi = \phi + \varepsilon\phi_\varepsilon \tag{6.60}$$

$$\mathbf{V} = \mathbf{v} + \varepsilon\mathbf{v}_\varepsilon. \tag{6.61}$$

Equivalently to (6.14), we can find a canonical forms for dual quaternions.[17] A series expansion for the trigonometric operators gives the dual trigonometric operators

$$\sin(\Phi) := \sin(\phi) + \varepsilon\phi_\varepsilon\cos(\phi), \tag{6.62}$$

$$\cos(\Phi) := \cos(\phi) - \varepsilon\phi_\varepsilon\sin(\phi). \tag{6.63}$$

The following lemma justifies the use of a canonical form. We explore this through explicit calculation of the dual representation.

[16] Cf. Selig [373].
[17] Cf. Daniilidis [80].

Lemma 6.1
Any unit dual quaternion $\boldsymbol{Q} \in \mathbb{DH}_1$ *can be written as*

$$\boldsymbol{Q} = \left[\cos\left(\frac{\Theta}{2}\right), \sin\left(\frac{\Theta}{2}\right) \boldsymbol{V}\right], \tag{6.64}$$

where $\boldsymbol{V} \in \mathbb{DH}$ *is a pure dual quaternion of form* (6.51).

Proof. Equation (6.64) can be seen as a parametrization of a rigid body motion. Chasles' Theorem[18] tells that any rigid displacement in space can be decomposed into a translation along a unique axis together with a rotation about this axis as visualized in Fig. 6.8. We explicitly construct the rigid transformation given by the dual quaternion for such a motion in form of (6.64).

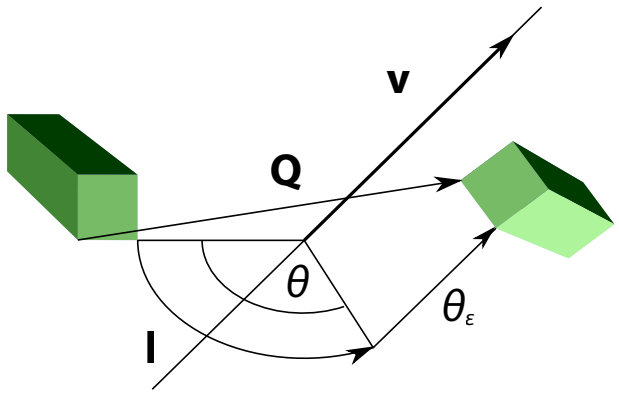


Fig. 6.8. **Screw linear displacement.** The rigid transformation described by the dual quaternion $\mathbf{Q}$ is decomposed into a rotation with angle θ about the axis $\mathbf{l}$ and a translation of length θ_ε in the direction of $\mathbf{v}$.

Let a rigid displacement be given by a rotation $\mathbf{R} \in \mathbb{R}^{3\times3}$ around the unit vector $\mathbf{v}$ with $\|\mathbf{v}\| = 1$ and rotation angle θ together with a translation $\mathbf{t} \in \mathbb{R}^3$. The dual quaternion parametrizing this motion is given by (6.31).
We now calculate the displacement parameters for the screw motion: These are the rotation angle θ, the screw axis $\mathbf{l}$ with a direction $\mathbf{v}$ and moment $\mathbf{v}_\varepsilon$ (i.e. $\mathbf{v}_\varepsilon = \mathbf{p} \times \mathbf{v} \;\forall\; \mathbf{p} \in \mathbf{l}$) as well as the pitch θ_ε. As the angle θ and the signed axis $\mathbf{v}$ are already given, we first calculate the pitch θ_ε as the projection of the translation onto the axis in the direction of $\mathbf{v}$:

$$\theta_\varepsilon = \mathbf{t}^{\mathrm{T}}\mathbf{v}. \tag{6.65}$$

To determine the moment $\mathbf{v}_\varepsilon$, we first pick a point $\mathbf{u}$ on the axis and express the translation $\mathbf{t}$ with the parameters θ_ε, $\mathbf{v}$, $\mathbf{R}$ and $\mathbf{u}$ as the sum of the part $\mathbf{t}_\parallel$ parallel to the axis and the perpendicular part $\mathbf{t}_\perp$ by

$$\mathbf{t} = \mathbf{t}_\parallel + \mathbf{t}_\perp \tag{6.66}$$
$$= \theta_\varepsilon \mathbf{v} + (\mathbf{I} - \mathbf{R})\,\mathbf{u}. \tag{6.67}$$

[18] Cf. Chen [66].

Rodrigues formula gives

$$\mathbf{Ru} = \mathbf{u} + \sin(\theta)\,\mathbf{v}\times\mathbf{u} + (1-\cos(\theta))\,\mathbf{v}\times(\mathbf{v}\times\mathbf{u}) \tag{6.68}$$

which we can substitute into (6.67) such that

$$\begin{aligned}
\mathbf{u} &\overset{(6.67)}{=} \mathbf{t} - \theta_\varepsilon \mathbf{v} + \mathbf{Ru} && (6.69)\\
&\overset{(6.65)}{=} \mathbf{t} - \left(\mathbf{t}^{\mathrm{T}}\mathbf{v}\right)\mathbf{v} + \mathbf{Ru} && (6.70)\\
&\overset{(6.68)}{=} \mathbf{t} - \left(\mathbf{t}^{\mathrm{T}}\mathbf{v}\right)\mathbf{v} + \mathbf{u} + \sin(\theta)\,\mathbf{v}\times\mathbf{u} + (1-\cos(\theta))\,\mathbf{v}\times(\mathbf{v}\times\mathbf{u}). && (6.71)
\end{aligned}$$

With $\mathbf{u}^{\mathrm{T}}\mathbf{v} = 0$, this gives

$$\mathbf{u} = \frac{1}{2}\left(\mathbf{t} - \left(\mathbf{t}^{\mathrm{T}}\mathbf{v}\right)\mathbf{v} + \cot\left(\frac{\theta}{2}\right)\mathbf{v}\times\mathbf{t}\right) \tag{6.72}$$

and we can write the moment vector as

$$\begin{aligned}
\mathbf{v}_\varepsilon &= \mathbf{u}\times\mathbf{v} && (6.73)\\
&= \frac{1}{2}\left(\mathbf{t}\times\mathbf{v} + \cot\left(\frac{\theta}{2}\right)\mathbf{v}\times(\mathbf{t}\times\mathbf{v})\right). && (6.74)
\end{aligned}$$

The rotation quaternion $\mathbf{r} = [q_0, \mathbf{q}]$ from $\mathbf{R}$ reads with (6.14) as $\mathbf{r} = [\cos(\theta/2), \sin(\theta/2)\mathbf{v}]$ and (6.74) becomes

$$\sin\left(\frac{\theta}{2}\right)\mathbf{v}_\varepsilon = \frac{1}{2}\left(\mathbf{t}\times\mathbf{q} + q_0\mathbf{t} - \cos\left(\frac{\theta}{2}\right)\left(\mathbf{v}^{\mathrm{T}}\mathbf{t}\right)\mathbf{v}\right). \tag{6.75}$$

Using (6.65), we can write

$$\sin\left(\frac{\theta}{2}\right)\mathbf{v}_\varepsilon + \frac{\theta_\varepsilon}{2}\cos\left(\frac{\theta}{2}\right)\mathbf{v} = \frac{1}{2}\left(\mathbf{t}\times\mathbf{q} + q_0\mathbf{t}\right). \tag{6.76}$$

This is precisely the pure quaternion of the dual part in (6.31). Thus we can write the dual quaternion representation of the rigid displacement as

$$\begin{aligned}
\mathbf{Q} &= [q_0, \mathbf{q}] + \varepsilon\left[-\frac{1}{2}\mathbf{q}^{\mathrm{T}}\mathbf{t}, \frac{1}{2}\left(q_0\mathbf{t} + \mathbf{t}\times\mathbf{q}\right)\right] && (6.77)\\
&\overset{(6.76)}{=} \left[\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\mathbf{v}\right] + \varepsilon\left[-\frac{\theta_\varepsilon}{2}\sin\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\mathbf{v}_\varepsilon + \frac{\theta_\varepsilon}{2}\cos\left(\frac{\theta}{2}\right)\mathbf{v}\right] && (6.78)\\
&\overset{(6.63)}{=} \cos\left(\frac{\Theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)\mathbf{v} + \varepsilon\sin\left(\frac{\theta}{2}\right)\mathbf{v}_\varepsilon + \varepsilon\frac{\theta_\varepsilon}{2}\cos\left(\frac{\theta}{2}\right)\mathbf{v} && (6.79)\\
&= \cos\left(\frac{\Theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)\mathbf{v} + \varepsilon\sin\left(\frac{\theta}{2}\right)\mathbf{v}_\varepsilon + \varepsilon\frac{\theta_\varepsilon}{2}\cos\left(\frac{\theta}{2}\right)\mathbf{v} + \underbrace{\varepsilon^2\frac{\theta_\varepsilon}{2}\cos\left(\frac{\theta}{2}\right)\mathbf{v}_\varepsilon}_{=0} && (6.80)\\
&= \cos\left(\frac{\Theta}{2}\right) + \left(\sin\left(\frac{\theta}{2}\right) + \varepsilon\frac{\theta_\varepsilon}{2}\cos\left(\frac{\theta}{2}\right)\right)(\mathbf{v} + \varepsilon\mathbf{v}_\varepsilon) && (6.81)\\
&\overset{(6.62)}{=} \cos\left(\frac{\Theta}{2}\right) + \sin\left(\frac{\Theta}{2}\right)(\mathbf{v} + \varepsilon\mathbf{v}_\varepsilon) && (6.82)\\
&\overset{(6.61)}{=} \left[\cos\left(\frac{\Theta}{2}\right), \sin\left(\frac{\Theta}{2}\right)\mathbf{V}\right] && (6.83)
\end{aligned}$$

□

This representation algebraically separates the pitch and angle values from the line information of the screw axis. The dual angle Θ encapsulates the information for both the rotation angle and the translation length while the dual vector $\mathbf{V}$ contains the axis of the screw motion and its direction vector.
The exponential map of a dual quaternion in the form $\mathbf{Q} = \mathbf{V}\frac{\Theta}{2}$ is given by[19]

$$\exp\left(\mathbf{V}\frac{\Theta}{2}\right) = \left[\cos\left(\frac{\Theta}{2}\right), \sin\left(\frac{\Theta}{2}\right)\mathbf{V}\right]. \tag{6.84}$$

The inverse function of a dual quaternion in canonical representation (6.64) thus can be calculated by

$$\log : \mathbb{DH}_1 \to T_{\mathbf{1}}\mathbb{DH}_1 \tag{6.85}$$

$$\left[\cos\left(\frac{\Theta}{2}\right), \sin\left(\frac{\Theta}{2}\right)\mathbf{V}\right] \mapsto \mathbf{V}\frac{\Theta}{2}. \tag{6.86}$$

Equipped with the tools on how to parametrize poses in space, we now investigate the diverse applications where these can be used starting with the estimation of the relative poses of cameras and tools with respect to each other in order to passively sense the surrounding geometry and actively interact with the recognized 3D world by automatic steering of robotic manipulators. To make these applications possible, we first take a closer look at the internal geometry of multi-camera systems.

6.3. Epipolar Geometry

How do humans visually sense the world? We use two similar organs of vision separated only by some base vector $\mathbf{b}$ as illustrated figuratively in figure 6.9, where the disparities between the rays code the depth information processed by our brain.
Let us get back again to the introductory question of this chapter: How can we make a computer see in three dimensions? We answer this in the supposedly most intuitive and natural way by describing this scenario in mathematical terms, which allow us to triangulate a point observed in two cameras of one **stereo camera system**. Hereby, we follow the explanations structure of Busam [47].

6.3.1. Geometric Analysis

Suppose we have two already calibrated cameras C_L and C_R left and right with their projection centres given by O_L and O_R as shown in Fig. 6.10. We call the vector $\mathbf{b}$ from O_L to O_R the base and the line through the projection centres the baseline.[20]
To formulate some ideas in an idealized manner and to directly obtain a simpler notation,

[19] Cf. Kavan et al. [206].
[20] Cf. Steger, Ulrich, and Wiedemann [395, p. 199].

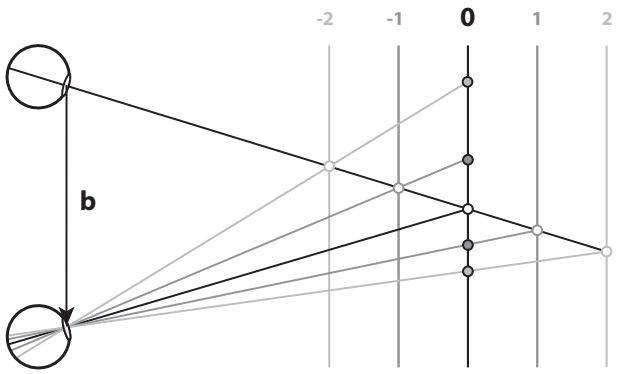


Fig. 6.9. **Binocular disparity.** The human visual system consists of a pair of eyes (top and bottom left) separated by a baseline vector **b**. The distance from the observer is decoded by the projection onto the retina. For a given ray of sight from the upper (left) eye, several depth layers are illustrated relative to a mid depth layer ("0"). Closer distance (e.g. $-1, -2$) gradually shift the projection further away from the left eye while further distances (e.g. $1, 2$) result in closer projections.

we now introduce a fictive **normalized coordinate system**[21] for the two cameras with the projections $\mathbf{P}_L$ and $\mathbf{P}_R$ as in equation (4.25) with

$$\mathbf{x}_L = \mathbf{P}_L \, \mathbf{x}_W = \mathbf{K}_L \Big(\mathbf{R}_L \mid \mathbf{t}_L \Big) \mathbf{x}_W \tag{6.87}$$

and

$$\mathbf{x}_R = \mathbf{P}_R \, \mathbf{x}_W = \mathbf{K}_R \Big(\mathbf{R}_R \mid \mathbf{t}_R \Big) \mathbf{x}_W . \tag{6.88}$$

If we apply $\mathbf{K}_L^{-1}$ to the point $\mathbf{x}_L$ we get the normalized coordinate

$$\hat{\mathbf{x}}_L = \mathbf{K}_L^{-1} \mathbf{K}_L \Big(\mathbf{R}_L \mid \mathbf{t}_L \Big) \mathbf{x}_W \tag{6.89}$$

$$= \Big(\mathbf{R}_L \mid \mathbf{t}_L \Big) \mathbf{x}_W , \tag{6.90}$$

which we can think of as an image of the world point $\mathbf{x}_W$ with respect to a camera at $\Big(\mathbf{R}_L \mid \mathbf{t}_L \Big)$ and an identity calibration matrix. Similarly we have

$$\hat{\mathbf{x}}_R = \Big(\mathbf{R}_R \mid \mathbf{t}_R \Big) \mathbf{x}_W \tag{6.91}$$

for the normalized coordinates $\hat{\mathbf{x}}_R$ within the right image. Hence we can always consider a pair of cameras with projections

$$\mathbf{P}_L = \Big(\mathbf{I} \mid \mathbf{0} \Big) \quad \text{and} \quad \mathbf{P}_R = \Big(\mathbf{R} \mid \mathbf{t} \Big), \tag{6.92}$$

in the same world coordinate system located at the projection centre O_L. For the base, this gives $\mathbf{b} = \mathbf{t}$.

Following Definition 2.2, the x-axis describes the width and the y-axis the height of the image. We always assume the origin of the image coordinate system to be in the upper left corner

[21] Cf. Faugeras [105, p. 43].

of an image. As a consequence of the projection, the origin in pixel coordinates lies in the lower right while the x-axis of Fig. 6.10 points to the left whereas the y-axis points upwards. The camera coordinate system is adjusted to O_L so that the z-axis points in viewing direction perpendicular to the image plane, x increases to the right and the y-axis points downwards.

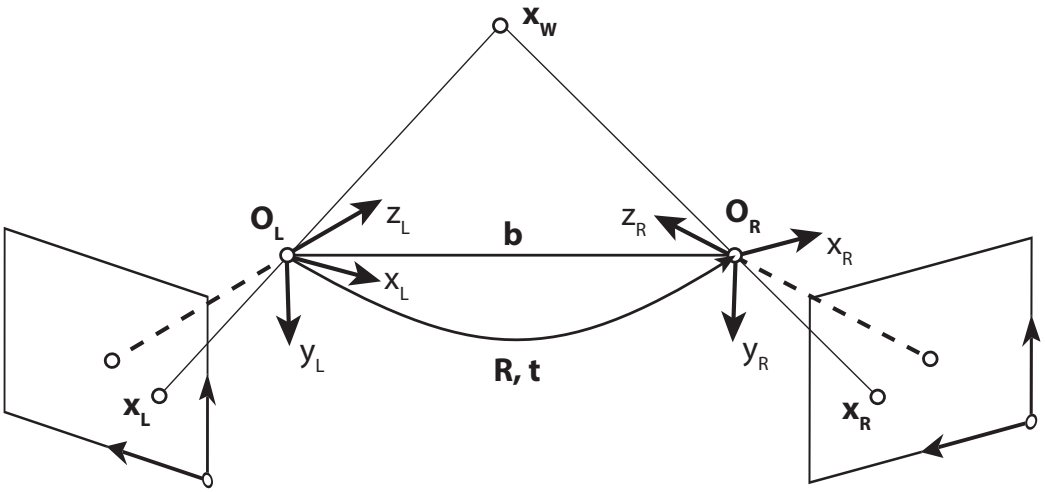


Fig. 6.10. **Schematic hardware stereo camera setup.** The baseline **b** separates the left camera with centre O_L from the right camera with centre O_R. A rigid transformation with rotation **R** and translation **t** transforms the left camera system x_L-y_L-z_L into the right coordinates x_R-y_R-z_R. A 3D world point $\mathbf{x}_W$ is projected onto $\mathbf{x}_L$ on the left and $\mathbf{x}_R$ on the right image plane.

6.3.1.1. Virtual Images

At first, we simplify the illustrated real scenario slightly. Since the different orientations of the axes and the origins in the middle of the scene may cause a mix-up of names and rather disturb the clear view for the analysis of the given geometry, we simply exchange the real images in our figure on its actual place for **virtual images** placed at twice the principal distance parallel to them in direction of the principle ray. This is the same setup as the one used in section 4.1 and the simplified scenario is shown in Fig. 6.11 where we note that all the indicated points are coplanar to the plane **E**.

An arbitrary point $\mathbf{x}_L$ in the left image may arise from the projection $\mathbf{P}_L$ of a world point $\mathbf{x}_W$. This transformation is not invertible, since the depth information is lost. As shown in Fig. 6.12, a given point $\mathbf{x}_L$ has indeed a whole line of potential world points $\mathbf{x}_W^1$, $\mathbf{x}_W^2$, $\mathbf{x}_W^3$, ... which can be assigned to it. If we want to know which point $\mathbf{x}_R$ in the right image corresponds to points coded by $\mathbf{x}_L$, we are restricted to the line l_R given by the projection of the plane **E** held by O_L, $\mathbf{e}_L$, and $\mathbf{x}_L$ into the right image. **E** is called the **epipolar plane**, the line l_R is called the **epipolar line** for $\mathbf{x}_L$.[22] These definitions are apparently symmetric by interchanging the descriptions of left and right.

[22] Cf. Hartley and Zisserman [165, pp. 239–241].

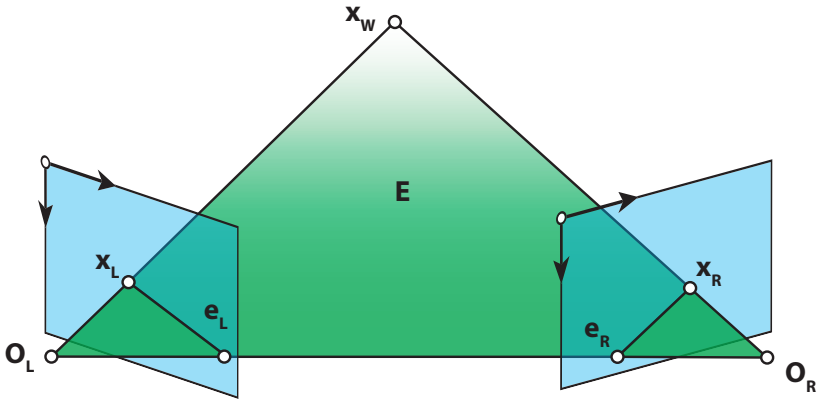


Fig. 6.11. **Virtual stereo camera setup.** The two virtual images are depicted in blue with the camera origins O_L and O_R. The world point $\mathbf{x}_W$ projects onto $\mathbf{x}_L$ on the left and $\mathbf{x}_R$ on the right virtual image plane. The epipolar plane $\mathbf{E}$ spanned by O_L, O_R, and $\mathbf{x}_L$ intersects with the virtual image planes in the epipolar points $\mathbf{e}_L$ and $\mathbf{e}_R$.

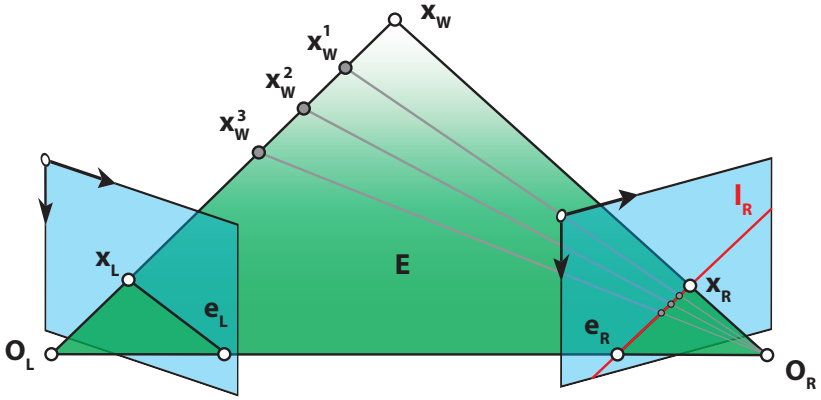


Fig. 6.12. **Schematical illustration of the epipolar geometry.** To recover the 3D coordinates of a world point $\mathbf{x}_W$ from its projection $\mathbf{x}_L$, multiple possible solutions lay on the ray from the left camera centre O_L through $\mathbf{x}_L$. This line is incident with the epipolar plane $\mathbf{E}$ and projects an epipolar line l_R into the right image. The epipolar line connects the epipolar point $\mathbf{e}_R$ with the correct projection $\mathbf{x}_R$ of the world point $\mathbf{x}_W$. Possible point combinations of $\mathbf{x}_L$ with the right image coordinates on l_R result in different other 3D world points $\mathbf{x}_W^1$, $\mathbf{x}_W^2$, $\mathbf{x}_W^3$, ... which are incorrect triangulations.

The **epipoles** $\mathbf{e}_L$ and $\mathbf{e}_R$ are the intersections of the image planes with the baseline joining the two camera centres. This line is invariant to movements of the world point $\mathbf{x}_W$ and its projections $\mathbf{x}_L$ and $\mathbf{x}_R$ respectively. Together with $\mathbf{x}_W$ this forms a pencil of epipolar planes around the baseline. Expressed in terms of the images, this gives two families of lines crossing $\mathbf{e}_L$ and $\mathbf{e}_R$.

6.3.2. Fundamental Matrix

Let us now **construct the epipolar line** l_R in the right image for a particular point $\mathbf{x}_L$ using homogeneous coordinates.

Suppose the points in image $\mathbf{I}_L$ and $\mathbf{I}_R$ are named as in Fig. 6.12 and the two projection

matrices are given by $\mathbf{P}_L$ and $\mathbf{P}_R$. As seen above, there is a whole line of possible world points corresponding to $\mathbf{x}_L$. We can construct this line with the two points O_L and

$$\mathbf{x}_W^0 = \mathbf{P}_L^+ \mathbf{x}_L \quad \text{such that} \quad \mathbf{P}_L \mathbf{P}_L^+ = \mathbf{I} \tag{6.93}$$

with the pseudoinverse matrix $\mathbf{P}_L^+$ of the projection $\mathbf{P}_L \in \mathbb{R}^{3\times4}$.[23] This gives the line

$$\mathbf{x}_W = \mathbf{x}_W^0 + \eta O_L, \quad \eta \in \mathbb{R}. \tag{6.94}$$

We choose one of the points on the line – namely $\mathbf{x}_W^0$ – and project it with $\mathbf{P}_R$ into the right image. This gives a point $\mathbf{x}_R$ on the epipolar line of the right image with

$$\mathbf{x}_R = \mathbf{P}_R \mathbf{x}_W^0 = \mathbf{P}_R \mathbf{P}_L^+ \mathbf{x}_L. \tag{6.95}$$

Since l_R is given by the join of two different points $\mathbf{e}_R$ and $\mathbf{x}_R$, we can write in homogeneous coordinates

$$l_R = \mathbf{e}_R \times \mathbf{x}_R = \mathbf{e}_R \times \left(\mathbf{P}_R \mathbf{P}_L^+ \mathbf{x}_L\right). \tag{6.96}$$

The epipole $\mathbf{e}_R$ itself is given by the projection $\mathbf{P}_R$ of O_L as

$$\mathbf{e}_R = \mathbf{P}_R O_L. \tag{6.97}$$

Substituting equation (6.97) into (6.96), we have

$$l_R = (\mathbf{P}_R O_L) \times \left(\mathbf{P}_R \mathbf{P}_L^+ \mathbf{x}_L\right). \tag{6.98}$$

Since

$$a \times b = \mathbf{S}_a b, \tag{6.99}$$

with the skew-symmetric matrix $\mathbf{S}_a$

$$\mathbf{S}_a = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}, \tag{6.100}$$

we can rewrite equation (6.98) as a matrix vector multiplication

$$l_R = \mathbf{S}_{\mathbf{P}_R O_L} \mathbf{P}_R \mathbf{P}_L^+ \mathbf{x}_L = \underbrace{\mathbf{S}_{\mathbf{e}_R} \mathbf{P}_R \mathbf{P}_L^+}_{\mathbf{F}} \mathbf{x}_L = \mathbf{F}\mathbf{x}_L \tag{6.101}$$

where $\mathbf{F} \in \mathbb{R}^{3\times3}$ is called the **fundamental matrix** (or bifocal tensor).[24]
We now give an example for the calculation of $\mathbf{F}$ for the case of two calibrated cameras.

[23] Cf. Xu and Zhang [456, pp. 75–78].
[24] Cf. Xu and Zhang [456, p. 34].

Example
Suppose the two projection matrices of a stereo camera system with the world origin in the left camera are given by

$$\mathbf{P}_L = \mathbf{K}_L \begin{pmatrix} \mathbf{I} \mid \mathbf{0} \end{pmatrix} \quad \text{and} \quad \mathbf{P}_R = \mathbf{K}_R \begin{pmatrix} \mathbf{R} \mid \mathbf{t} \end{pmatrix} \tag{6.102}$$

and we write

$$\mathbf{P}_L^+ = \begin{pmatrix} \mathbf{K}_L^{-1} \\ \mathbf{0}^{\mathrm{T}} \end{pmatrix}. \tag{6.103}$$

Then, the fundamental matrix is given as

$$\mathbf{F} = \mathbf{S}_{\mathbf{e}_R} \mathbf{P}_R \mathbf{P}_L^+ = \mathbf{S}_{\mathbf{P}_R O_L} \mathbf{P}_R \mathbf{P}_L^+ = \mathbf{S}_{\mathbf{K}_R \mathbf{t}} \mathbf{K}_R \mathbf{R} \mathbf{K}_L^{-1}, \tag{6.104}$$

which brings $\mathbf{F} = \mathbf{S}_\mathbf{t}\mathbf{R}$ in normalized coordinates of equation (6.92).

Geometrically speaking, $\mathbf{F}$ maps an arbitrary point from the 2D projective image plane of the left image to one of the epipolar lines from the pencil through $\mathbf{e}_R$, which gives a 1D projective space. $\mathbf{F}$ factors as a product of $\mathbf{S}_{\mathbf{e}_R}$ and $\mathbf{M} := \mathbf{P}_R \mathbf{P}_L^+$, thus $\mathbf{F}$ must be of rank 2 which can be seen in equation (6.101), where

$$\operatorname{rank}(\mathbf{M}) = 3 \quad \text{and} \quad \operatorname{rank}\left(\mathbf{S}_{\mathbf{e}_R}\right) = 2. \tag{6.105}$$

In terms of the point $\mathbf{x}_R$ the meaning of $\mathbf{F}$ is given by

$$l_R^{\mathrm{T}} \mathbf{x}_R = \mathbf{x}_R^{\mathrm{T}} l_R = \mathbf{x}_R^{\mathrm{T}} \mathbf{F} \mathbf{x}_L = 0, \tag{6.106}$$

which is also true for every scalar multiple $\eta\mathbf{F}$ of $\mathbf{F}$ with $\eta \in \mathbb{R}$ and gives a necessary condition for two points to correspond.

Altogether we formulate a definition of the fundamental matrix analogeously to Hartley et al. [165, p. 245].

Definition 6.4
*The **fundamental matrix** $\mathbf{F}$ of a stereo camera system with camera centres $O_L \neq O_R$ is given by the homogeneous matrix $\mathbf{F} \in \mathbb{R}^{3\times 3}$ with $\operatorname{rank}(\mathbf{F}) = 2$ which satisfies*

$$\boldsymbol{x}_R^T \mathbf{F} \boldsymbol{x}_L = 0 \tag{6.107}$$

for all corresponding points $\boldsymbol{x}_L \leftrightarrow \boldsymbol{x}_R$.

The matrix $\mathbf{F}$ decodes the entire epipolar geometry for the two images.
Besides the equations for the epipolar lines

$$l_R = \mathbf{F}\mathbf{x}_L \quad \text{and} \quad l_L = \mathbf{F}^{\mathrm{T}} \mathbf{x}_R, \tag{6.108}$$

we can detect the epipoles with the fundamental matrix, since

$$l_R^{\mathrm{T}}\mathbf{e}_R = (\mathbf{F}\mathbf{x}_L)^{\mathrm{T}}\,\mathbf{e}_R = \mathbf{x}_L^{\mathrm{T}}\mathbf{F}^{\mathrm{T}}\mathbf{e}_R = 0 \quad \forall \mathbf{x}_L, \tag{6.109}$$

$$l_L^{\mathrm{T}}\mathbf{e}_L = \left(\mathbf{F}^{\mathrm{T}}\mathbf{x}_R\right)^{\mathrm{T}}\mathbf{e}_L = \mathbf{x}_R^{\mathrm{T}}\mathbf{F}\mathbf{e}_L = 0 \quad \forall \mathbf{x}_R. \tag{6.110}$$

The epipoles $\mathbf{e}_L$ and $\mathbf{e}_R$ are given by the non-trivial kernel of $\mathbf{F}$ and $\mathbf{F}^{\mathrm{T}}$ respectively, so that

$$\mathbf{F}^{\mathrm{T}}\mathbf{e}_R = 0, \tag{6.111}$$

$$\mathbf{F}\mathbf{e}_L = 0. \tag{6.112}$$

Thus it can be interesting to determine the fundamental matrix from a given set of corresponding points if we do not have the projection matrices $\mathbf{P}_L$ and $\mathbf{P}_R$ from a calibration.

6.3.2.1. Calculating the Fundamental Matrix

Let us write the fundamental matrix as

$$\mathbf{F} = \begin{pmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{pmatrix} \tag{6.113}$$

and the homogenized coordinates of the two images as

$$\mathbf{x}_L = \begin{pmatrix} x_{Lx} \\ x_{Ly} \\ 1 \end{pmatrix} \quad \text{and} \quad \mathbf{x}_R = \begin{pmatrix} x_{Rx} \\ x_{Ry} \\ 1 \end{pmatrix}. \tag{6.114}$$

Similar to the previous ideas of section 3.8, we expand the equation from Definition 6.4 in order to minimize the algebraic residual. We get

$$\begin{aligned}
0 &= \mathbf{x}_R^{\mathrm{T}}\mathbf{F}\mathbf{x}_L && (6.115)\\
&= x_{Lx}x_{Ly}f_{11} + x_{Ly}x_{Rx}f_{12} + x_{Ly}f_{13} + x_{Lx}x_{Ry}f_{21} + x_{Rx}x_{Ry}f_{22} && (6.116)\\
&\quad + x_{Ry}f_{23} + x_{Lx}f_{31} + x_{Rx}f_{32} + f_{33} && (6.117)\\
&= \underbrace{\begin{pmatrix} f_{11} & f_{12} & f_{13} & f_{21} & f_{22} & f_{23} & f_{31} & f_{32} & f_{33} \end{pmatrix}}_{\mathbf{r}^{\mathrm{T}}} \cdot && (6.118)\\
&\quad \underbrace{\begin{pmatrix} x_{Lx}x_{Ly} & x_{Ly}x_{Rx} & x_{Ly} & x_{Lx}x_{Ry} & x_{Rx}x_{Ry} & x_{Ry} & x_{Lx} & x_{Rx} & 1 \end{pmatrix}^{\mathrm{T}}}_{\mathbf{d}} && (6.119)\\
&= \mathbf{r}^{\mathrm{T}}\mathbf{d} && (6.120)\\
&=: F(\mathbf{r}, \mathbf{d}). && (6.121)
\end{aligned}$$

With the n measured corresponding points $x_L^i \leftrightarrow x_R^i$ we can form a design matrix

$$\mathbf{D} = \begin{pmatrix} \mathbf{d}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{d}_n^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} x_{Lx}^1 x_{Ly}^1 & x_{Ly}^1 x_{Rx}^1 & x_{Ly}^1 & x_{Lx}^1 x_{Ry}^1 & x_{Rx}^1 x_{Ry}^1 & x_{Ry}^1 & x_{Lx}^1 & x_{Rx}^1 & 1 \\ & & & \vdots & & & & & \\ x_{Lx}^n x_{Ly}^n & x_{Ly}^n x_{Rx}^n & x_{Ly}^n & x_{Lx}^n x_{Ry}^n & x_{Rx}^n x_{Ry}^n & x_{Ry}^n & x_{Lx}^n & x_{Rx}^n & 1 \end{pmatrix} \tag{6.122}$$

to formulate a minimization problem for the algebraic residuals $F(\mathbf{r}, \mathbf{d}_i)$. Since the matrix $\mathbf{F}$ is homogeneous, we introduce the condition $\|\mathbf{r}\| = 1$ to avoid the trivial solution for equation (6.115) and hence get

$$\min \|\mathbf{D}\ \mathbf{r}\|^2 \tag{6.123}$$

$$\text{subject to } \|\mathbf{r}\| = 1. \tag{6.124}$$

Introducing a Lagrange multiplier gives

$$\min \|\mathbf{D}\ \mathbf{r}\|^2 - \lambda\left(\|\mathbf{r}\| - 1\right), \tag{6.125}$$

which yields in a similar way as in section 3.8 to an eigenvalue problem given by

$$\mathbf{D}^{\mathrm{T}}\mathbf{D}\ \mathbf{r} = \lambda\ \mathbf{r}. \tag{6.126}$$

Since $\mathbf{M} := \mathbf{D}^{\mathrm{T}}\mathbf{D} \in \mathbb{R}^{9\times 9}$ is symmetric and positive semi-definite, for its eigenvalues it holds

$$\lambda_k \in \mathbb{R}_0^+ \quad \forall k \in \{1, \ldots, 9\}. \tag{6.127}$$

Moreover, because

$$\|\mathbf{D}\ \mathbf{r}\|^2 = \mathbf{r}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathbf{D}\ \mathbf{r} = \mathbf{r}^{\mathrm{T}}\mathbf{M}\ \mathbf{r} = \lambda_k \mathbf{r}^{\mathrm{T}}\ \mathbf{r} = \lambda_k \|\mathbf{r}\|^2 = \lambda_k, \tag{6.128}$$

we look for the normalized eigenvector to the smallest eigenvalue λ_k with $k \in \{1, \ldots, 9\}$. The problem now is that it is not guaranteed that the rank constraint $(\mathrm{rank}(\mathbf{F}) = 2)$ of Definition 6.4 is satisfied. As proposed by Zhang [480, pp. 166–167] one can integrate such a condition a posteriori. In order to do this, we perform a singular value decomposition of the so far computed matrix

$$\hat{\mathbf{F}} = \mathbf{U}\hat{\Sigma}\mathbf{V}^{\mathrm{T}}, \tag{6.129}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices and

$$\hat{\Sigma} = \begin{pmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \sigma_3 \end{pmatrix} \tag{6.130}$$

is a diagonal matrix with $\sigma_1 \geq \sigma_2 \geq \sigma_3$.
Setting $\sigma_3 = 0$ and

$$\Sigma := \begin{pmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & 0 \end{pmatrix} \tag{6.131}$$

yields an estimation of the fundamental matrix

$$\mathbf{F} := \mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}}. \tag{6.132}$$

It can be shown that this procedure minimizes the Frobenius norm $\left\|\mathbf{F} - \hat{\mathbf{F}}\right\|$ under the condition that $\operatorname{rank}(\mathbf{F}) = 2$.[25]
When we combine all the steps, we come up with a method to estimate the fundamental matrix $\mathbf{F}$ presented in Algorithm 6.1.

Algorithm 6.1. Computation of Fundamental Matrix

Input parameters:

- Corresponding points $\mathbf{x}_L^i \leftrightarrow \mathbf{x}_R^i$ with $i \in \{1, \ldots, n\}$

Computation steps:

1. Set up design matrix
$$\mathbf{D} = \begin{pmatrix} x_{Lx}^1 x_{Ly}^1 & x_{Ly}^1 x_{Rx}^1 & x_{Ly}^1 & x_{Lx}^1 x_{Ry}^1 & x_{Rx}^1 x_{Ry}^1 & x_{Ry}^1 & x_{Lx}^1 & x_{Rx}^1 & 1 \\ & & & \vdots & & & & & \\ x_{Lx}^n x_{Ly}^n & x_{Ly}^n x_{Rx}^n & x_{Ly}^n & x_{Lx}^n x_{Ry}^n & x_{Rx}^n x_{Ry}^n & x_{Ry}^n & x_{Lx}^n & x_{Rx}^n & 1 \end{pmatrix}$$
2. Calculate scatter matrix $\mathbf{M} := \mathbf{D}^{\mathrm{T}}\mathbf{D}$
3. Solve eigenvalue problem $\mathbf{M}\,\mathbf{r} = \lambda\mathbf{r}$
4. Get $(\lambda_{min}, \mathbf{r}_{min})$ with $\lambda_{min} = \min\{\lambda_k \mid k = 1, \ldots, 9\}$
5. Fill $\hat{\mathbf{F}}$ with $\mathbf{r}_{min}$ (Equations (6.113), (6.118))
6. Perform singular value decomposition $\hat{\mathbf{F}} = \mathbf{U}\hat{\Sigma}\mathbf{V}^{\mathrm{T}}$ (Equations (6.129), (6.130))
7. Set $\sigma_3 = 0$ and form Σ (Equation (6.131))
8. Calculate fundamental matrix $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}}$

Output:

- Fundamental matrix $\mathbf{F}$

The process is often referred to as the **8-point algorithm** as proposed by Longuet-Higgins [260, p. 135] who first published a similar technique. The problem, however, is that the minimized error $F(\mathbf{r}, \mathbf{d})$ is an algebraic one. There are several other methods to minimize the geometric distance without artificially imposing the rank criterium after minimization. One of these is for example the Gold Standard method proposed by Hartley et al. [165, pp. 284–285].

[25] Cf. Zhang [480, pp. 191–192].

The so-parametrized geometry can now be used to formulate ways to extract 3D measurements from a set of images. Before we extract depth information from two images, we now look closer at the underlying geometry and make use of the representation of our points with homogeneous coordinates one more time.

6.3.3. Rectification

In the previous section we developed a method to determine the geometry of the stereo system with a given point set of corresponding points. We now use this knowledge to transform the two images in a way that they become coplanar in order to **simplify** the **search for a new point correspondence** within two transformed images **to one dimension**. We make this dimension the **x-axis** of some standard coordinate system as shown in Fig. 6.13.

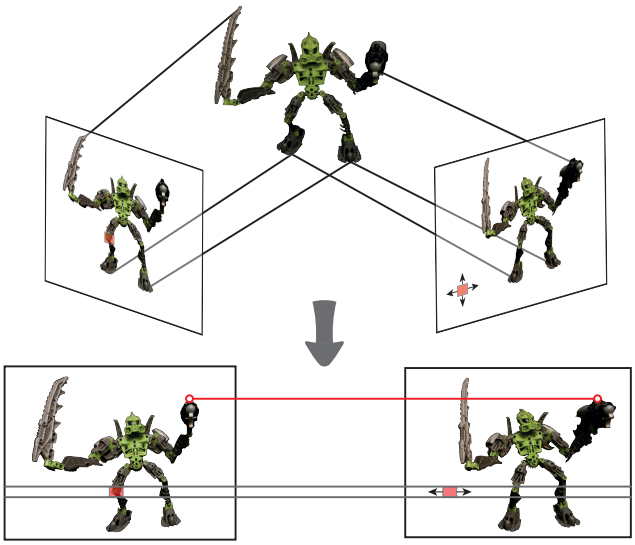


Fig. 6.13. **Rectified pair of images.** Two images are taken from a 3D object (top). Each is acquired from a different points of view (mid row). A 3D object part projects to a specific location in the image (as illustrated by the red area in the left image, mid row). The corresponding projection in the other image is restricted by the epipolar geometry and lies on the epipolar line. Thus the corresponding patch may be shifted in x- or y-direction along the line. A "rectifying" transformation as illustrated by the arrow changes the images (bottom) and aligns epipolar lines. The search space for corresponding points becomes restricted to the dimension aligned with the horizontal image axis.

The principle procedure of such a task is twofold. At first, we want to rotate the image planes of both images so that they become parallel. This transformation ideally makes sure that the epipolar lines are set up horizontally. Secondly, we adjust the image planes with a translation perpendicular to the plane in order to make them coplanar. Figure 6.14 illustrates this procedure for a pair of stereo images. The second step of the figure shows that this process is not unique. It works with every two translations that give coplanar image planes. A desirable transformation pair $\mathbf{H}_L$ and $\mathbf{H}_R$ would be a pair for which only little scaling for both images $\mathbf{I}_L$ and $\mathbf{I}_R$ is necessary.
Let us now formulate a way to estimate these two transformations.

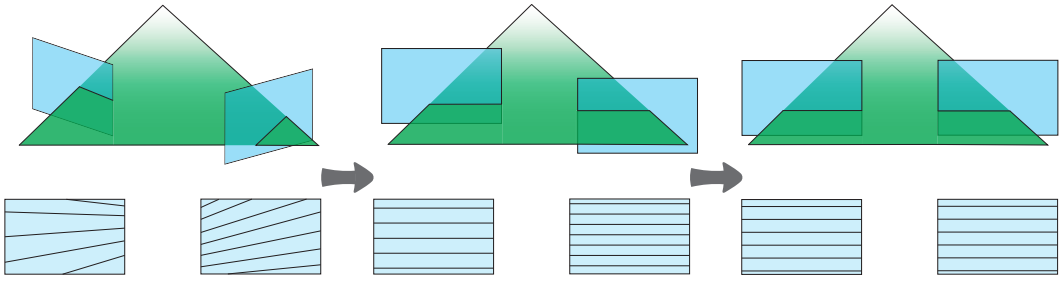


Fig. 6.14. **Rectification of image planes.** Two spatially separated images are shown (left) together with an example epipolar plane (top) and several epipolar lines (bottom). A first transformation makes the epipolar lines parallel and horizontally aligned (centre). The image planes become parallel. A second transformation (right) makes the image planes coplanar with a shift along the viewing direction. The epipolar lines become incident.

6.3.3.1. Calculating the Homographies

Suppose we have n corresponding points $\mathbf{x}_L^i \leftrightarrow \mathbf{x}_R^i$ with $i \in \{1, \ldots, n\}$ and already an estimation for the fundamental matrix $\mathbf{F}$ for instance from Algorithm 6.1. We now want to calculate two mappings that transform the points from the actual image planes onto coplanar virtual ones. We call these transformations of the images the two homographies $\mathbf{H}_L$ and $\mathbf{H}_R$.
Again, we want to make use of homogeneous coordinates to formulate rigid transformations as matrix-vector multiplications. A planar rigid transformation $\mathbf{W}$ that consists of a rotation $\mathbf{R} \in \mathbb{R}^{2\times 2}$ and a translation with the vector $\mathbf{t} \in \mathbb{R}^2$ is directly described with one matrix multiplication[26] with

$$\mathbf{W} = \begin{pmatrix} \boxed{\mathbf{R}} & \mathbf{t} \\ -0- & 1 \end{pmatrix}. \tag{6.133}$$

We start with the right image $\mathbf{I}_R$. The least deformation of such a homography is around the origin. As a first step, we therefore shift the origin to the centre of $\mathbf{I}_R$ by the translation $\mathbf{t}$ and rotate the image with $\mathbf{R}$ so that the epipole $\mathbf{e}_R$ is mapped onto the x-axis. We write this first transformation as a matrix $\mathbf{W}$ such that

$$\mathbf{W}\mathbf{e}_R = \begin{pmatrix} f & 0 & 1 \end{pmatrix}^{\mathrm{T}}. \tag{6.134}$$

We now want the epipolar lines to become parallel. Following the ideas of Hartley [163, p. 199], we push $\mathbf{e}_R$ along the x-axis to the point at infinity. A matrix $\mathbf{G}$ with

$$\mathbf{G}\mathbf{W}\mathbf{e}_R = \begin{pmatrix} f & 0 & 0 \end{pmatrix}^{\mathrm{T}} \tag{6.135}$$

[26] Cf. Richter-Gebert and Orendt [348, pp. 19–20].

is given by

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1/f & 0 & 1 \end{pmatrix}. \tag{6.136}$$

After the transformation $\mathbf{H}_R := \mathbf{GW}$, the epipolar lines of $\mathbf{I}_R$ are parallel. Figure 6.15 illustrates the consecutive operations.

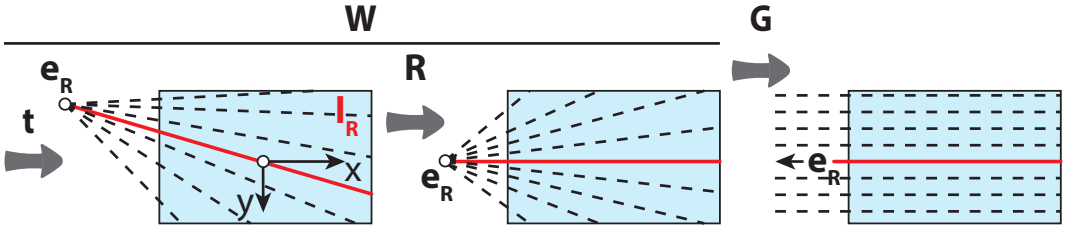


Fig. 6.15. **Parallelization of epipolar lines.** A first transformation $\mathbf{W}$ of image $\mathbf{I}_R$ shifts the origin by a translation $\mathbf{t}$ to the image centre (left) and rotates the image with a rotation $\mathbf{R}$ such that the epipole $\mathbf{e}_R$ is mapped onto the x-axis as shown in the middle. A consecutive transformation $\mathbf{G}$ (right) makes the epipolar lines parallel to the x-axis. The epipole $\mathbf{e}_R$ becomes a point at infinity.

We then seek for a matching transformation of the second image which projects the epipolar lines of $\mathbf{I}_L$ onto the transformed epipolar lines of $\mathbf{I}_R$. How we find such a homography can be answered with the help of the following two theorems. The first describes possible virtual projections that satisfy epipolar constraints and the second gives some criteria for homographies.

Theorem 6.1 (Stereo projections from fundamental matrix)
The general ***two projection matrices*** $\boldsymbol{P}_L \in \mathbb{R}^{3\times4}$ *and* $\boldsymbol{P}_R \in \mathbb{R}^{3\times4}$ *that project 3D points onto two image planes are* ***given by the fundamental matrix*** $\boldsymbol{F}$ *via*

$$\boldsymbol{P}_L = \left(\boldsymbol{I} \mid \boldsymbol{0}\right) \tag{6.137}$$

with the identity matrix $\boldsymbol{I} \in \mathbb{R}^{3\times3}$, $\boldsymbol{0} \in \mathbb{R}^3$ *and*

$$\boldsymbol{P}_R = \left(\boldsymbol{S}_{\boldsymbol{e}_R}\boldsymbol{F} + \boldsymbol{e}_R\boldsymbol{v}^T \mid \eta\boldsymbol{e}_R\right) \tag{6.138}$$

with the epipole $\boldsymbol{e}_R$ *and an arbitrary* $\boldsymbol{v} \in \mathbb{R}^3$ *and* $\eta \in \mathbb{R} \setminus \{0\}$.

Calculating first $\mathbf{e}_R$ with equation (6.111), choosing $\mathbf{v} = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}^\mathrm{T}$ and $\eta = 1$ gives the possible projection matrix[27]

$$\mathbf{P}_R = \left(\mathbf{S}_{\mathbf{e}_R}\mathbf{F} \mid \mathbf{e}_R\right). \tag{6.139}$$

With this theorem, we can state the second fact.

[27] Note the similarity to equation (6.92).

Theorem 6.2 (Matching homographies for stereo vision)
Let $\boldsymbol{I}_L$ and $\boldsymbol{I}_R$ be two images with the fundamental matrix $\boldsymbol{F}$. Then ***two homographies*** $\boldsymbol{H}_L$ *and* $\boldsymbol{H}_R$ *of* $\boldsymbol{I}_L$ *and* $\boldsymbol{I}_R$ ***match*** *in terms of epipolar line equality* ***if and only if***

$$\boldsymbol{H}_L = \left(\boldsymbol{I} + \boldsymbol{H}_R \boldsymbol{e}_R \boldsymbol{a}^T\right) \boldsymbol{H}_R \boldsymbol{M} \tag{6.140}$$

with $\boldsymbol{M} := \boldsymbol{P}_R \boldsymbol{P}_L^+$ as in theorem 6.1 and equation (6.93), the identity matrix $\boldsymbol{I} \in \mathbb{R}^{3\times 3}$ and an arbitrary $\boldsymbol{a} \in \mathbb{R}^3$.

The detailed proofs would be of minor value for our further studies and are left out here. The interested reader may be referred to Hartley et al. [165, pp. 255–256] for theorem 6.1 and to Hartley [163, pp. 119–120] for theorem 6.2.

Since we are interested in particular in the case where $\mathbf{H}_R$ transforms the epipole $\mathbf{e}_R$ of the right image to the point at infinity $\begin{pmatrix} f & 0 & 0 \end{pmatrix}^{\mathrm{T}}$, we can simplify equation (6.140) for our purposes to

$$\begin{aligned} \mathbf{H}_L &= \left(\mathbf{I} + \mathbf{H}_R \mathbf{e}_R \mathbf{a}^{\mathrm{T}}\right) \mathbf{H}_R \mathbf{M} && (6.141)\\ &= \underbrace{\left(\mathbf{I} + \begin{pmatrix} f & 0 & 0 \end{pmatrix}^{\mathrm{T}} \mathbf{a}^{\mathrm{T}}\right)}_{\mathbf{B}} \mathbf{H}_R \mathbf{M} && (6.142)\\ &= \mathbf{B}\mathbf{H}_0 && (6.143) \end{aligned}$$

with $\mathbf{H}_0 := \mathbf{H}_R \mathbf{M}$ and

$$\mathbf{B} = \begin{pmatrix} 1 + f a_1 & f a_2 & f a_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} b_1 & b_2 & b_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{6.144}$$

This leaves three parameters which have no influence on the rectification of the two images since they only appear in the first row of the matrix. What we want is to avoid unnecessary image distortions. Fixing all other parameters, we therefore formulate an optimization problem for the disparity. With n known point correspondences $\mathbf{x}_L^i \leftrightarrow \mathbf{x}_R^i$ with $i \in \{1, \ldots, n\}$, this gives the following minimization of the sum of squared distances:

$$\begin{aligned} &\min \sum_{i=1}^{n} \left\| \mathbf{H}_L \mathbf{x}_L^i - \mathbf{H}_R \mathbf{x}_R^i \right\|^2 && (6.145)\\ &= \min_{\mathbf{b}} \sum_i \left\| \mathbf{B}\mathbf{H}_0 \mathbf{x}_L^i - \mathbf{H}_R \mathbf{x}_R^i \right\|^2 && (6.146)\\ &= \min_{\mathbf{b}} \sum_i \left\| \mathbf{B}\hat{\mathbf{x}}_L^i - \hat{\mathbf{x}}_R^i \right\|^2 && (6.147)\\ &= \min_{\mathbf{b}} \sum_i \left(b_1 \hat{\mathbf{x}}_{L1}^i + b_2 \hat{\mathbf{x}}_{L2}^i + b_3 \hat{\mathbf{x}}_{L3}^i - \hat{\mathbf{x}}_{R1}^i \right)^2 + \left(\hat{\mathbf{x}}_{L2}^i - \hat{\mathbf{x}}_{R2}^i \right)^2 + \left(\hat{\mathbf{x}}_{L3}^i - \hat{\mathbf{x}}_{R3}^i \right)^2 && (6.148) \end{aligned}$$

$$= \min_{\mathbf{b}} \sum_i \left(b_1 \hat{\mathbf{x}}^i_{L1} + b_2 \hat{\mathbf{x}}^i_{L2} + b_3 \hat{\mathbf{x}}^i_{L3} - \hat{\mathbf{x}}^i_{R1} \right)^2 + \underbrace{\sum_i \left(\hat{\mathbf{x}}^i_{L2} - \hat{\mathbf{x}}^i_{R2} \right)^2 + \left(\hat{\mathbf{x}}^i_{L3} - \hat{\mathbf{x}}^i_{R3} \right)^2}_{C} \tag{6.149}$$

$$= \min_{\mathbf{b}} \sum_i \left(b_1 \hat{\mathbf{x}}^i_{L1} + b_2 \hat{\mathbf{x}}^i_{L2} + b_3 \hat{\mathbf{x}}^i_{L3} - \hat{\mathbf{x}}^i_{R1} \right)^2 + C \tag{6.150}$$

with the homogenized points $\hat{\mathbf{x}}^i_L := \mathbf{H}_0 \mathbf{x}^i_L$ and $\hat{\mathbf{x}}^i_R := \mathbf{H}_R \mathbf{x}^i_R$ and the parameters

$$\mathbf{b} = \begin{pmatrix} b_1 & b_2 & b_3 \end{pmatrix}^\mathrm{T} \in \mathbb{R}^3. \tag{6.151}$$

Since C is a constant value with respect to the parameter vector $\mathbf{b}$, this is equivalent to

$$\min_{\mathbf{b}} \sum_i \left(b_1 \hat{\mathbf{x}}^i_{L1} + b_2 \hat{\mathbf{x}}^i_{L2} + b_3 \hat{\mathbf{x}}^i_{L3} - \hat{\mathbf{x}}^i_{R1} \right)^2 \tag{6.152}$$

which gives a linear least squares problem and can thus be solved algorithmically.
To summarize this procedure all in one, we formulate Algorithm 6.2 and Fig. 6.16 shows some point correspondences for real images before and after the rectification.

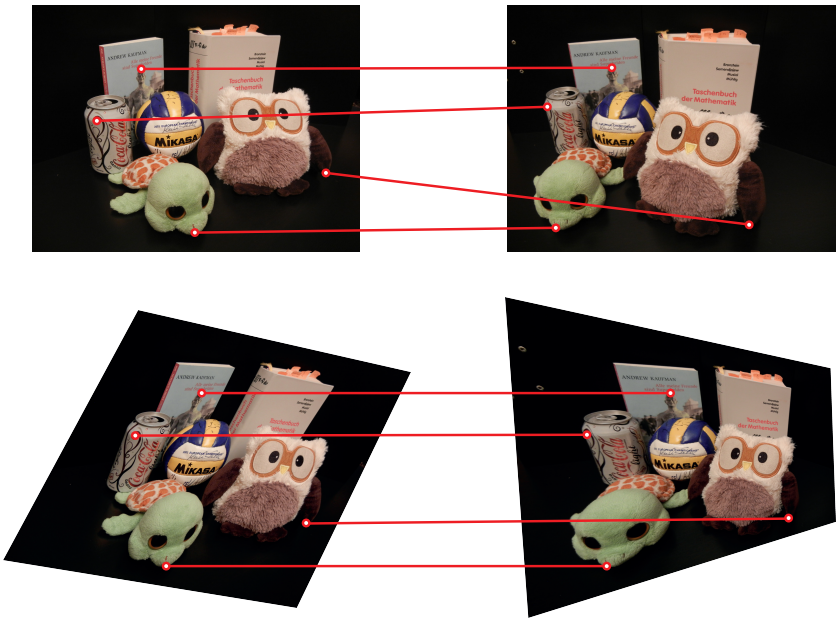


Fig. 6.16. **Corresponding points before and after rectification.** The upper row shows a pair of images before rectification where different corresponding points do not lie on parallel lines. The epipolar lines after rectification (lower row) are parallel such that correspondences for the same points are connected with parallel horizontal lines.

It may look like this process is computationally very complex for newly acquired images of a stereo camera rig. In fact, after the offline calculation of the two homographies by Algorithm 6.2, we can store the individual pixel transformation in a **lookup table** and are ready for efficient online rectification as long as the relative position of the two cameras is fixed. This can be achieved, for instance, by mounting the cameras on a solid rigid frame. Furthermore, taking for example bilinear interpolation as in section 3.7 into account even enhances the quality of the rectified output images. To achieve accurate point measurements, it is for example possible to use the same calibration target as in section 4.2.

Algorithm 6.2. Rectification of Stereo Images

Input parameters:

- A pair of stereo images $\mathbf{I}_L$ and $\mathbf{I}_R$
- Corresponding points $\mathbf{x}_L^i \leftrightarrow \mathbf{x}_R^i$ in homogeneous coordinates, $i \in \{1, \ldots, n\}$

Computation steps:

1. Estimate fundamental matrix $\mathbf{F}$ with Algorithm 6.1
2. Calculate epipole $\mathbf{e}_R$ by solving $\mathbf{F}^{\mathrm{T}}\mathbf{e}_R = 0$ (Equation (6.111))
3. Compute pose matrix for transformation of $\mathbf{I}_R$
   - Shift origin of $\mathbf{I}_R$ to the centre $\rightarrow$ translation $\mathbf{t}$
   - Rotate $\mathbf{e}_R$ onto x-axis $\rightarrow$ rotation matrix $\mathbf{R}$
   - Form pose matrix $\mathbf{W}$ with $\mathbf{t}$ and $\mathbf{R}$ (Equation (6.133))
4. Calculate x-coordinate f with equation (6.134) as $\mathbf{W}\mathbf{e}_R = \begin{pmatrix} f & 0 & 1 \end{pmatrix}^{\mathrm{T}}$
5. With equation (6.135) form matrix $\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1/f & 0 & 1 \end{pmatrix}$
6. Calculate homography for right image $\mathbf{H}_R := \mathbf{G}\mathbf{W}$
7. Shift origin of $\mathbf{I}_L$ to the centre
8. Form matrices $\mathbf{P}_L = \begin{pmatrix} \mathbf{I} \mid \mathbf{0} \end{pmatrix}$ and $\mathbf{P}_R = \begin{pmatrix} \mathbf{S}_{\mathbf{e}_R}\mathbf{F} \mid \mathbf{e}_R \end{pmatrix}$ (Theorem 6.1) to calculate helper matrix $\mathbf{M} := \mathbf{P}_R\mathbf{P}_L^+$ (Theorem 6.2)
9. Get matching homography $\mathbf{H}_0 := \mathbf{H}_R\mathbf{M}$
10. Adjust homography for left image
    - Apply transformations to homogenized points: $\hat{\mathbf{x}}_L^i := \mathbf{H}_0\mathbf{x}_L^i$ and $\hat{\mathbf{x}}_R^i := \mathbf{H}_R\mathbf{x}_R^i \; \forall i \in \{1, \ldots, n\}$
    - Solve least squares optimization for $\mathbf{b} = \begin{pmatrix} b_1 & b_2 & b_3 \end{pmatrix}^{\mathrm{T}} \in \mathbb{R}^3$ (Equation (6.152)): $\min_{\mathbf{b}} \sum_i \left(b_1\hat{\mathbf{x}}_{L1}^i + b_2\hat{\mathbf{x}}_{L2}^i + b_3\hat{\mathbf{x}}_{L3}^i - \hat{\mathbf{x}}_{R1}^i \right)^2$
    - Form matrix $\mathbf{B}$ with equation (6.144): $\mathbf{B} = \begin{pmatrix} b_1 & b_2 & b_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
    - Calculate homography for left image $\mathbf{H}_L = \mathbf{B}\mathbf{H}_0$ (Equation (6.143))
11. Resample images $\mathbf{I}_L$ and $\mathbf{I}_R$ with transformations $\mathbf{H}_L$ and $\mathbf{H}_R$ ($\rightarrow \mathbf{I}_L^{rec}, \mathbf{I}_R^{rec}$)

Output:

- Homographies $\mathbf{H}_L$, $\mathbf{H}_R$ and rectified images $\mathbf{I}_L^{rec}$, $\mathbf{I}_R^{rec}$

The structure of the images after rectification is then not only utterly helpful for the search of corresponding points by reducing the search space to a pixel line but also for the extraction of depth information from $\mathbf{I}_L^{rec}$ and $\mathbf{I}_R^{rec}$ which we investigate as a next step.

6.3.4. Triangulation

We want to discuss how to calculate the depth information from a newly measured corresponding point pair $\mathbf{x}_L \longleftrightarrow \mathbf{x}_R$ within rectified images.
For this purpose we leave out the virtual images of section 6.3.1 for a moment and focus on the rectification scenario in terms of images behind the projection centres as illustrated in Fig. 6.17.

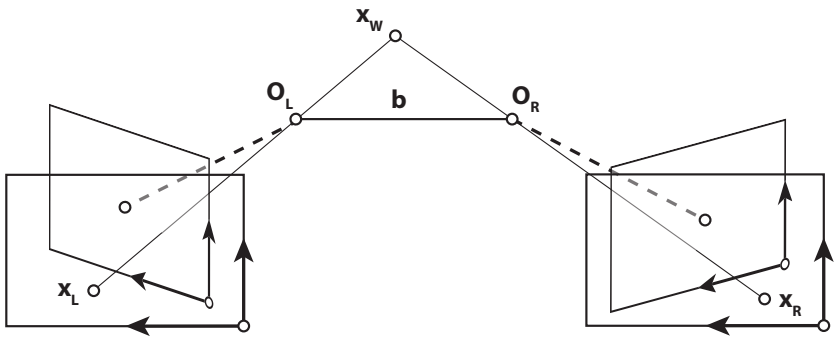


Fig. 6.17. **Rectification scenario with real image planes.** Two cameras are separated by a base distance b. The world point $\mathbf{x}_W$ is projected through the two pinholes O_L and O_R onto the left and right images (dotted line). The point coordinates after image rectification are given by $\mathbf{x}_L$ and $\mathbf{x}_R$ for the left and the right image respectively.

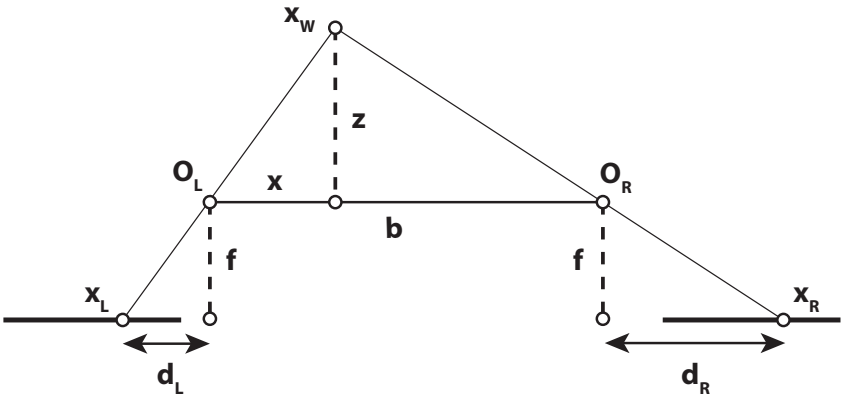


Fig. 6.18. **Geometry on epipolar plane.** The world point $\mathbf{x}_W = (x, y, z)$ is projected to $\mathbf{x}_L$ and $\mathbf{x}_R$ onto two rectified images. The pinholes O_L and O_R of the two cameras are separated by a baseline b parallel to the image planes at a distance f away from the common plane. The coordinate shift of the projected points $\mathbf{x}_L$ and $\mathbf{x}_R$ from the principle points of the cameras is given by d_L and d_R.

We look at the epipolar plane for some world point $\mathbf{x}_W$ with the two camera centres O_L and O_R where b gives the baseline. The geometry is illustrated in Fig. 6.18. The planes of the rectified images are parallel to the base and f gives their distance to the camera centres. $\mathbf{x}_L$ and $\mathbf{x}_R$ are the corresponding points in the two rectified images that arise from the projection of $\mathbf{x}_W$. The

illustrated distance z shows the desired depth coordinate of $\mathbf{x}_W$. If we locate the origins of the image coordinate systems for the rectified images at the base points of the principle rays, the differences of $\mathbf{x}_L$ and $\mathbf{x}_R$ to the base points are directly given by their x-coordinates x_{Lx} and x_{Rx} in the particular image coordinates.
The disparity d of the two projected points is then given by

$$d = x_{Rx} - x_{Lx}, \tag{6.153}$$

where x_{Rx} and x_{Lx} are the signed x-coordinates of the projected point.
We observe that the two triangles $O_L O_R \mathbf{x}_W$ and $\mathbf{x}_L \mathbf{x}_R \mathbf{x}_W$ are similar and we can therefore use the intercept theorem to write

$$\frac{z}{b} = \frac{z+f}{b+d}. \tag{6.154}$$

Rearranging this gives the depth

$$z = \frac{bf}{d} = \frac{bf}{d_{\text{pix}}\, s_{\text{x}}}, \tag{6.155}$$

where d_{pix} is the disparity in pixels and s_{x} the pixel width. We keep the different scaling in mind but use just d from now on to keep the notation as simple as possible. The parameters b, f, and s_{x} are known from calibration and rectification. After calculation they remain fixed. **The only value that varies is the disparity.**
The coordinate value of x for the world point $\mathbf{x}_W$ can then be calculated from the relation

$$\frac{z}{x} = \frac{f}{x_{Lx}}. \tag{6.156}$$

Thus we have

$$x = \frac{z x_{Lx}}{f} = \frac{b x_{Lx}}{d}. \tag{6.157}$$

The values of x and z are with equations (6.155) and (6.157) inversely proportional to the depth: The larger the disparity the closer is the point $\mathbf{x}_W$ to the image plane. This is illustrated in Fig. 6.19 where it can also be seen, that the resolution induced by a regular sampling becomes coarser for areas further away from the image planes.

To conclude this section we write down the **world coordinates of an arbitrary point $\mathbf{x}_W$** which projects to the corresponding point pair $\mathbf{x}_L \leftrightarrow \mathbf{x}_R$ with dehomogenized coordinates $\mathbf{x}_L = (x_{Lx}, x_y)$ and $\mathbf{x}_R = (x_{Rx}, x_y)$ of a rectified image pair. In terms of a coordinate system situated in O_L, the world coordinates are given by:

$$\mathbf{x}_W = (x, y, z) \tag{6.158}$$

$$= \left(\frac{b x_{Lx}}{x_{Rx} - x_{Lx}}, x_y, \frac{bf}{x_{Rx} - x_{Lx}} \right). \tag{6.159}$$

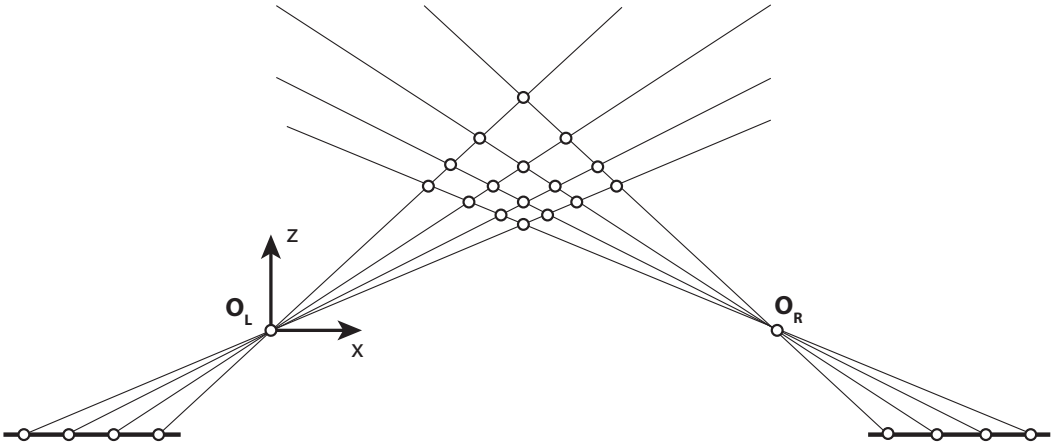


Fig. 6.19. **Different disparities and resolution in depth.** For equidistant image points (bottom) of two rectified images, the rays of sight through the two camera pinholes O_L and O_R intersect at different points (top). The spatial resolution in the overlapping field of view decreases for larger z values.

6.4. Depth Estimation

A topic we have not touched so far was the actual question of how to detect a stereo correspondence. How do we know that a point from the right image matches a point in the left one?

For the calculation of the fundamental matrix and the point correspondences for the rectification procedure, we can use the methods of chapter 3 and especially the algorithm for the extraction of ellipse centre coordinates (Algorithm 3.5). We can move a unicoloured planar object with differently coloured circles around in the visible area of the stereo camera system. If we acquire the images with both cameras simultaneously, we can directly calculate the fundamental matrix and rectification parameters as the correspondences are known by the colour-coding. For efficiency reasons, we extract several ellipse centres with Algorithm 3.5 from one image using the same target as for the camera calibration in section 4.2 shown in Fig. 6.20. The extraction of the marker coordinates from the images follows the same principle as described in section 4.2. Having fixed a certain distance between the dots, it is then possible to scale the axes of our world coordinate system according to a special unit such as metres.

Suppose all calibration, rectification and scaling is done. Extracting depth information from any new 3D world point within two images can still be a highly challenging task. The most difficult process then is to **solve** the **correspondence problem between** the **two images**. We differentiate between algorithms that reliably **triangulate a sparse set of points** and methods that estimate a **dense depth map** which assigns a distance value to each pixel individually. Depending on the underlying task either one of those can be the method of choice. For an accurate real-time pose tracker which we describe in chapter 7 we chose the former while dense fusion of different measurements in chapter 10.1 requires the latter.

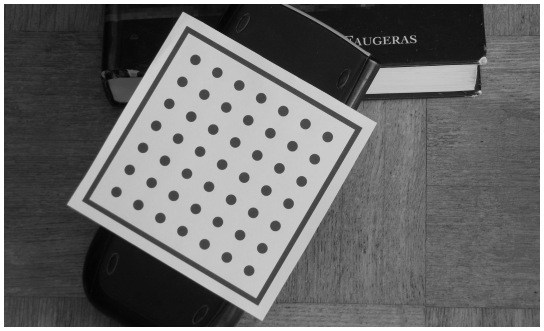


Fig. 6.20. **Calibration target.** The circular marker coordinates on the white background can be extracted robustly by Algorithm 3.5 with sub-pixel precision. With a known distance between the markers, it is possible to scale the coordinate frames to metric units.

6.4.1. Literature Overview

Before we explain the details of the approach chosen in the consecutive chapters, we take a tour through the depth estimation literature to summarize the state-of-the-art. This also provides an understanding of why certain approaches are advantageous over others depending on the use case.
Since depth estimation is an elemental task in computer vision, there are numerous methods and approaches to calculate distances depending on various inputs. Classical **multi-view geometry** utilizes several synchronized images while **structure from motion (SfM)** approaches use video sequences and temporally align images for triangulation. **Simultaneous localization and mapping (SLAM)** pipelines reconstruct the environment while later observations re-identify landmarks within the constructed map. With the advent of deep learning, recent approaches also tackle the ill-posed problem of **monocular depth estimation** or fuse multiple sensor modalities such as LiDAR and RGB images in **depth completion** networks.

6.4.1.1. Stereo Vision

The correspondence problem is a crucial part of binocular and multi-view vision and matching patches by visual content is at the core of most stereo methods. Dense stereo reconstruction algorithms classically follow a pipeline that consists of the general steps:[28]

1. **Matching cost computation**
2. **Cost aggregation**
3. **Disparity computation**
4. **Disparity refinement**

Early methods are mostly based on template matching techniques which compare small image windows under similarity measurements. The simplest and fastest algorithms use the **sum of absolute grey value differences** (**SAD**) or the **sum of squared grey value differences** (**SSD**) to compare these patches. The approaches are not invariant to illumination changes,

[28] Cf. Scharstein and Szeliski [366].

but illumination-invariant methods exist. One of them is using for instance **normalized cross correlation** (**NCC**) which allows to calculate similarities respecting the mean and the standard deviation within the chosen window.[29]
Traditional matching algorithms for stereo vision often follow a form of **semi-global matching** scheme as originally proposed by Hirschmüller [176]. Since the algorithm can be parallelized, efficient real-time implementations utilize the proposal of Hirschmüller [177] with sub-pixel metrics such as the one proposed by Birchfield et al. [25] to implement block matchers (**SGBM**) directly on chips.

Some scholars such as Lazaros et al. [238] as well as Tippetts et al. [412] try to summarize the long history of depth estimation from image pairs.
The **cost volume** $C(x, y, d)$ – commonly used to predict the disparity (or depth) d at pixel location (x, y) – can be interpreted as a probability measure for the likelihood of a specific depth d at the coordinate (x, y). Modern approaches make use of this and reformulate the classical pipeline with neural networks. Fig. 6.21 illustrates intermediate steps of the pioneering CNN from Zbontar et al. [474] for stereo matching directly after the matching cost computation and before all post processing and Fig. 6.22 shows the identical scene and its disparity as well as the cost volume after the post processing steps.

The utilized **Siamese network** of Zbontar et al. [474] compares image patches at different locations. Both the query and the potential target of a rectified image pair are fed into two towers with shared weights such that a cost volume can be formed through deep feature combination. While the first deep learning approaches provide robust disparity estimates, they are unable to run in **real-time** and their resolution is limited. More efficient approaches such as StereoNet[30] still rely on a relatively low-resolution cost volume to enhance computation time while **hierarchical upsampling refines the disparity** estimate through multiple residual corrections with the help of the RGB input image. Thus 60 fps are possible for 720p images on a consumer PC with an Nvidia Titan X GPU. Its successor ActiveStereoNet[31] extends the work with self-supervision to the domain of active sensing while maintaining the core efficiency. Other works such as the fast bilateral solver from Barron et al. [12] and the pixel-to-pixel mapping from Lutio et al. [267] also propose to upsample the depth map resolution with **RGB guidance**.
More than two views are addressed by the work of Choi et al. [68] who utilize multiple binocular stereo pipelines to create individual cost volumes with confidence. These are consecutively fused with a depth regression network. This requires knowledge of the amount of stereo pairs a priori and the number cannot be changed. The approach of Yao et al. [462] extends this principle and makes the pipeline agnostic to the amount of input views through the use of a differentiable homography warping that fuses the image information on top of one reference cost volume. Statistical measures are used to combine cross-view information with a variance-based metric.

Training a **stereo** network is possible with a large corpus of various **datasets** such as the KITTI benchmark[32] and the CityScapes dataset[33] which include synchronized stereo pairs from a

[29] A comparative analysis and an overview of the characteristics and performance of various classical methods is described by Roma, Santos-Victor, and Tomé [350].
[30] Cf. Khamis et al. [217].
[31] Cf. Zhang et al. [478].
[32] Cf. Geiger, Lenz, and Urtasun [139].
[33] Cf. Cordts et al. [72].

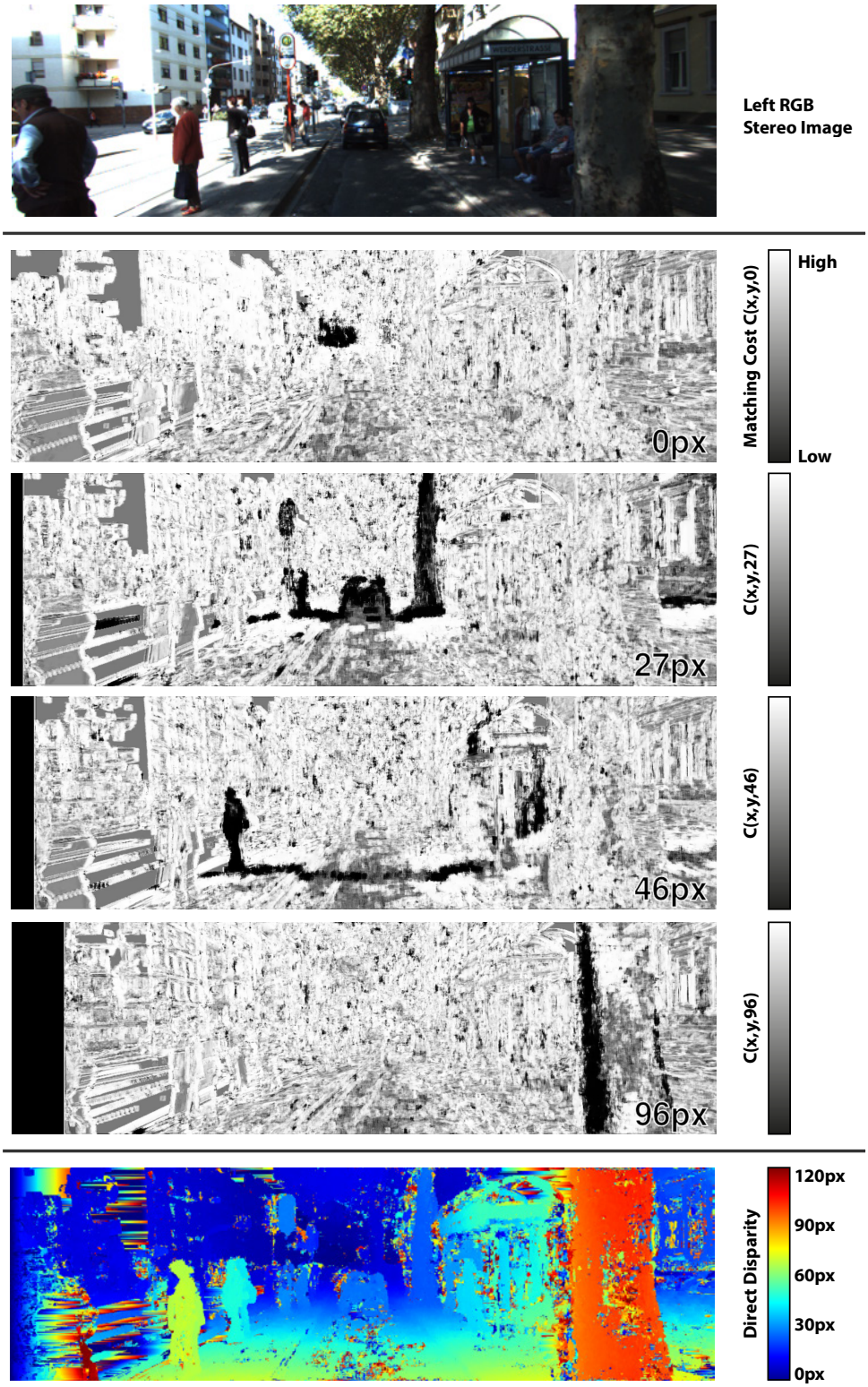


Fig. 6.21. **Stereo matching with neural network.** A stereo RGB pair (left image shown on top) is fed to the neural network of Zbontar and LeCun [474] for stereo matching. The matching cost $C(x, y, d)$ for various disparity levels is shown. Note the visible dark line approaching from far ($d = 0$) to closer distances ($d = 96$). The raw matching cost calculated by the network contains outliers and noise which are visible in the directly calculated disparity map (bottom) before post processing. The disparity map illustrates the direct maxima along the disparity dimension.

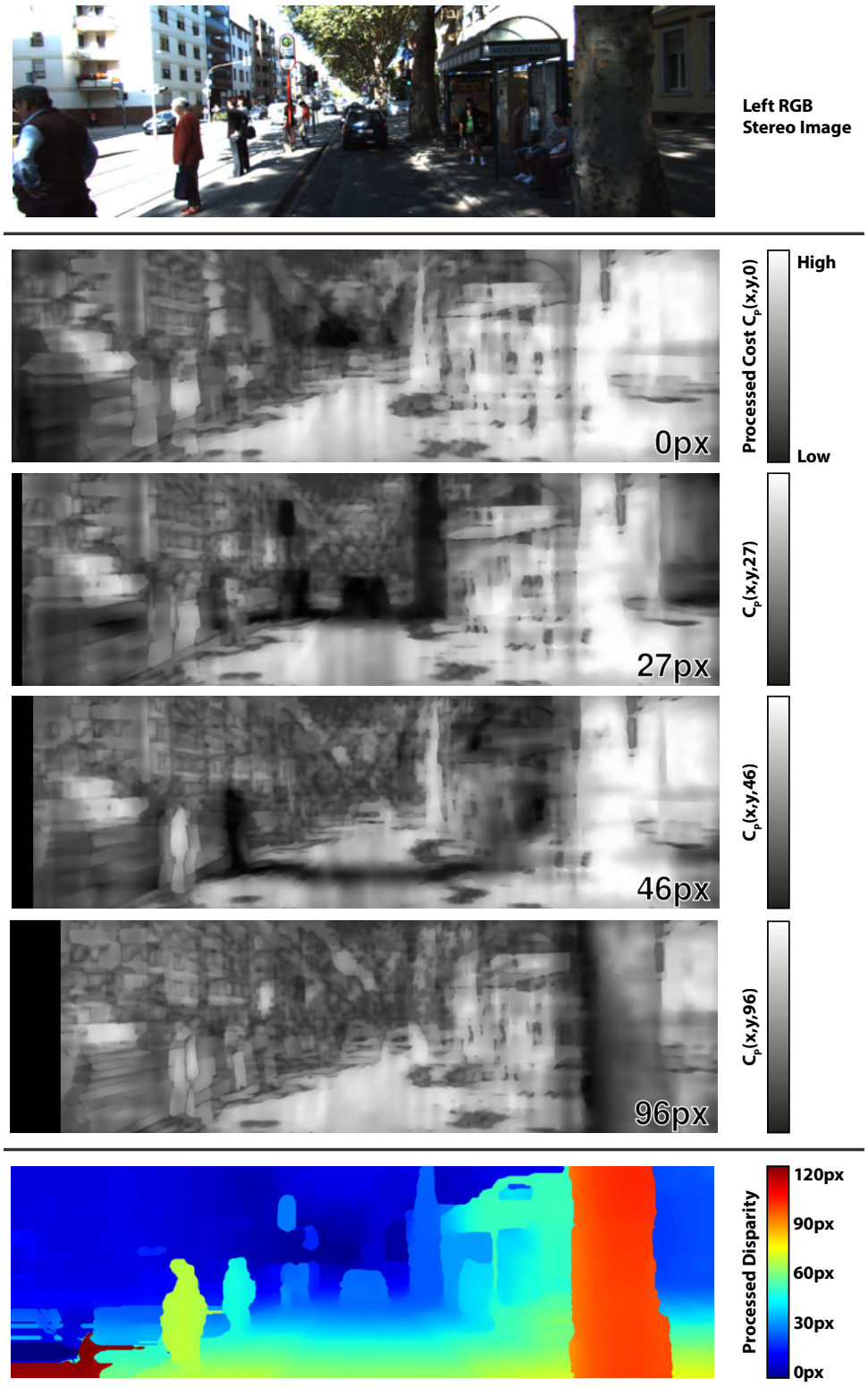


Fig. 6.22. **CNN stereo matching and post processing.** A stereo RGB pair (left image shown on top) is fed to the neural network of Zbontar and LeCun [474] for stereo matching. The post processed matching cost is illustrated as $C_p(x, y, d)$ for various disparity levels. Note the smoother boundaries in the disparity volume and the much cleaner processed disparity map at the bottom.

binocular camera pair rigidly mounted on a car. KITTI provides accurate but sparse ground truth measurements from a LiDAR laser scanner that is also mounted on the roof. While these scenes are restricted to driving environments, the Middlebury dataset[34] provides a diverse set of indoor scenes with high quality ground truth that can be used for supervision and Scene Flow[35] gives pixel-perfect ground truth for its synthetic renderings. Synthetic data training requires high quality renderings.[36] Further data generators such as SYNTHIA[37], Virtual KITTI[38] and its point cloud pendant[39], the CARLA simulator[40], and others[41] are used to simulate accurate data to train depth estimation networks. However, they mainly address driving scenarios.

Networks trained on one of these large scale datasets therefore usually suffer from **domain shift problems** when they are applied to another environment. However, ground truth data acquisition is a challenging task with real data and reliable supervision is often unavailable. Synthetic data is often not realistic enough and mainly addresses specific niche environments. Thus domain transfer across datasets and from synthetic to real data is naturally addressed by various groups. Tonioni et al. [415] propose a way for parameter fine tuning such that stereo networks generalize better to new data without the need for additional ground truth in the target domain. More recently, the same group with Tonioni et al. [416] suggests continuous online domain adaptation for disparity estimation with real-time applicability in unseen environments.

6.4.1.2. Depth Estimation beyond Multi-view Geometry

Spatio-temporal cues from different images in a video sequence also provide the possibility for triangulation even though the calibration scale is unknown and a depth or baseline normalization is often used instead.[42] For this, the pose between the images is estimated separately or on the fly. Temporal sequences are aligned in classical SfM[43] and SLAM frameworks[44] such as LSD-SLAM[45], ORB-SLAM[46], and DSO[47] while overlapping images with varying camera viewpoint are considered in the recent works of Agarwal et al. [2] as well as Knapitsch et al. [223].

Single view depth estimation is an ill-posed problem with naturally arising ambiguities by its geometric nature. Even though scale cannot be uniquely recovered solely relying on geometry, data-driven methods either estimate a normalized depth map or implicitly learn scale-awareness. Usually an encoder-decoder architecture is used for this task where the input is

[34] Cf. Scharstein et al. [365].
[35] Cf. Mayer et al. [280].
[36] Cf. Mayer et al. [279].
[37] Cf. Ros et al. [351].
[38] Cf. Gaidon et al. [131].
[39] Cf. Francis et al. [122].
[40] Cf. Dosovitskiy et al. [90].
[41] E.g. Miralles [287] for Atapour-Abarghouei and Breckon [8].
[42] The unknown scale in monocular video sequences can be recovered with visio-inertial pipelines where an additional sensor with an inertial measurement unit (IMU) provides the additional scale. This can be achieved through calculation of the relative motion between consecutive frames by robust integration over the acceleration signal.
[43] Cf. the pipelines of Faugeras and Lustman [106] as well as Huang and Netravali [188].
[44] A benchmark of various RGB-D visual odometry and SLAM frameworks is provided by Handa et al. [161].
[45] Cf. Engel, Schöps, and Cremers [98].
[46] Cf. Mur-Artal, Montiel, and Tardós [297] as well as Mur-Artal and Tardós [298].
[47] Cf. Engel, Koltun, and Cremers [97].

an RGB image and the output a depth map. The first learning-based method of this kind is proposed by Eigen et al. [95] with a fully connected layer close to the bottleneck limiting its application beyond training resolution. Further scholars propose fully convolutional residual architectures.[48] These networks require full supervision and ground truth extraction is usually complicated or not possible at all.

Stereo vision can help, though. If multiple cameras are considered during training time, **self-supervision** becomes possible by warping pixels from one view to the other with the current depth estimate. In the other reference system, a photo-consistency measure comparing the warped pixel with its counterpart usually provides the training signal. Thus, not only do additional ground truth labels become unnecessary but also the calibration-based error propagation is diminished. The works of Xie et al. [455] as well as Garg et al. [134] formulate ways to learn without ground truth by utilizing this kind of **stereo self-supervision** during training. However, their image formation models are not differentiable leading to limited quality results. Godard et al. [144] propose to use left-right consistency checks during training with a stereo vision systems and a differentiable image sampling strategy to improve the accuracy and Poggi et al. [331] use trinocular imaging for supervision.
The main source of information for depth estimation from a single static image is the dataset. Thus, these approaches are very sensitive to the data used in training and suffer from domain shift errors when applied to other image sources. To address this drawback, Guo et al. [156] use stereo matching[49] as a proxy to benefit from pre-training on synthetic data across domains and MegaDepth[50] is trained using photos from publicly available web sources.
While state-of-the-art monocular depth methods estimate a reliable depth ordering, they often suffer from over-smoothing.[51] Artefacts become visible as "flying pixels" in the free space close to depth discontinuities.

Some mono depth pipelines combine both classical SfM and self-supervision to pre-compute both **depth and camera poses** that can be used for supervision.[52] Zhan et al. [475] combine single view depth with the simultaneous estimation of the relative camera pose between images in a video sequence. The assumption underlying this concepts is that the scene is temporarily rigid and nothing moves. In this way, a frame at time t and another frame at time $t+n$ can be used as a stereo pair for triangulation while the baseline is normalized. For the often-considered driving scenarios, however, this assumption is wrong as multiple other vehicles as well as pedestrians move independently of the camera motion. In order to avoid incorrect supervision, Yin et al. [464] mask incoherently moving objects by using a 2D optical flow to estimate rigid scene content. Dense 3D optical flow and depth estimation can also be entangled.[53] A more elaborate network architecture is proposed by Zhou et al. [484] who use one part for mapping and multiple networks to track the motion. Despite the quality improvements of suchlike approaches, they suffer from bigger computational cost and require more training data. The memory footprint for the latter, for instance, only allows for a 80×60 pixel input resolution. Integrating pose, flow and depth mutually benefit each other and the

[48] Cf. Laina et al. [233].
[49] Also Watson et al. [446] incorporate information from stereo algorithms.
[50] Cf. Li and Snavely [253].
[51] Cf. Godard et al. [145].
[52] Cf. Klodt and Vedaldi [222] as well as Yang et al. [461].
[53] Cf. Zhao et al. [482].

estimation can be unified.[54] Moreover, additional semantic information[55] can improve depth and vice versa.

As most ground truth data is acquired with LiDAR laser scanners, also **modality combination** is considered with the problem of depth completion where the sparse active sensing of a LiDAR scan is fused with dense passively acquired information from RGB imagery. The sampling mask for the sparse signal is a crucial element for modality fusion.[56] Classical image processing techniques are used by Ku et al. [229] to solve this problem and Jaritz et al. [194] apply a learning based encoder-decoder network that encodes the different input modalities in a common latent space where feature fusion is possible before a consecutive decoder reconstructs a depth map.
It turns out that even a small amount of randomly sampled depth values can significantly improve the quality of the predicted depth map in comparison to monocular depth pipelines.[57] More recent approaches in this domain also consider self-supervision utilizing a stereo view or an image sequence without the need for ground truth annotations. Mutual pose prediction across an image sequence is for instance considered by Ma et al. [268] where a photometric loss provides the signal for backpropagation.

6.4.2. Sparse Stereo Matching

Despite various approaches for depth estimation, multiple views are still one of the best options to visually retrieve high-accuracy depth maps with precise triangulation, correct scale and sharp boundaries.[58] Our first target is an efficient and reliable pipeline with most accurate results applicable to domains such as medical environments where precision is essential. Thus, we follow a binocular stereo approach and detect the depth for special image features rather than a full depth map for the entire image. Since our later work focuses on centre coordinates of ellipses, these shapes code our features.

6.4.2.1. Disparity Gradient

We start with a rectified image pair and triangulate 3D coordinates efficiently and robustly. The 2D coordinates are decoded by the centres of identical ellipse markers as a results of Algorithm 3.5. Thus we cannot use surrounding texture for reliable triangulation and fully rely on geometric constraints. For a search along the epipolar line, a small neighbourhood may result in only one possible matching partner or several as illustrated in Fig. 6.23 where the search space for potential stereo matches is highlighted.

[54] Zou, Luo, and Huang [488] combine these concepts with different branches in a single network.
[55] Cf. Jiao et al. [196].
[56] Cf. Uhrig et al. [427].
[57] Cf. Ma and Karaman [269].
[58] Cf. Smolyanskiy, Kamenev, and Birchfield [385].

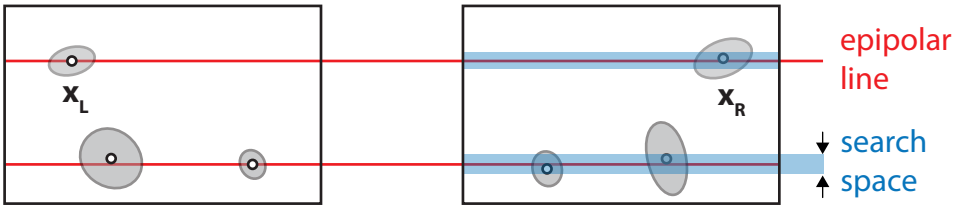


Fig. 6.23. **Possible matching partners for rectified stereo images.** Three ellipses with their centres are extracted by Algorithm 3.5 on the left and right rectified image. The epipolar line (red) defines the search space (blue) for a matching partners in the right image. The search space for the upper left point $\mathbf{x}_L$ and the lower right one in the left image is illustrated. Projective geometry and the rectification setup constraint possible matches. Matches such as $\mathbf{x}_L \longleftrightarrow \mathbf{x}_R$ that decode a 3D point are required to lie on the same epipolar line.

If there is just one potential partner $\mathbf{x}_R$ within this area it is most likely[59] the partner of $\mathbf{x}_L$ from the left image and the matching is trivial. The corresponding point pair is $\mathbf{x}_L \longleftrightarrow \mathbf{x}_R$ in such a case. If, however, this decision is not zero-one or occlusions occur, we need some algorithm to decide. There are many different approaches to detect these matches. We follow the ideas of Pollard et al. [332] and introduce another constraint on the point set: the **boundedness of the disparity gradient**.

Let us investigate two 3D points $\mathbf{x}_W^1$, $\mathbf{x}_W^2$ which project to $\mathbf{x}_L^i = \left(x_{Lx}^i, x_y^i\right)$ in the left image $\mathbf{I}_L$ and to $\mathbf{x}_R^i = \left(x_{Rx}^i, x_y^i\right)$ in the right rectified image $\mathbf{I}_R$, $i \in \{1,2\}$. We can define a **cyclopean image** $\mathbf{I}_C$ by averaging the coordinate values, which gives the coordinates

$$\mathbf{x}_C^i = \left(\frac{x_{Lx}^i + x_{Rx}^i}{2}, x_y^i\right), \quad i \in \{1,2\}. \tag{6.160}$$

These image coordinates are shown in Fig. 6.24.

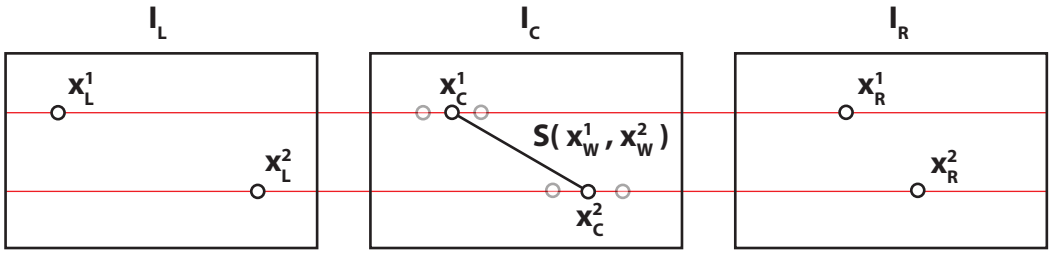


Fig. 6.24. **Left, right and cyclopean image.** The left image $\mathbf{I}_L$ (left) and the right image $\mathbf{I}_R$ (right) show the projections of two 3D points $\mathbf{x}_W^1$ and $\mathbf{x}_W^2$. The points project onto $\mathbf{x}_L^i$ and $\mathbf{x}_R^i$, $i \in \{1,2\}$ in the two images. The coordinates in their cyclopean image $\mathbf{I}_C$ (centre) are illustrated by $\mathbf{x}_C^i$ together with their cyclopean separation $S\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right)$.

[59] Unfavourable occlusion in the right image and another visible point on the same epipolar line can still cause problems.

The distance of $\mathbf{x}_C^1$ and $\mathbf{x}_C^2$ in this image is called the **cyclopean separation** $S\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right)$ of $\mathbf{x}_W^1$ and $\mathbf{x}_W^2$. It is given by

$$S\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right) = \sqrt{\left(\frac{x_{Lx}^1 + x_{Rx}^1}{2} - \frac{x_{Lx}^2 + x_{Rx}^2}{2}\right)^2 + \left(x_y^1 - x_y^2\right)^2} \tag{6.161}$$

$$= \sqrt{\frac{1}{4}\left(\underbrace{x_{Lx}^1 - x_{Lx}^2}_{d_L} + \underbrace{x_{Rx}^1 - x_{Rx}^2}_{d_R}\right)^2 + \left(x_y^1 - x_y^2\right)^2} \tag{6.162}$$

$$= \frac{1}{2}\sqrt{(d_L + d_R)^2 + \left(x_y^1 - x_y^2\right)^2}. \tag{6.163}$$

On the other hand, the **disparity difference** D is

$$D\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right) = \left(x_{Rx}^1 - x_{Lx}^1\right) - \left(x_{Rx}^2 - x_{Lx}^2\right) \tag{6.164}$$

$$= \left(x_{Rx}^1 - x_{Rx}^2\right) - \left(x_{Lx}^1 - x_{Lx}^2\right) \tag{6.165}$$

$$= d_R - d_L. \tag{6.166}$$

The **disparity gradient** is then the ratio of the disparity difference to the cyclopean separation with

$$\Gamma\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right) = \frac{D\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right)}{S\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right)} \tag{6.167}$$

$$= \frac{2\left(d_R - d_L\right)}{\sqrt{(d_L + d_R)^2 + \left(x_y^1 - x_y^2\right)^2}}. \tag{6.168}$$

This disparity gradient can be expected to be limited with[60]

$$\Gamma\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right) \in [-1, 1] \tag{6.169}$$

for two real world points $\mathbf{x}_W^1$ and $\mathbf{x}_W^2$ that can be seen simultaneously from a stereo vision system. We can use this to check for consistencies within putative triangulated point clouds where correct points mutually support each other.

6.4.2.2. Robust Coordinate Triangulation

With the thoughts from section 6.4.2.1 we score all possible matches according to all other point pairs that either support the pairing if $\left|\Gamma\left(\mathbf{x}_W^1, \mathbf{x}_W^2\right)\right| \leq 1$ or not. Weighting this score by the reciprocal distance from the considered match, we formulate Algorithm 6.3 to extract world coordinates from rectified stereo images by always taking the matched pair with the highest score.

We note that the epipolar constraint reduces the search space within the first loops and the uniqueness of a point pair guarantees that Q_L decreases since we delete the already matched pair in every outer loop.

[60] Cf. Šonka, Hlavac, and Boyle [389, p. 590].

Algorithm 6.3. World Coordinates from Rectified Images

Input parameters:

- Rectified image pair $\mathbf{I}_L^{rec}$ and $\mathbf{I}_R^{rec}$
- Parameters: Base b, focal length f, search bandwidth t_{epi}

Preprocessing:

1. Get sets Q_L and Q_R with 2D coordinates from $\mathbf{I}_L^{rec}$ and $\mathbf{I}_R^{rec}$ (Algorithm 3.5)
2. Prepare point cloud for 3D coordinates: $C = \emptyset$

Computation steps:

while $Q_L \neq \emptyset$ ***and*** *(*$\max S_{ij} \neq 0$ ***or*** *first iteration)* **do**

  for $\boldsymbol{x}_L^i \in Q_L$ **do**

    Prepare sets of potential matches: $M_i = \emptyset$

    // Look for potential matching partners along epipolar line

    for $\boldsymbol{x}_R^j \in Q_R$ **do**

      if $\left| x_{Ly}^i - x_{Ry}^j \right| < t_{epi}$ **then**

        Add point to potential point set $M_i = M_i \cup \mathbf{x}_R^j$

        Prepare score $S_{ij} = 0$

  // Calculate scores S_{ij} for potential matching pairs $\mathbf{x}_L^i \leftrightarrow \mathbf{x}_R^j$

  for $\boldsymbol{x}_L^i \in Q_L$ **do**

    for $\boldsymbol{x}_R^j \in M_i$ **do**

      Get world point $\mathbf{x}_W^1 = \left(\frac{b x_{Lx}^i}{x_{Rx}^j - x_{Lx}^i}, \frac{x_{Ly}^i + x_{Ry}^j}{2}, \frac{bf}{x_{Rx}^j - x_{Lx}^i} \right)$ (Equation (6.159))

      // Update scores

      for $\boldsymbol{x}_L^k \in Q_L \setminus \left\{ x_L^i \right\}$ **do**

        for $\boldsymbol{x}_R^l \in M_k$ **do**

          Get world point $\mathbf{x}_W^2 = \left(\frac{b x_{Lx}^k}{x_{Rx}^l - x_{Lx}^k}, \frac{x_{Ly}^k + x_{Ry}^l}{2}, \frac{bf}{x_{Rx}^l - x_{Lx}^k} \right)$

          // Check if disparity limit is violated (Equation (6.169))

          if $\left| \Gamma\left(\boldsymbol{x}_W^1, \boldsymbol{x}_W^2 \right) \right| < 1$ **then**

            Increment likelihood score: $S_{ij} = S_{ij} + \frac{1}{\left\| \mathbf{x}_W^1 - \mathbf{x}_W^2 \right\|}$

  Calculate indices with highest score $\left\{ (i,j) \mid S_{ij} = \max_{ij} S_{ij} \right\}$

  From $\mathbf{x}_L^i \leftrightarrow \mathbf{x}_R^j$ get 3D point $\mathbf{x}_W = \left(\frac{b x_{Lx}^i}{x_{Rx}^j - x_{Lx}^i}, \frac{x_{Ly}^i + x_{Ry}^j}{2}, \frac{bf}{x_{Rx}^j - x_{Lx}^i} \right)$ (Equation (6.159))

  Add point to cloud: $C = C \cup \mathbf{x}_W$

  Delete coordinate pair from sets: $Q_L = Q_L \setminus \left\{ \mathbf{x}_L^i \right\}$ and $Q_R = Q_R \setminus \left\{ \mathbf{x}_R^j \right\}$

Output:

- Set C of 3D world points

With this algorithm and the ideas explained beforehand, it is now possible to accurately extract world coordinates from any scene with an object that has circular markers on its surface. In chapter 7 we will extract such a cloud C within every frame of a stereo video stream and use such markers to detect the movement of the underlying object by describing its pose with the parametrization from section 6.2.
In the next chapter we also apply the theoretical knowledge established so far in the previous sections and use the developed techniques and algorithms to formulate a robust passive real-time tracking system for marked rigid body objects based on stereo camera observations.

7 High Performance Optical Tracking

> *Le temps*
> *est du mouvement*
> *sur de l'espace.*
>
> **– Joseph Joubert**
> (Essais et maximes de J. Joubert)[1]

The aim of the following chapter is to build a **high performance optical tracking system (OTS)** that allows usage in various industrial as well as medical applications. The OTS will not only be the backbone for medical tool tracking (cf. section 7.7), it also enables cooperative robotic applications (cf. section 7.9) and serves as the core technology to fuse different image modalities such as thermal imaging, ultrasound sensing, and gamma radiation imaging as discussed in chapter 10.

Everything we considered so far was static and we also start with a tracking by detection approach without temporal information. However, we finally also include the only preliminary definition from chapter 2 that has not yet been used: the video concept from section 2.2.2. Along with definition 2.3 for videos comes a consecutive set of images on which we can perform the developed algorithms over time. This allows us to formulate a processing for certain dynamic changes within the scene.

Since industrial environments can change drastically and many medical scenarios require high accuracy even under challenging illumination, we need a flexible, reliable and precise system. In the following sections, we make use of the already generated pipelines to accurately track artificially tagged objects in real-time and study the **movement of a static point set** within three-dimensional space. We then formulate a method to robustly determine the rigid body motion of these objects and discuss its efficiency and capabilities necessary to apply the OTS to a set of real problems from different fields.

Through a collaboration with the university hospital Klinikum rechts der Isar and the interdisciplinary research lab (IFL), medical case studies with the tracking system are performed in nuclear medicine, assistive movement therapy, and diagnostic sonography. We detail a first application in section 7.9 and use the OTS in later considerations within chapter 10 for both medical (10.2, 10.3) and industrial (10.1) sensor fusion.

[1] "Time is movement in space.", J. Joubert. Pensées, essais et maximes de J. Joubert suivis de lettres à ses amis et précédés d'une notice sur sa vie, son caractère et ses travaux: Volume 1 [Titre XIX, p. 322]. Tome Premier, Librairie de C. Gosselin, 1842.

Before we start to describe the tracking framework with a mathematical model following Busam et al. [49], we talk about useful hardware parts for a special setup that allows such a procedure and present the state-of-the-art in marker tracking.

7.1. System Components

A desirable method for the estimation of the position and orientation of an arbitrary object in space is ideally independent of certain patterns on the surface of the object. In some previous algorithms, our feature points have been described by circles and ellipses, their projected equivalents. Not every object offers such a special geometry on its surface and we thus generate such a geometry artificially. We simply stick **retro-reflective** circular **markers** on the surface of the object we want to track. The material is chosen to specifically reflect infrared (IR) electromagnetic waves.

7.1.1. Stereo Vision System

We mount a **ring** of IR LEDs for a **flash** around the lens of the camera and trigger the illumination with the camera exposure. If the diodes are triggered, a strong flash in the direction of the line of sight occurs as shown in Fig. 7.1. Since the markers are retro-reflective for this spectrum we have some weak diffuse reflection in different directions and a strong reflection back to the ring of LEDs as well as to the camera lens.

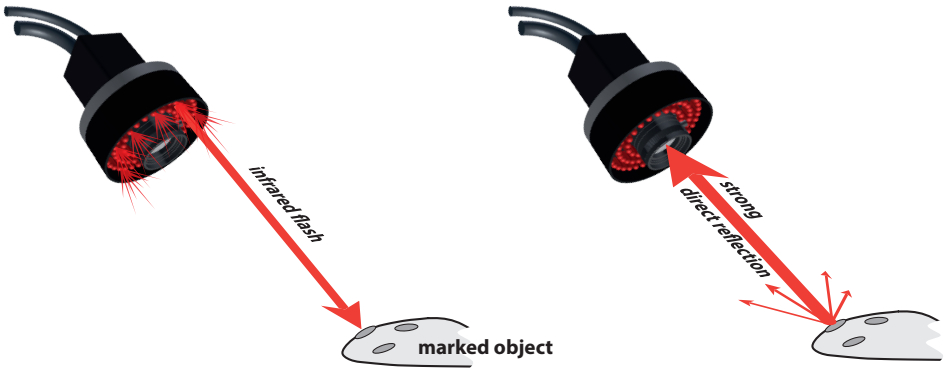


Fig. 7.1. Ring flash and retro-reflective markers. An LED ring is mounted around the lens of the camera and flashes during camera exposure (left). An object is marked with retro-reflective circular markers (bottom). These markers strongly reflect the light back to its source such that the combined reflected signal of the full ring shines into the lens while only a small part of the energy is dissipated through diffuse reflection (right).

This also offers another benefit. If we use a filter for this particular spectrum, we firstly get high intensity peaks for the circular image parts and, secondly, suppress other noisy sections since the retro-reflection of the flash is set to be much stronger than the natural reflections from other sources. This simplifies the search for ellipses and makes it more robust against other circular shaped structures within the visible area and less susceptible for ambient light.

Furthermore, we mount the two cameras on a rigid bar fixed on a tripod. The whole hardware setup for the triangulation of point clouds looks as the one shown in Fig. 7.2.

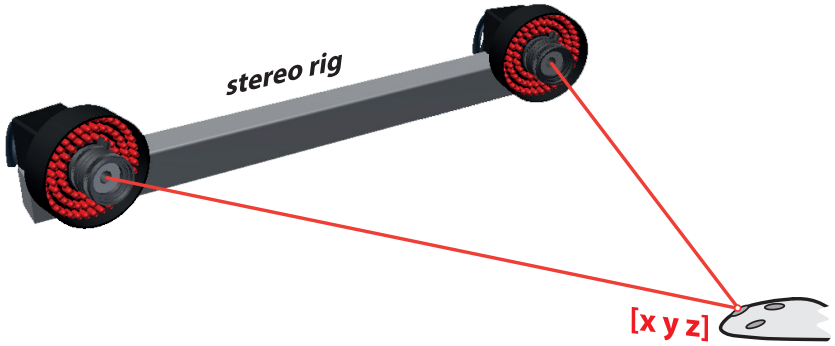


Fig. 7.2. **Stereo camera rig.** A binocular pair of identical cameras is fixed to a rigid bar (top left). After calibration, the acquisition of images with synchronized illumination allows for triangulation of 3D coordinates coded by circular markers attached to an object.

The **vision system hardware** is build around two GC1932MP cameras (SMARTEK Vision, Croatia) with Fujinon HF8XA-1 C-mount lenses with 8 mm focal length to observe a field of view of 58.4° × 44.6° on a 2/3" sensor. An IF 093 NIR band-pass filter (Schneider-Kreuznach, Germany) is used in each camera. Two ring lights FLDR-i70A (FALCON Illumination, Malaysia) illuminate the working volume with 875 nm wavelength and are triggered and powered synchronously by an IPSC2 strobe controller from SMARTEK Vision, Croatia at 24 VDC with 750 mA each. The cameras run in hardware trigger mode and receive their exposure signal also from the strobe controller. Acquisition time is set to 1.5 ms at a maximum frame rate of 24 Hz.

A CAD drawing of the first **prototype demonstrator** is illustrated in Fig. 7.3, where a stereo system observes a moving object in the form of an ultrasound transducer for which the pose has to be determined. The object moves on a fixed trajectory around a pivot point in the middle and the results together with tracking parameters and a 3D visualization can be shown in real-time on the attached screen in the background. Fig. 7.4 shows the first prototype realization of the setup with a demonstrator where the object of interest is a small airplane model. The knob on the frontal part is mounted to regulate the speed of the movement of the object around its trajectory. The lower part shows version two of the prototype modeling as well as a live demonstration with the more advanced housing at a trade fair. The two objects, torso and ultrasound transducer can be freely moved while they are tracked by the system which gives real-time feedback on the screen even with the high intensity ambient light at the booth.

7.1.2. Tracking Markers

Traditional tracking systems often used in medical applications consist of a metal rigid body frame with multiple marker spheres as shown in Fig. 7.5 on the left. The rigid metal frame is usually attached to the medical instrument such that the spherical markers are visible for a tracking system. Depending on the size of the frame and the tool, the setup becomes sometimes difficult to handle. Partial occlusion of individual markers through stain, blood, etc. as well

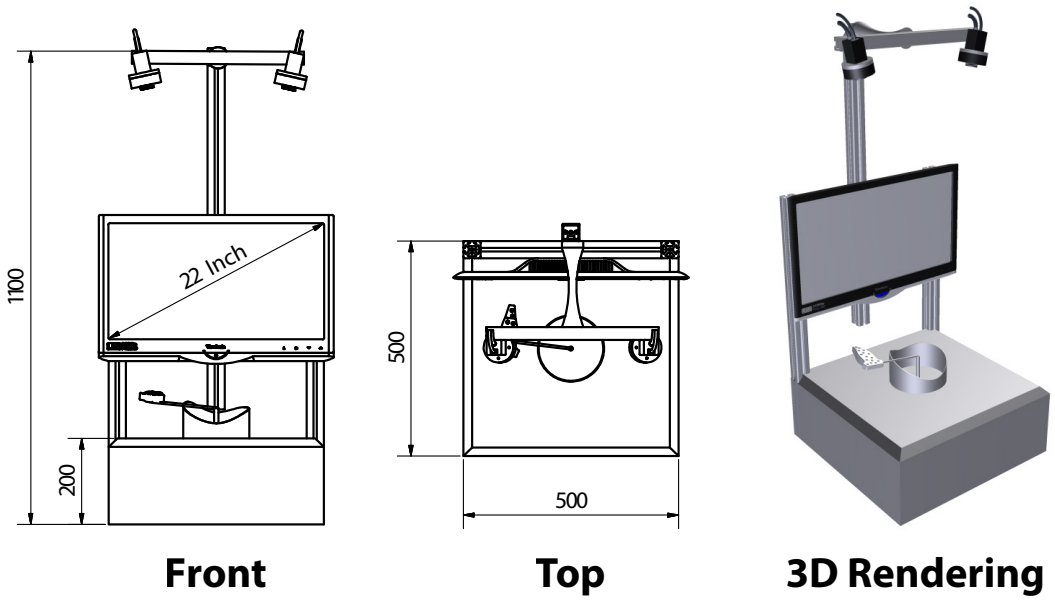


Fig. 7.3. **First OTS design.** The left and the middle image show a frontal and a top view of an OTS hardware prototype. Two cameras are mounted at a height of 110 cm above ground and 90 cm over a small box. The inner part of the box allows for space to fit the small electronics, a strobe controller, cabling and a computation unit while the connection cables for the stereo vision system on top can be fixed to the vertical bar. The stereo camera rig with identical cameras equipped with narrow band-pass filters and ring lights for the active illumination is put on a flexible arm on top of the setup such that it fully observes a table with an ultrasound transducer phantom. The phantom is equipped with retro-reflective markers and fixed to a moving bar that rotates on a trajectory around a pivot point to illustrate dynamic behaviour. The current status, tracking parameters, 3D visualization, and the stereo video signal can be shown for real-time feedback on a 22 inch monitor. For better visualization, a 3D rendering is shown on the right.

as line-of-sight obstacles impede the tracking process or stop pose estimation fully. Various ultrasound scans, for instance, cannot be realized with a rigidly attached frame at all. These include abdominal vascular ultrasound scans for which the transducer is put with pressure to the scanned tissue and becomes partly invisible in particular in adipose patients. A **multi-redundant marker setup** with self-adhesive markers allows for natural use of the medical instrument while the physician can fully focus on the medical question at hand. The tracked poses in case of vascular scans in the mentioned example can then be used for registration and 3D reconstruction to enable vascular diagnostics.

This still leaves the choice for the **size of the markers**. A large radius makes sure that the fitted elliptic shape for the contour is highly accurate even in case of noisy spots around the contour line, whereas a very small marker is error-prone to noise. However, a smaller marker is only slightly deformed by the shape of the surface and allows for the use of different markers even on small objects. We use reasonably small circles of 5 mm diameter for our experiments.

Having fixed the hardware framework so far, we can now formulate the tracking problem in mathematical terms and consider a robust pose estimation method.[2] To get a general idea of the process, let us first consider marker-based methods used in the literature and then split the consecutive task up into smaller pieces.

[2]We thereby follow the formulation of Busam [47].

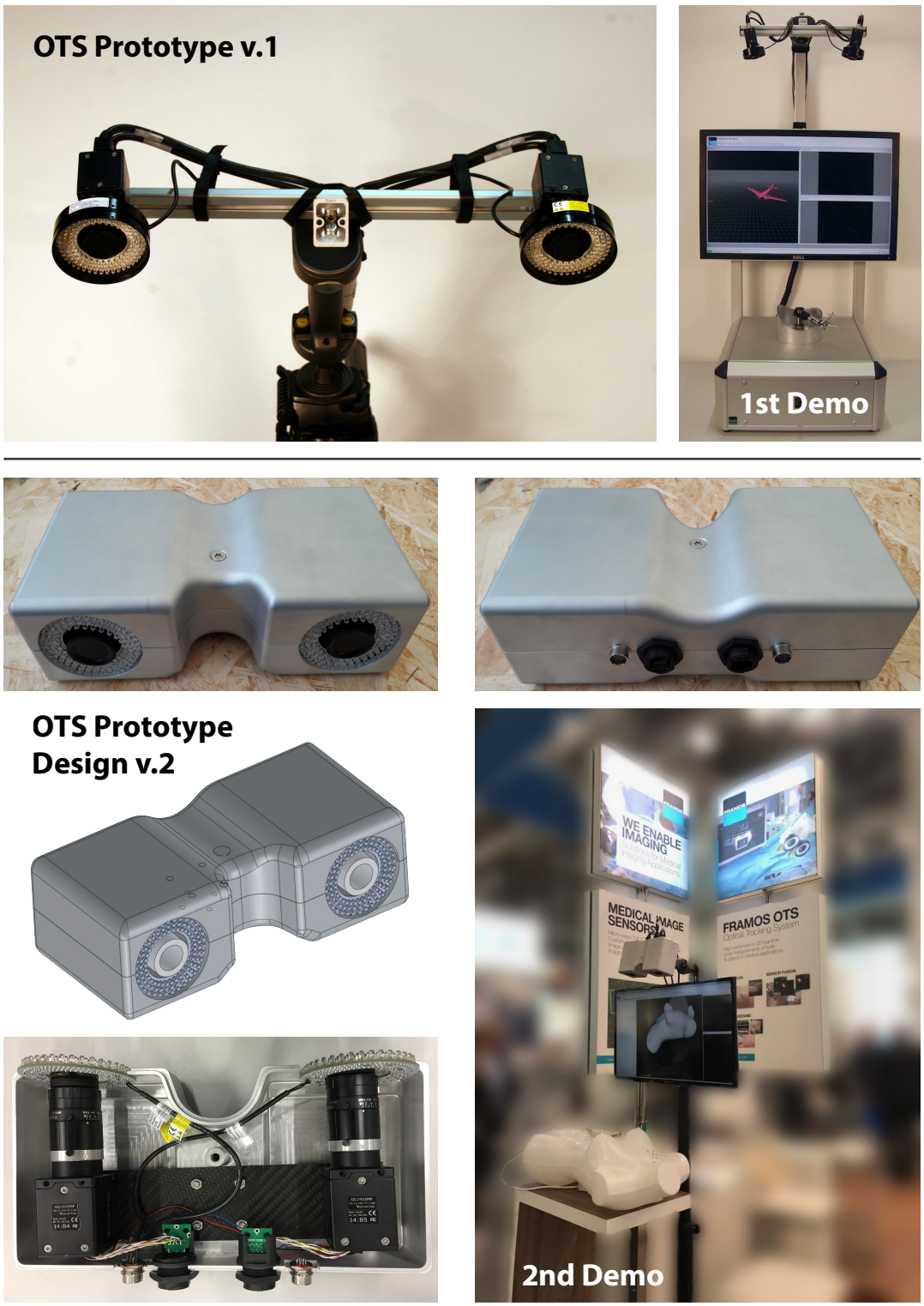


Fig. 7.4. **OTS prototype realization v.1 and v.2.** The optical tracker is realized as a demonstrator with a small airplane model (top) in the design version illustrated in Fig. 7.3. In version 1, the two cameras are rigidly attached to a metal bar. Consecutive updates are realized with version two (bottom). The stereo cameras are integrated into a housing (centre) where the cameras are sandwiched between two carbon fiber plates (see bottom left for the lower plate). The fibers are put in the direction of the baseline for maximal stiffness and temperature-stable calibration parameters while the external housing protects the whole rig. The setup is demonstrated (right) at a trade fair where two objects (a torso and an ultrasound transducer) are tracked individually.

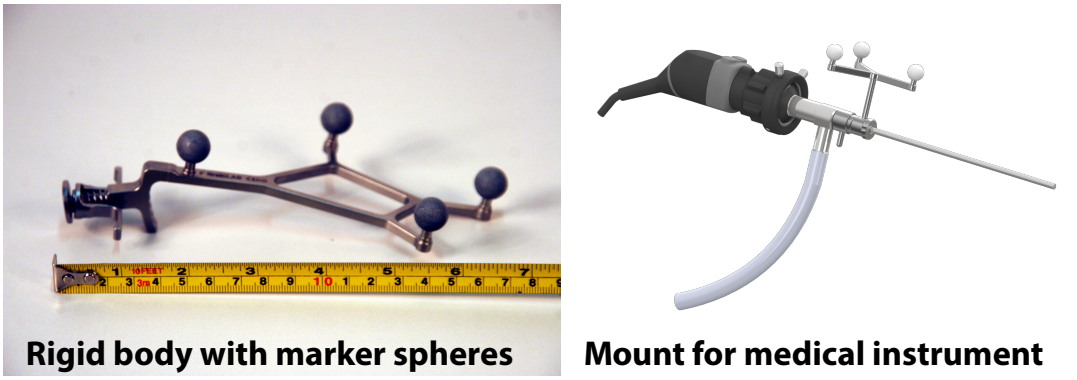


Fig. 7.5. **Tracking marker comparison.** A rigid body marker with four exchangeable spheres is shown on top left. Despite their size, these are usually attached to medical instruments (top right). Self-adhesive markers (bottom) with 5 mm diameter allow for intuitive use of the medical instrument while generating a multi-redundant cue for an optical tracking algorithm.

7.2. Marker Tracking Literature

Marker-based optical pose estimation has a wide application in **augmented (AR) and mixed reality (MR/XR)** applications where static or dynamic virtual content is overlaid on top of an RGB image. Besides gaming and remote support, various fields benefit from this technology to create content, fuse information (cf. chapter 10) or enable robotic vision (cf. section 7.9) and human-machine interaction (cf. section 10.3). Diagnostic sonography is just one of many medical applications, for instance, where AR glasses or augmented screens such as the one shown in the prototype in Fig. 7.6 can help the physician to spatially fuse data. Tracking is a crucial element in image-guided surgery to obtain the position and orientation of tools and the patient during medical procedures and various solutions exist. While mechanical and magnetic trackers can be used, optical tracking systems are often the preferred choice in **computer aided surgery** due to their accuracy and flexibility.[3]
We briefly review the literature on marker tracking, commercially available systems used in rigid body tracking, and natural extensions with less physical constraints.

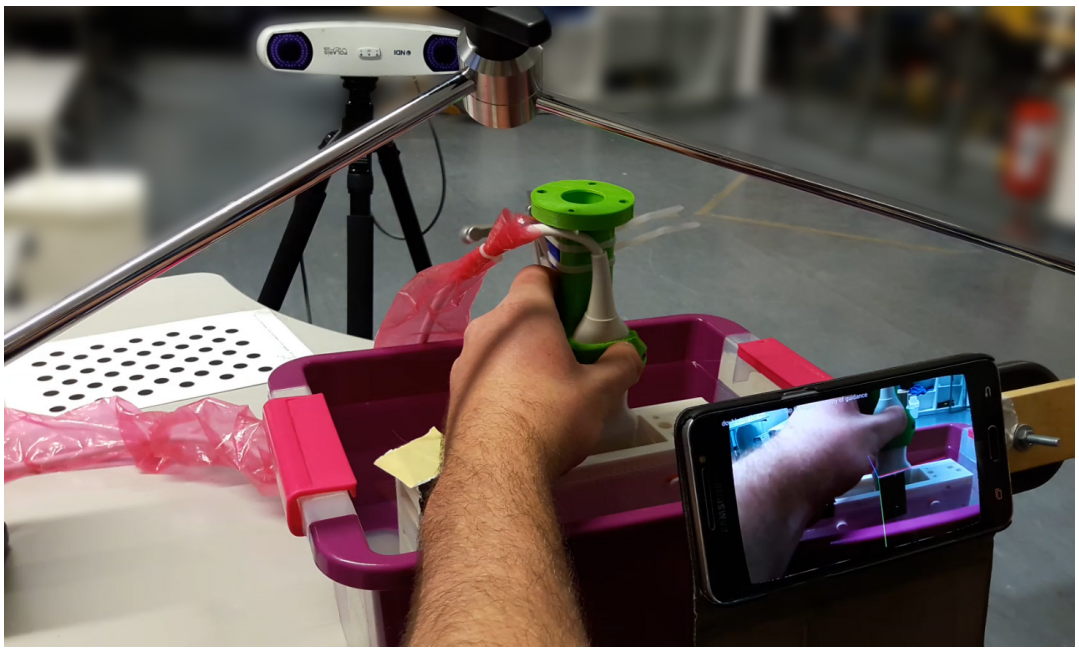


Fig. 7.6. **Mobile augmented reality.** An optical stereo tracking system (top, white) tracks both the rigid body marker attached to the green mount of an ultrasound transducer (centre) and a spherical marker frame on a wooden bar (right) where a mobile phone is fixed. All sensors are co-calibrated such that the live ultrasound video can be sent wireless to the mobile phone together with the correct pose such that it can be rendered at the current position on top of the RGB video stream of the mobile camera. A 3D coordinate frame additionally visualizes the live ultrasound coordinates.

7.2.1. Fiducial Markers

Besides the direct use of calibration boards as described in section 4.2 to estimate the camera pose relative to the marker, other **fiducial markers** are used to calculate the position

[3] Cf. Marinetto et al. [276].

and orientation of an attached object relative to the camera. Typically, a QR code like structure in black and white is used to provide high contrast features and to decode a marker ID simultaneously.

Early method rely on planar markers such as the ARToolkit from Kato et al. [204], the coded markers used by Naimark et al. [303] (Intersense), and AR-Tag presented by Fiala [111]. Cyclic codes are used by Bergamasco et al. [20] and a random dot pattern is proposed by Uchiyama et al. [426] which is extended to an efficient and robust marker that utilizes perspective invariants by Birdal et al. [28]. Many augmented reality applications enjoy the wide use of AprilTag[4], Pi-Tag[5], or the more recent ArUco as introduced by Garrido-Jurado et al. [137] which is available in the computer vision library OpenCV[6] together with its checkerboard extension ChArUco as illustrated in Fig. 7.7. With the advent of deep learning, the ChArUco marker tracker has become a robust upgrade by Hu et al. [186]. Their deep ChArUco system consists of a data driven ArUco detector that works well also under varying scene illumination. The pipeline then combines this with a sub-pixel refiner and the Perspective-*n*-Point (P*n*P) algorithm to retrieve accurate pose information even under varying image and light conditions.

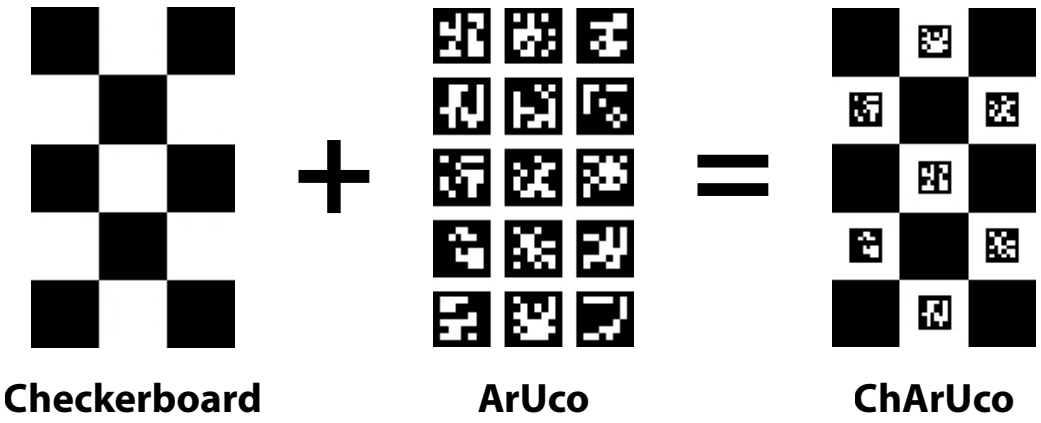


Fig. 7.7. **ChArUco marker as a combination of checkerboard and ArUco marker.** Corner detection of a checkerboard (left) can be done with high precision while the ArUco marker (in the middle) can be detected rapidly with its coded ID. The ChArUco marker (right) combines the advantages of both approaches by encapsulating different ArUco markers in the chessboard pattern.

In a first approach, we used an ArUco marker tracking to fuse different medical sensors[7] and noticed that the placement of planar markers on the object requires an attachment which is not always trivial to produce and impedes the natural use of the objects by changing its geometry. Accurate detection, however, is a must in many applications where more flexible and modifiable structures are superior. Moreover, the printer and paper quality of these usually self-made markers hamper the quality and lifespan of the tracking target. Various commercial systems have been evolved.

[4]Cf. Olson [316].
[5]Cf. Bergamasco, Albarelli, and Torsello [21].
[6]Cf. Bradski [38].
[7]Cf. Esposito et al. [101]. See also chapter 10.3.

7.2.2. Tracking Systems

There are various **commercially available tracking systems** to track rigid objects. Most of them require a statically mounted rigid body marker similar to the one shown in Fig. 7.5 to be attached to the object of interest. Multiple systems enjoy use in particular as medical trackers and have been tested and compared over the years.[8] Commercial tracking companies also support **comparative research studies**. The more recent study of Elfring et al. [96], for instance, is supported partly by Stryker Leibinger, Freiburg, Germany.

Tested trackers include the FlashPoint 580 (Image Guided Technology, Inc., Boulder, CO, USA) and various stereo vision systems such as the Northern Digital Inc. (NDI) Polaris P4, which is arguably the most prominent tracking system for surgical interventions as well as its successor, the NDI Polaris Spectra (successor of P4) and its small baseline version Polaris Vicra. Other stereo systems are FusionTrack from Atracsys Inc., Puidoux, Switzerland and the Navigation System II Camera from Stryker, Freiburg, Germany. OptiTrack (NaturalPoint, Inc., Corvallis, OR, USA) is tested by Marinetto et al. [276]. The tracker consists of multiple individual cameras that are co-calibrated and is evaluated with different occlusion scenarios in an eight cameras setup.
These systems usually work with **passive marker spheres** as shown in Fig. 7.5 and illuminate the scene with infrared light. Also **active infrared emitting diodes (IREDs)** in the near infrared (NIR) spectrum are used. Both solutions share the need for a known and calibrated rigid body that is likely to change the dimensions of the tracked object significantly. As a result, this can also impede the surgical workflow.

Optical trackers suffer from **line-of-sight restrictions** as the tracking system usually needs visibility of the full set of markers on a rigid body. While we propose a more flexible solution, partial visibility remains a need.
An alternative solution circumventing this issue is **electromagnetic (EM) tracking** where a changing EM-field induces a current in sensing elements attached to the object of interest. However, these systems need additional cabling to connect the sensors and interfere with metallic objects commonly present in industrial setups as well as in the OR, which makes measurements unreliable. Problem solving for measurement errors in optical trackers is often more straightforward as an occluder can be easily removed while a metallic object may not directly be visible which can make it hard to understand the cause of an error in these systems. Moreover, due to their technical nature, the clinical usage is restricted. EM tracking, for instance, cannot be used in patients with pacemakers. The team of Franz et al. [123] reviews EM trackers for medical applications extensively and the interested reader may be referred to their paper for more details.
We will not investigate EM tracking further, but address other possibilities to circumvent the strong line-of-sight requirement of commonly used tracking systems in sections 7.8 and 8.1.

[8] Cf. Chassat and Lavallée [64], Schmerber and Chassat [368], Khadem et al. [216] for early evaluations and Wiles, Thompson, and Frantz [452] as well as Maier-Hein et al. [270] for later ones.

7.2.3. Natural Markers

Besides smaller and more flexible marker setups, the natural extension is to use implicit decoding in object features to track without physical attachment. Such a **marker-free tracking** approach eliminates also the additional effort of marker-to-object calibration which adds complexity and can be time-demanding (cf. section 7.7). If the camera is mounted on the object (**inside-out tracking** case), a rich environment may help for navigation and provide enough details for orientation. We discuss this specific use case in section 8.1. However, in the classical outside-in scenario – where an object is tracked and the tracking system is statically observing the scene – the performance of marker-based methods is still much better than the one for markerless approaches. The medical requirements for multi-redundancy and sub-millimeter precision can additionally be met through the use of markers.

Before we discuss modern proposals to marker-free tracking in chapter 8, we now focus on describing an algorithm that enables a flexible, robust but accurate system with self-adhesive circular markers of retro-reflective material.

7.3. Matching Pose and Points

Real-time pose measurements of tools and objects is a core requirement in image guided medical applications and accurate and precise information is necessary to allow seamless surgeries. Intraoperative navigation and multi-modality fusion can simplify the workflow of physicians and boost diagnostic confidence levels. An algorithm for pose detection and marker matching needs to be robust to be of practical use in such setups. While the algorithmic design follows a pragmatic approach keeping in mind the medical use case, the OTS technology we develop hereafter is applicable to many other problems that require accurate estimation of position and orientation of objects in space.

In this section, we introduce the **mathematical concept behind** our **proposed optical tracking systems**, explain how it can quickly adapt with its self-adhesive markers to the geometry of the tracked object and estimate the pose even under severe occlusions such that the physician can fully concentrate on the medical aspects of the surgery rather than the technical challenges of the tracker. The pose calculation is independent of the marker distribution and the used algorithm is multi-redundant to enable robust pose detection with the proposed stereo camera system. The high-resolution tracking can thereby adaptively be adjusted with additional markers depending on the surrounding circumstances. Additional to the pose estimation capability, we present a teaching algorithm (cf. section 7.5) to autonomously build a marker model for the object of interest or to enhance the current one. To enable compatibility with existing solutions, the algorithm also works with traditional disposable spherical rigid body markers as shown in section 7.1.2 as well as reusable active IREDs.

7.3.1. Process Overview

Following the basics of the previous chapters, we use a calibrated stereo camera rig whose cameras acquire the scene. Once the two image planes are rectified, we save the homographies for the pixel transformation in a lookup table. Hence, we are ready for online acquisition of rectified images. The mathematical background and a detailed discussion of these preceding steps are given in sections 4.2 and 6.3. Using Algorithm 6.3, we can extract the 3D marker point coordinates of the markers on the object of interest in every frame of a video sequence. An algorithm that estimates a point set motion then uses the detected point sets of two consecutive frames n and $n+1$ to match them. Such a matching represents a transformation of the point set from frame n that maps the set with a rotation and a translation approximately onto the point set of image pair $n+1$.
The whole processing chain is illustrated in Fig. 7.8.

The matching part has so far only been explained informally and therefore needs further mathematical explanations. These are given in the following paragraphs.

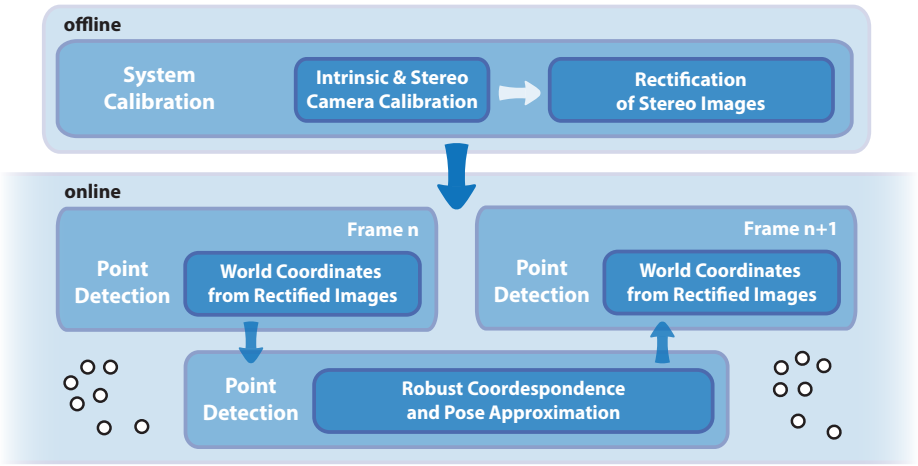


Fig. 7.8. **Processing for marker-based object tracking.** In a first stage (top), the intrinsic camera parameters are individually determined and the stereo vision system is calibrated (cf. Algorithm 4.1) such that the binocular image pair can be rectified (cf. Algorithm 6.2). While the rectification transformations can be calculated offline, each new incoming frame pair in the video stream (bottom) needs online processing. After triangulation of the world coordinates (cf. Algorithm 6.3 from markers, we match poses and corresponding points between frames or from an object model to the current measurement (cf. Algorithm 7.3).

7.3.2. Point Set Registration

A rigid body described uniquely by its shape or certain object features such as attached markers has **6 degrees of freedom** (DoF) for an arbitrary motion. These DoF consist of a three dimensional vector for a translation in space and the three rotational parameters. Following the insights of section 6.2, we call a description of the position and orientation of the object a **pose**. The optical tracking task then boils down to the estimation of the object's pose in each frame. This can either be done directly from frame to frame in a relative manner as pictured

in Fig. 7.8 or from a fixed object to the current frame by taking the information of the previous into account or not.

In any case, we have to match two point clouds in a way that transforming the first with the estimated pose approximates the second. Due to occlusion, errors, or mismeasurements, the clouds are not necessarily of the same size. Thus a one-to-one correspondence is not always guaranteed. We call these clouds

$$X = \left\{ \mathbf{x}_j \in \mathbb{R}^3 \mid 1 \leq j \leq J \right\}, \tag{7.1}$$

$$Y = \left\{ \mathbf{y}_k \in \mathbb{R}^3 \mid 1 \leq k \leq K \right\}. \tag{7.2}$$

They are symbolized with explained steps in a translation example with blue and red dots in Fig. 7.9. The correspondences $\mathbf{x}_j \longleftrightarrow \mathbf{y}_k$ are shown as connecting lines between the points of the two sets. We note that not every point here has a partner. An estimated transformation maps the blue points of set X onto the green points which then ideally lie up to measurement error close to the red points of set Y. An example with noisy point measurements is illustrated in Fig. 7.10.

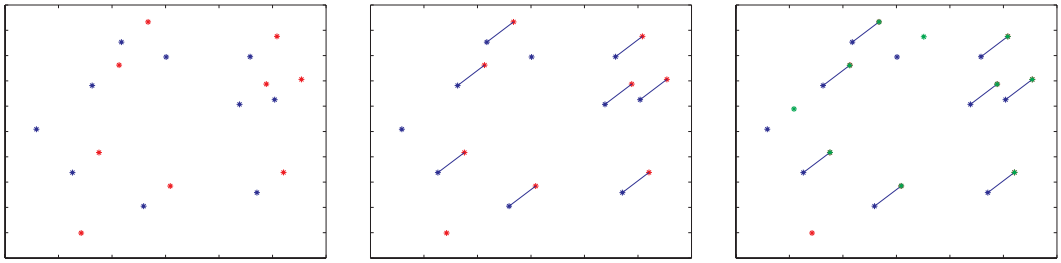


Fig. 7.9. Two point clouds with correspondence and pose estimation. Two point sets X and Y are illustrated in blue and red on the left. They are illustrations of two measurements of a translational motion at different time steps. Note that the sets do not have the same cardinality due to measurement noise. The correspondences are depicted in the middle as connecting blue lines. Not every point from set X has a partner in Y and vice versa. The relative pose between them is then used to transform the blue set X. The transformed set is visualized in green on the right. Even the location of non-visible points (i.e. missing measurements from set Y) can be determined as the points correspond to markers on a rigid body and move coherently.

Given such two point sets we therefore may ask two questions. Firstly: what are the **point correspondences**? And secondly: what is the **relative pose** transforming one cloud into the other? The answers to these questions are, however, interconnected since an answer to the first simplifies the latter task and vice versa.

Previous point set registration approaches such as **iterative closest points (ICP)**, proposed by Besl and McKay [23], are very fast to converge but fail in case of sparse point cloud data, high noise and an initialization far from the correct solution: all of which issues that may arise in our case.
Efficient ICP variants[9] such as multi-scale EM-ICP Granger et al. [153] using expectation maximization and the combination with the Levenberg-Marquardt solver by Fitzgibbon [116] share the same problem of sensitivity to the initial guess. This issue has been addressed by Go-ICP[10] where the authors use a branch-and-bound scheme to search the entire SE(3) space

[9] Cf. Rusinkiewicz and Levoy [359].
[10] Cf. Yang, Li, and Jia [460] as well as Yang et al. [459].

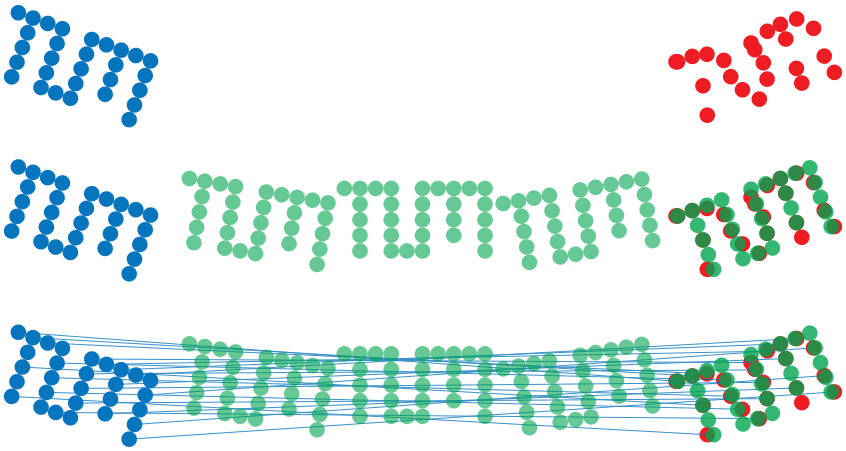


Fig. 7.10. **Point cloud alignment in the presence of noise.** The source object cloud is illustrated in blue (left) and the measured target cloud in the presence of noise in red (right). Intermediate pose transformations gradually move the estimate (green) in the middle onto the target. The final point correspondences are shown with connecting lines (bottom).

of rigid transformations for the globally optimal ICP solution at the cost of runtime.
Other approaches use **kernel correlation (KC)**[11] for registration in the presence of high noise where only close points are considered for potential matches and Myronenko et al. [302] propose **coherent point drift (CPD)** which is independent of the transformation model.[12] This probabilistic formulation using a maximum likelihood estimation and the assumption of motion coherence can be also applied for non-rigid registrations.
We have real-time requirements and sparse point clouds where occlusion-dependent missing points are likely to happen. Moreover, our algorithm is required to converge even if we have no close prior for the current pose. Thus, we take inspiration from **robust point matching (RPM)** as introduced by Gold et al. [147] who use soft assignment of correspondences and deterministic annealing for non-rigid registration of 2D point sets from written characters. We extend the approach to efficient and robust 3D pose fitting with rigid transformations using the specific structure of dual quaternions as developed in section 6.2.5.

In general, these rigid poses can be modeled by a rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$ and a translation $\mathbf{t} \in \mathbb{R}^3$. The correspondence can be expressed by a permutation matrix for which also 0-rows and -columns are allowed. We define this **match-matrix M** with entries m_{jk} by

$$m_{jk} = \begin{cases} 1, & \text{if } \mathbf{x}_j \leftrightarrow \mathbf{y}_k \\ 0, & \text{otherwise.} \end{cases} \tag{7.3}$$

[11] Cf. Tsin and Kanade [422].
[12] Cf. also Myronenko and Song [301].

7.3.3. Energy Functional

The addressed definitions allow us to formulate the problem as a minimization problem of an **energy functional**[13]

$$\min \; \mathrm{E}(\mathbf{M}, \mathbf{R}, \mathbf{t}) \tag{7.4}$$

where the energy E can be expressed as

$$\mathrm{E}(\mathbf{M}, \mathbf{R}, \mathbf{t}) = \sum_{j=1}^{J} \sum_{k=1}^{K} \mathrm{m}_{jk} \left\| \mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t} \right) \right\|^2 - \alpha \sum_{j=1}^{J} \sum_{k=1}^{K} \mathrm{m}_{jk}. \tag{7.5}$$

The first term with the Euclidean norm $\|\cdot\| = \|\cdot\|_2$ gives the distance of the estimation $\mathbf{R}\mathbf{x}_j + \mathbf{t}$ from the corresponding point $\mathbf{y}_k$ which we want to minimize. Since the trivial solution of this alone would be a non-correspondence scenario, we also use a second term which pushes the system towards matches.
How can we understand this? Rewriting equation (7.5) yields

$$\mathrm{E}(\mathbf{M}, \mathbf{R}, \mathbf{t}) = \sum_{j,k} \mathrm{m}_{jk} \left(\left\| \mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t} \right) \right\|^2 - \alpha \right). \tag{7.6}$$

The parameter α can be understood as a control parameter for the noise toleration of the equation and gives a threshold error distance for the squared distance of every pair $\mathbf{x}_j$, $\mathbf{y}_k$ where $\mathbf{y}_k$ is the actual measurement and $\hat{\mathbf{y}}_k = \mathbf{R}\mathbf{x}_j + \mathbf{t}$ its potential estimation. If it holds

$$\left\| \mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t} \right) \right\|^2 < \alpha, \tag{7.7}$$

for some $\mathbf{x}_j$, $\mathbf{y}_k$, it is

$$\left\| \mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t} \right) \right\|^2 - \alpha < 0 \tag{7.8}$$

and choosing $\mathrm{m}_{jk} = 1$ is favoured over $\mathrm{m}_{jk} = 0$ for a minimization. On the other hand, in case of

$$\left\| \mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t} \right) \right\|^2 > \alpha, \tag{7.9}$$

$\mathrm{m}_{jk} = 0$ is preferred.

For further interpretations and writing simplicity, we call the residuum of this estimation

$$\mathrm{d}_{jk} = \left\| \mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t} \right) \right\|. \tag{7.10}$$

For matched points with $\mathrm{m}_{jk} = 1$ this gives exactly the distance error of the estimated point $\hat{\mathbf{y}}_k$ from its partner $\mathbf{y}_k$ within the point cloud Y.

[13] Cf. Gold et al. [146].

The constraints of this minimization problem arise from the definition of the match-matrix and are in particular

$$\sum_k m_{jk} \leq 1 \quad \forall\, j \in \{1,2,\ldots,J\} \tag{7.11}$$

$$\sum_j m_{jk} \leq 1 \quad \forall\, k \in \{1,2,\ldots,K\} \tag{7.12}$$

$$m_{jk} \in \{0,1\} \quad \forall\, j \in \{1,2,\ldots,J\}, k \in \{1,2,\ldots,K\}. \tag{7.13}$$

The first inequality guarantees that every point $\mathbf{x}_j$ has at most one corresponding partner in the set Y. Vice versa, the second one makes sure that every point $\mathbf{y}_k$ has at most one partner in X. At last, the binary constraint assures that there is either a correspondence or not.

Altogether this gives a mixed minimization problem with a continuous part in the energy functional and a discrete part within the constraints. Furthermore, the constraints consist of two inequalities. In total we have 6 DoF for the pose and $J \cdot K$ decisions for the entries of the match matrix. There exist general solvers for problems like this, though real-time processing with such an algorithm is not possible if it does not take the special framework of the scenario into account.[14]

In the following, we want to develop an approach to approximate a solution to this complex problem that masters our runtime requirements. For reasons of clarity, we try to follow an intuitive pathway were we gradually develop a solution starting from this generic energy functional.

7.3.4. Constraint Relaxation

In order to develop a method that is able to solve the minimization in real time we transform several pieces of the original problem.
As just mentioned, the current formulation is neither completely discrete nor fully continuous. It is a **mixed problem** with three constraints, two inequalities and the binary limitation for the entries of the match-matrix $\mathbf{M}$. As a first step, we convert the inequalities into equalities by reshaping of $\mathbf{M}$. A second step then translates the entire problem into a fully continuous setting with a discrete counterpart.

At the moment, it is allowed for the rows and columns of the match-matrix to sum up to 0. This is the case, if there is no partner for some point from one of the sets X or Y. This brings the two inequalities

$$\sum_{k=1}^{K} m_{jk} \leq 1 \quad \forall\, j \in \{1,2,\ldots,J\} \tag{7.14}$$

$$\sum_{j=1}^{J} m_{jk} \leq 1 \quad \forall\, k \in \{1,2,\ldots,K\}. \tag{7.15}$$

[14] Cf. Mittelmann and Spellucci [289].

By appending another row $\mathbf{s}_r$ and another column $\mathbf{s}_c$ to the matrix $\mathbf{M}$, we get the matrix

$$\hat{\mathbf{M}} = \left(\begin{array}{c|c} \mathbf{M} & \mathbf{s}_c \\ \hline -\ \mathbf{s}_r\ - & \end{array} \right) \tag{7.16}$$

and can set these constraints to

$$\sum_{k=1}^{K+1} \hat{\mathrm{m}}_{jk} = 1 \quad \forall\ j \in \{1,2,\ldots,J\}, \tag{7.17}$$

$$\sum_{j=1}^{J+1} \hat{\mathrm{m}}_{jk} = 1 \quad \forall\ k \in \{1,2,\ldots,K\}, \tag{7.18}$$

which represents a normalization constraint on the rows and columns of the matrix. The entries of the vectors $\mathbf{s}_r$ and $\mathbf{s}_c$ are called **slack variables**[15] and they are 0 except for the case of no corresponding points in the other set. The additional row entry is 1 if the point $\mathbf{x}_j$ has no partner in Y and 0 otherwise. Respectively the auxiliary column entry is 1 if the point $\mathbf{y}_k$ has no partner in the set X.
We establish this matrix extension only to simplify the handling of the matrix $\mathbf{M}$ within $\hat{\mathbf{M}}$. The slack vectors are thus irrelevant for the actual minimization and can be neglected.

On the other hand, the constraint for the extended match-matrix entries

$$\hat{\mathrm{m}}_{jk} \in \{0,1\} \quad \forall\ j \in \{1,2,\ldots,J+1\}, k \in \{1,2,\ldots,K+1\} \tag{7.19}$$

adds the discrete part to the minimization. Interpreting now the entries $\hat{\mathrm{m}}_{jk}$ of the matrix $\hat{\mathbf{M}}$ as probabilities for a correspondence or non-correspondence, our match-matrix becomes a stochastic matrix $\bar{\mathbf{M}}$ by setting

$$\bar{\mathrm{m}}_{jk} \in [0,1] \quad \forall\ j \in \{1,2,\ldots,J+1\}, k \in \{1,2,\ldots,K+1\} \tag{7.20}$$

together with the summation constraints.[16] This gives a possibility to describe the problem in a continuous manner. The discrete case is then a special case of this, where the probability of two points being partners is either 1 or 0.

7.3.5. Mutual Approximation Updates

In the end, we are not interested in the correspondence probabilities of all different point pairs, but rather want to decide if a point of one set is a partner to a certain point of the other set. How can this decision be determined? Given some stochastic matrix $\bar{\mathbf{M}}$ with slacks, the probably

[15] Cf. Gold and Rangarajan [148, pp. 380–381].
[16] Such a matrix is also called doubly stochastic matrix (cf. Sinkhorn [381]) to make clear that the rows as well as the columns are normalized. Since we only deal with such doubly stochastic matrices, we refrain from this distinction.

most intuitive decision concerning the correspondence is to take the row (column) with the highest probability. This can then either be a special point if $j \in \{1, 2, \ldots, J\}$ ($k \in \{1, 2, \ldots, K\}$) or the lack of a partner if $j = J + 1$ ($k = K + 1$). Since these probabilities depend highly on the estimation of the pose for the problem, we are not interested in an inflexible, unilateral discrete solution to this as the pose is not known in advance either. What we do instead is illustrated in Fig. 7.11.

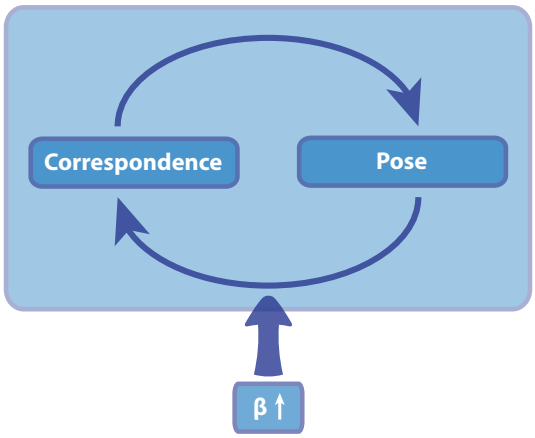


Fig. 7.11. **Mutual updates for correspondence and pose.** In each iteration of a step-wise approximation of the correct solutions, the correspondence problem is considered for a fixed pose (left). A consecutive update of the pose (right) is then again followed by a correspondence update while a confidence parameter β is increased. This mutual update process is repeated until convergence.

We update the approximation for the stochastic correspondence matrix $\bar{\mathbf{M}}$ with all the information of the pose we already have; this might even be none in the beginning. With this approximation we then calculate a new estimation for the pose which gives the input for the correspondence approximation again. The conceptual idea now is that the more loops we do, the more precise our estimation becomes. Thus, an early approximation is ranked with a low confidence β and the confidence of the approximation increases with the number of iterations.

How can we model this mathematically? Formally speaking, we look for a way to find a stochastic matrix $\bar{\mathbf{M}}$ that converges to the extended match-matrix $\hat{\mathbf{M}}$ if we increase some control parameter. We write formally

$$\bar{\mathbf{M}}(\beta) \overset{\beta \to \infty}{\longrightarrow} \hat{\mathbf{M}} \tag{7.21}$$

with the control parameter β that represents the **confidence level**.

As a next step, we **model both parts, the correspondence and the pose estimation within such an iteration loop separately**. Let us begin with the correspondence.

7.3.6. Correspondence Estimation

To model the correspondence approximation we have to find a formal way to describe both ideas, the assignment of probabilities to every point pair of our setting and the incorporation of the confidence level of the current estimate. In order to do this, we first look for a possibility to assign a certain positive value to every pair of points which we then normalize to get a probability.

The energy functional (7.5) can be summarized as

$$\mathrm{E} = \sum_{j,k} \mathrm{m}_{jk} \left(\mathrm{d}_{jk}^2 - \alpha \right) \tag{7.22}$$

with the distance error d_{jk} as defined in (7.10). If we differentiate this with respect to m_{jk}, we get

$$\mathrm{Q}_{jk} := \frac{\partial \mathrm{E}}{\partial \mathrm{m}_{jk}} \tag{7.23}$$

$$= \mathrm{d}_{jk}^2 - \alpha. \tag{7.24}$$

This gives small values $\mathrm{Q}_{jk} \in [-\alpha, 0]$ for point pairs within the toleration domain. Outside of this, the value $\mathrm{Q}_{jk} > 0$ increases as the error distance enlarges. As a start, we now assign strictly positive values to the represented point combinations according to the rank ordering of Q_{jk}. In addition, we scale the negative value of Q_{jk} with the parameter $\beta > 0$ which yields

$$\mathrm{q}_{jk} := \exp\left(-\beta \mathrm{Q}_{jk}\right) \tag{7.25}$$

$$= \exp\left(-\beta \left(\mathrm{d}_{jk}^2 - \alpha \right)\right). \tag{7.26}$$

The value of q_{jk} is small for non-corresponding points and big for corresponding pairs. If we normalize this by the sum of the row entries, for example, we get

$$\frac{\exp\left(-\beta \mathrm{Q}_{jk}\right)}{\sum_j \exp\left(-\beta \mathrm{Q}_{jk}\right)}, \tag{7.27}$$

which takes value 1 for the maximal value of the row as $\beta \to \infty$. All other values become 0 as $\beta \to \infty$. This method can be seen as the iterative counterpart for a maximization along the row by increasing the parameter β. It is therefore called **softmax** and was proposed by Bridle [40, pp. 212–213].[17] We use this idea for our problem although it is not without further ado transferable to the complete scenario. A **matrix normalization** in our case is not only desired across all rows, but at the same time across all columns. Fortunately, the entries of the matrix consisting of q_{jk} are all strictly positive and we can thus iteratively normalize the rows and the columns alternatingly to get a stochastic matrix with slacks that satisfies the normalization constraints (7.17) and (7.18).[18] With these ideas, we formulate Algorithm 7.1 for the update of our stochastic correspondence matrix $\bar{\mathbf{M}}$.

[17] While Bridle [40] introduced this term to machine learning, the function was already used by Boltzmann [33] and Gibbs [141].

[18] This is due to theorem 2 by Sinkhorn [381, p. 877] which guarantees the convergence to such a stochastic matrix. It is only formulated for square matrices. To use this result, we can embed our problem within a larger problem by

Algorithm 7.1. Update Correspondence Matrix with Alternating Normalization

Input parameters:

- Point sets: $X = \left\{\mathbf{x}_j \in \mathbb{R}^3 \mid 1 \leq j \leq J\right\}$, $Y = \left\{\mathbf{y}_k \in \mathbb{R}^3 \mid 1 \leq k \leq K\right\}$
- Pose parameters: $\mathbf{R}$, $\mathbf{t}$
- Variable Parameters: β
- Fixed Parameters: t_{mat}, ε, maximal number of loops I_n

Computation steps:

1. Initialize parameters: $I = 1$, $t_{norm} > t_{mat}$, $\bar{\mathbf{M}}_{prev} = \mathbf{0} \in \mathbb{R}^{(J+1)\times(K+1)}$
2. Prepare matrix with positive entries
   - $\mathrm{Q}_{jk} = \dfrac{\partial \mathrm{E}}{\partial \mathrm{m}_{jk}}$ (Equation (7.23))
3. Compute ordered positive values
   - $\bar{\mathrm{m}}_{jk} = \exp\left(-\beta \mathrm{Q}_{jk}\right) \quad \forall j \in \{1,2,\ldots,J\}, k \in \{1,2,\ldots,K\}$ (Equation (7.25))
4. Fill slack entries
   - $\bar{\mathrm{m}}_{jk} = 1 + \varepsilon \quad \forall j \in \{1,2,\ldots,J\}, k = K+1$
   - $\bar{\mathrm{m}}_{jk} = 1 + \varepsilon \quad \forall j = J+1, k \in \{1,2,\ldots,K\}$
5. Normalize matrix $\bar{\mathbf{M}}$

   while $t_{norm} > t_{mat}$ ***and*** $I < I_n$ **do**

   // Update $\bar{\mathbf{M}}$ by column normalization

   $$\bar{\mathrm{m}}'_{jk} = \frac{\bar{\mathrm{m}}_{jk}}{\sum_{k=1}^{K+1} \bar{\mathrm{m}}_{jk}} \quad \forall j \in \{1,2,\ldots,J\}$$

   // Update $\bar{\mathbf{M}}$ by row normalization

   $$\bar{\mathrm{m}}_{jk} = \frac{\bar{\mathrm{m}}'_{jk}}{\sum_{j=1}^{J+1} \bar{\mathrm{m}}'_{jk}} \quad \forall k \in \{1,2,\ldots,K\}$$

   // Update loop-break conditions

   Calculate matrix deviation: $t_{norm} = \left\|\bar{\mathbf{M}} - \bar{\mathbf{M}}_{prev}\right\|$

   Save previous correspondence matrix: $\bar{\mathbf{M}}_{prev} = \bar{\mathbf{M}}$

   Increment iteration counter: $I = I + 1$

Output:

- Normalized stochastic correspondence matrix $\bar{\mathbf{M}}$

The use of the methods softmax and the alternating normalization together is often referred to as **softassign**.[19] There are several publications that deal with an acceleration of such methods based on GPU programming. Tamaki et al. [402] present a general CUDA-based implementation for point set registration and Slomp et al. [382] investigate a GPU-based method for the application in photomosaics. The method itself has been improved for very large problems with the help of spectral graph theory by Lozano et al. [266]. Beyond that, many extensions are made for applications within protein structure analysis by Jain et al. [192] and shape fitting in medical imaging tasks by Rangarajan et al. [342]. These approaches combine the rudimental ideas of softassign with other thoughts.

For our further research, we focus on the concept of mutual approximation and look at the processing chain illustrated in Fig. 7.12 where these ideas are part of Algorithm 7.1 represented by the left box.
One advantage of this method is the fact that by incrementing β only after every new pose estimation, the chance of converging to a local minimum becomes smaller. This justifies the prior translation of the discrete part of the problem into a continuous environment.

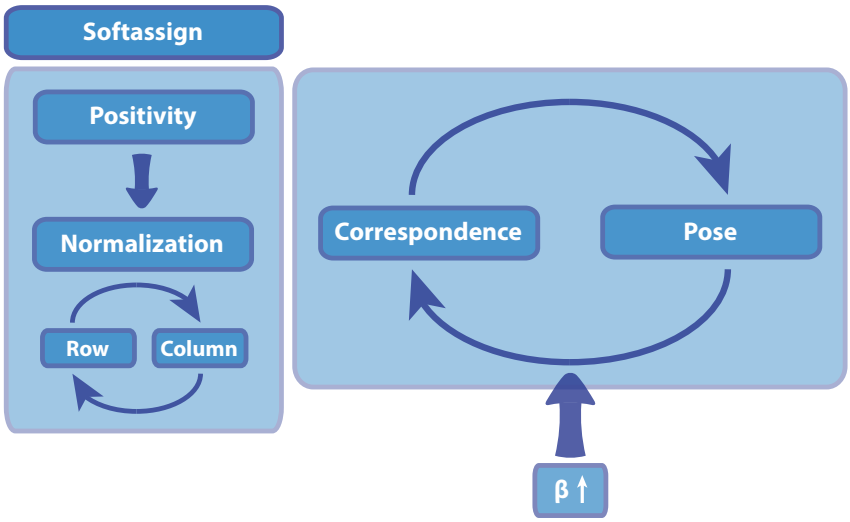


Fig. 7.12. **Update of correspondence matrix with softassign.** In the bidirectional update process for correspondence and pose (right) as shown in Fig. 7.11, the correspondence is updated via softassign (left). A match matrix with positive entries is initialized by partial derivation of the energy functional and adding slacks. Mutual normalization respecting the rows and columns yield convergence to a stochastic matrix.

So far, we analysed the correspondence side in detail. Let us now focus on the other component of the approximation: the pose estimation.

extending the smaller dimension of the stochastic matrix. Choosing all suchlike created entries to be zero except for the slacks then makes the theorem applicable in our case and we can forget about the added entries afterwards.
[19] Cf. Gold et al. [149, pp. 1022–1024].

7.3.7. Pose Estimation

How is it possible to approximate a pose given two point sets and their correspondences? Let us write down once again the energy functional from (7.5):

$$\mathrm{E}(\mathbf{R},\mathbf{t}) = \sum_{j=1}^{J}\sum_{k=1}^{K} \mathrm{m}_{jk} \left\| \mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t}\right)\right\|^2 - \alpha \sum_{j=1}^{J}\sum_{k=1}^{K} \mathrm{m}_{jk}. \tag{7.28}$$

This time, E only depends on the six parameters given by $\mathbf{R}$ and $\mathbf{t}$ since the current correspondence estimation from section 7.3.6 fixes the entries of $\mathbf{M}$ for the moment. Thus we can write

$$\mathrm{E}(\mathbf{R},\mathbf{t}) = \sum_{j,k} \mathrm{m}_{jk} \left\| \mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t}\right)\right\|^2 + C \tag{7.29}$$

with the constant $C \in \mathbb{R}$. Since we are still interested in minimizing this term for the arguments $\mathbf{R}$ and $\mathbf{t}$, we neglect the scalar C which does not change the solution. One possible way to approximate the parameters we look for would thus be to use an algorithm that minimizes this least squares problem. A potential candidate is given by the well-studied Levenberg-Marquardt algorithm. Although we only have 6 DoF within the equation, the runtime is critical and we therefore do not want to use such an iterative procedure in every loop. In the following, we use the theory of **(dual) quaternions** (cf. sections 6.2.3 and 6.2.5) to formulate the minimization of the energy functional as an eigenvalue problem.

We use a unit dual quaternion to represent a rotation $\mathbf{R} \in \mathbb{R}^{3\times3}$ and a translation $\mathbf{t} \in \mathbb{R}^3$. To ease the notation, we write $\mathbf{1} = (1,0,0,0)^\mathrm{T}$, $\mathbf{i} = (0,1,0,0)^\mathrm{T}$, $\mathbf{j} = (0,0,1,0)^\mathrm{T}$, and $\mathbf{k} = (0,0,0,1)^\mathrm{T}$ as basis elements of $\mathbb{H}$ to represent quaternions in vector notation. Following the ideas presented by Walker et al. [440, pp. 361–362], we can derive from the dual quaternion $\mathbf{Q} = \mathbf{r} + \varepsilon\mathbf{s} \in \mathbb{DH}$ the representation

$$\begin{pmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0}^\mathrm{T} & 1 \end{pmatrix} = \mathbf{W}(\mathbf{r})^\mathrm{T}\,\mathbf{P}(\mathbf{r}) \quad \text{and} \quad \begin{pmatrix} \mathbf{t} \\ 0 \end{pmatrix} = \mathbf{W}(\mathbf{r})^\mathrm{T}\,\mathbf{s} \tag{7.30}$$

with the two matrices that arise from the quaternion $\mathbf{r} \in \mathbb{H}$ with $\mathbf{r} = \left(r_1, r_2, r_3, r_4\right)^\mathrm{T}$ as

$$\mathbf{P}(\mathbf{r}) = \begin{pmatrix} r_4 & -r_3 & r_2 & r_1 \\ r_3 & r_4 & -r_1 & r_2 \\ -r_2 & r_1 & r_4 & r_3 \\ -r_1 & -r_2 & -r_3 & r_4 \end{pmatrix} \quad \text{and} \quad \mathbf{W}(\mathbf{r}) = \begin{pmatrix} r_4 & r_3 & -r_2 & r_1 \\ -r_3 & r_4 & r_1 & r_2 \\ r_2 & -r_1 & r_4 & r_3 \\ -r_1 & -r_2 & -r_3 & r_4 \end{pmatrix}. \tag{7.31}$$

If we write a point quaternion for the point $\mathbf{p} \in \mathbb{R}^3$ as

$$p = \begin{pmatrix} \mathbf{p} \\ 0 \end{pmatrix}, \tag{7.32}$$

we can thus reformulate our energy functional with the point quaternions $\boldsymbol{x}_j$ and $\boldsymbol{y}_k$ representing the points $\mathbf{x}_j$ and $\mathbf{y}_k$. It holds

$$\mathrm{E}(\mathbf{r},\mathbf{s}) = \sum_{j,k} \mathrm{m}_{jk} \left\| \boldsymbol{y}_k - \left(\mathbf{W}(\mathbf{r})^{\mathrm{T}} \mathbf{P}(\mathbf{r}) \boldsymbol{x}_j + \mathbf{W}(\mathbf{r})^{\mathrm{T}} \mathbf{s} \right) \right\|^2 + C \tag{7.33}$$

$$\overset{\substack{\text{Appendix} \\ A.1}}{=} \sum_{j,k} \mathrm{m}_{jk} \left(\mathbf{s}^{\mathrm{T}}\mathbf{s} - 2\mathbf{r}^{\mathrm{T}} \mathbf{P}\left(\boldsymbol{y}_k\right)^{\mathrm{T}} \mathbf{W}\left(\boldsymbol{x}_j\right) \mathbf{r} + 2\mathbf{s}^{\mathrm{T}} \left(\mathbf{W}\left(\boldsymbol{x}_j\right) - \mathbf{P}\left(\boldsymbol{y}_k\right) \right) \mathbf{r} \right) + C + D \tag{7.34}$$

with a constant $C + D \in \mathbb{R}$ that can be neglected for the minimization. The individual steps that yield this result can be found in appendix A.1. Without changing the minimum, we set the constant to 0 and dividing by 2. We then rewrite the energy functional as a quadratic function in $\mathbf{r}$ and $\mathbf{s}$ with

$$\mathrm{E}(\mathbf{r},\mathbf{s}) = \mathbf{r}^{\mathrm{T}} \underbrace{\left(-\sum_{j,k} \mathrm{m}_{jk} \mathbf{P}\left(\boldsymbol{y}_k\right)^{\mathrm{T}} \mathbf{W}\left(\boldsymbol{x}_j\right) \right)}_{\mathbf{C}_1} \mathbf{r} + \mathbf{s}^{\mathrm{T}} \underbrace{\left(\frac{1}{2} \sum_{j,k} \mathrm{m}_{jk}\, \mathbf{I} \right)}_{\mathbf{C}_2} \mathbf{s} \tag{7.35}$$

$$+ \mathbf{s}^{\mathrm{T}} \underbrace{\left(\sum_{j,k} \mathrm{m}_{jk} \left(\mathbf{W}\left(\boldsymbol{x}_j\right) - \mathbf{P}\left(\boldsymbol{y}_k\right) \right) \right)}_{\mathbf{C}_3} \mathbf{r} \tag{7.36}$$

$$= \mathbf{r}^{\mathrm{T}} \mathbf{C}_1 \mathbf{r} + \mathbf{s}^{\mathrm{T}} \mathbf{C}_2 \mathbf{s} + \mathbf{s}^{\mathrm{T}} \mathbf{C}_3 \mathbf{r}. \tag{7.37}$$

If we want to minimize this with respect to the unit constraints (6.29) and (6.30), we can add Lagrange multipliers and get

$$\mathrm{E}(\mathbf{r},\mathbf{s}) = \mathbf{r}^{\mathrm{T}} \mathbf{C}_1 \mathbf{r} + \mathbf{s}^{\mathrm{T}} \mathbf{C}_2 \mathbf{s} + \mathbf{s}^{\mathrm{T}} \mathbf{C}_3 \mathbf{r} + \lambda_1 \left(\mathbf{r}^{\mathrm{T}} \mathbf{r} - 1 \right) + \lambda_2 \left(\mathbf{r}^{\mathrm{T}} \mathbf{s} \right). \tag{7.38}$$

The necessary condition for the minimization then reads

$$\nabla \mathrm{E} = (\partial_{\mathbf{r}} \mathrm{E}, \partial_{\mathbf{s}} \mathrm{E})^{\mathrm{T}} \overset{!}{=} \mathbf{0}. \tag{7.39}$$

The partial derivatives can be calculated as

$$\partial_{\mathbf{r}} \mathrm{E} = \left(\mathbf{C}_1 + \mathbf{C}_1^{\mathrm{T}} \right) \mathbf{r} + \mathbf{C}_3^{\mathrm{T}} \mathbf{s} + 2\lambda_1 \mathbf{r} + \lambda_2 \mathbf{s} \overset{!}{=} \mathbf{0} \tag{7.40}$$

and

$$\partial_{\mathbf{s}} \mathrm{E} = \left(\mathbf{C}_2 + \mathbf{C}_2^{\mathrm{T}} \right) \mathbf{s} + \mathbf{C}_3 \mathbf{r} + \lambda_2 \mathbf{r} \overset{!}{=} \mathbf{0} \tag{7.41}$$

$$\mathbf{s}^{\mathrm{T}} \left(\mathbf{C}_2 + \mathbf{C}_2^{\mathrm{T}} \right) + \mathbf{r}^{\mathrm{T}} \mathbf{C}_3^{\mathrm{T}} + \lambda_2 \mathbf{r}^{\mathrm{T}} = \mathbf{0}^{\mathrm{T}}. \tag{7.42}$$

Multiplying equation (7.42) with $\mathbf{r}$ from the right gives

$$\lambda_2 = -\mathbf{r}^{\mathrm{T}} \mathbf{C}_3^{\mathrm{T}} \mathbf{r} = \mathbf{r}^{\mathrm{T}} \mathbf{C}_3 \mathbf{r} = 0 \tag{7.43}$$

where the second step is due to the fact that $\mathbf{C}_3$ – as the addition of two skew symmetric matrices – is skew symmetric itself. If we insert this into (7.41), we get the following due to the diagonality of $\mathbf{C}_2$:

$$\left(\mathbf{C}_2 + \mathbf{C}_2^{\mathrm{T}}\right)\mathbf{s} = -\mathbf{C}_3\mathbf{r} \tag{7.44}$$

$$\mathbf{s} = -\left(\mathbf{C}_2 + \mathbf{C}_2^{\mathrm{T}}\right)^{-1}\mathbf{C}_3\mathbf{r} = -(2\mathbf{C}_2)^{-1}\mathbf{C}_3\mathbf{r} = -\frac{1}{2}\mathbf{C}_2^{-1}\mathbf{C}_3\mathbf{r}. \tag{7.45}$$

Substituting this into equation (7.40), we end up with

$$\left(\mathbf{C}_1 + \mathbf{C}_1^{\mathrm{T}}\right)\mathbf{r} - \frac{1}{2}\mathbf{C}_3^{\mathrm{T}}\mathbf{C}_2^{-1}\mathbf{C}_3\mathbf{r} = -2\lambda_1\mathbf{r} \tag{7.46}$$

$$\underbrace{\frac{1}{2}\left(\frac{1}{2}\mathbf{C}_3^{\mathrm{T}}\mathbf{C}_2^{-1}\mathbf{C}_3 - \left(\mathbf{C}_1 + \mathbf{C}_1^{\mathrm{T}}\right)\right)}_{\mathbf{A}}\mathbf{r} = \lambda_1\mathbf{r} \tag{7.47}$$

$$\mathbf{A}\mathbf{r} = \lambda_1\mathbf{r}. \tag{7.48}$$

Due to the properties of $\mathbf{P}\left(\boldsymbol{y}_k\right)$ and $\mathbf{W}\left(\boldsymbol{x}_j\right)$ from $\mathbf{C}_1$, we can simplify $\mathbf{A}$ even more.[20]

$$\mathbf{A} = \frac{1}{2}\left(\frac{1}{2}\mathbf{C}_3^{\mathrm{T}}\mathbf{C}_2^{-1}\mathbf{C}_3 - 2\mathbf{C}_1\right) \tag{7.49}$$

$$= \frac{1}{4}\mathbf{C}_3^{\mathrm{T}}\mathbf{C}_2^{-1}\mathbf{C}_3 - \mathbf{C}_1. \tag{7.50}$$

The equation $\mathbf{A}\mathbf{r} = \lambda_1\mathbf{r}$ describes the **eigenvalue problem** of the symmetric matrix $\mathbf{A} \in \mathbb{R}^{4\times4}$. Which of the four possible solutions now minimizes our energy functional? For this to see, we multiply equation (7.40) by $\frac{1}{2}$ which gives

$$-\frac{1}{2}\mathbf{C}_3^{\mathrm{T}}\mathbf{s} - \lambda_1\mathbf{r} = \frac{1}{2}\left(\mathbf{C}_1 + \mathbf{C}_1^{\mathrm{T}}\right)\mathbf{r} \tag{7.51}$$

$$-\frac{1}{2}\mathbf{s}^{\mathrm{T}}\mathbf{C}_3 - \lambda_1\mathbf{r}^{\mathrm{T}} = \frac{1}{2}\mathbf{r}^{\mathrm{T}}\left(\mathbf{C}_1 + \mathbf{C}_1^{\mathrm{T}}\right). \tag{7.52}$$

Multiplication with $\mathbf{r}$ from the right yields

$$-\frac{1}{2}\mathbf{s}^{\mathrm{T}}\mathbf{C}_3\mathbf{r} - \lambda_1 = \frac{1}{2}\mathbf{r}^{\mathrm{T}}(2\mathbf{C}_1)\mathbf{r} = \mathbf{r}^{\mathrm{T}}\mathbf{C}_1\mathbf{r}. \tag{7.53}$$

On the other hand, multiplying equation (7.41) by $\frac{1}{2}\mathbf{s}^{\mathrm{T}}$ gives

$$-\frac{1}{2}\mathbf{s}^{\mathrm{T}}\mathbf{C}_3\mathbf{r} = \frac{1}{2}\mathbf{s}^{\mathrm{T}}\left(\mathbf{C}_2 + \mathbf{C}_2^{\mathrm{T}}\right)\mathbf{s} = \frac{1}{2}\mathbf{s}^{\mathrm{T}}(2\mathbf{C}_2)\mathbf{s} = \mathbf{s}^{\mathrm{T}}\mathbf{C}_2\mathbf{s} \tag{7.54}$$

Inserting now (7.53) and (7.54) into the energy functional (7.37) we get

$$\mathrm{E}(\mathbf{r},\mathbf{s}) = \mathbf{r}^{\mathrm{T}}\mathbf{C}_1\mathbf{r} + \mathbf{s}^{\mathrm{T}}\mathbf{C}_2\mathbf{s} + \mathbf{s}^{\mathrm{T}}\mathbf{C}_3\mathbf{r} \tag{7.55}$$

$$= -\frac{1}{2}\mathbf{s}^{\mathrm{T}}\mathbf{C}_3\mathbf{r} - \lambda_1 - \frac{1}{2}\mathbf{s}^{\mathrm{T}}\mathbf{C}_3\mathbf{r} + \mathbf{s}^{\mathrm{T}}\mathbf{C}_3\mathbf{r} \tag{7.56}$$

$$= -\lambda_1, \tag{7.57}$$

which is minimal for the maximal eigenvalue λ_1 of $\mathbf{A}$.

[20] For the properties of the matrices $\mathbf{P}$ and $\mathbf{W}$, see appendix A.1.

To find the **dominant eigenvector** corresponding to the largest eigenvalue, we do not have to calculate all eigenvalues separately. Since the matrix $\mathbf{A}$ is real and symmetric, the eigenvectors are orthogonal. For a speed-up of this calculation, we can therefore use a **power iteration**.[21] For a non-degenerated start quaternion $\bar{\mathbf{r}}_0$, the sequence

$$\bar{\mathbf{r}}_{n+1} = \frac{\mathbf{A}\bar{\mathbf{r}}_n}{\|\mathbf{A}\bar{\mathbf{r}}_n\|} \overset{n\to\infty}{\longrightarrow} \mathbf{r}_{max} \tag{7.58}$$

converges to this normalized dominant eigenvector. Finally we can determine the dual quaternion part $\mathbf{s}_{max}$ from $\mathbf{r}_{max}$ with equation (7.45). A pose in terms of a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$ is given by the relation equation of quaternions and homogeneous transformations specified in (7.30).
Let us now summarize these steps in Algorithm 7.2 for the pose estimation.

As a last step, we formally fuse both ideas, the correspondence estimation from section 7.3.6 and the quaternion-based pose estimation from section 7.3.7.

7.3.8. Fusion of Approximations

The initial idea in section 7.3.5 to tackle the problem was to **mutually update both the point correspondences and the pose while increasing the confidence level**. Schematically, the entire process can be illustrated as shown in Fig. 7.13 where we see the assignment estimation on the left and the pose approximation on the right.

With the help of the already formulated algorithms for both sides, we write Algorithm 7.3 to join the two parts.[22] The idea for the confidence update is to choose an exponentially increasing step size. The values for β_0, β_{inc}, and β_{max} can be determined empirically. We use $\beta_0 = 10^{-4}$, $\beta_{inc} = 1.053$, and $\beta_{max} = 10^3$ in all our experiments. In this way, we have a low confidence level in the beginning and the confidence level increases with acceleration the more iterations we perform. In our tests, we experience softassign convergence usually within less than 5 loops and thus set a relaxed iteration maximum of $I_{max} = 10$.

Our next step is to combine these thoughts with the concepts of previous chapters. We focus in particular on testing and evaluation of Algorithm 7.3 and analyse its performance for real-time 3D tracking applications.

[21] Cf. Deuflhard and Hohmann [87, pp. 138–139].
[22] A two way approximation with an increasing control parameter is also proposed by Gold et al. [149, p. 1026].

Algorithm 7.2. Update Pose with Quaternion Method

Input parameters:

- Point sets: $X = \left\{\mathbf{x}_j \in \mathbb{R}^3 \mid 1 \le j \le J\right\}$, $Y = \left\{\mathbf{y}_k \in \mathbb{R}^3 \mid 1 \le k \le K\right\}$.
- Normalized correspondence matrix $\bar{\mathbf{M}}$ from Algorithm 7.1
- Fixed Parameters: t_{pow}, maximal number of loops n_{max}

Computation steps:

1. Initialize iteration counter: $n = 1$
2. Quaternion initialization: $\bar{\mathbf{r}}_1 = \begin{cases} \mathbf{r}_{max} & \text{from last run} \\ (1,1,1,1)^{\mathrm{T}} & \text{for first run} \end{cases}$
3. Generate quaternion representation of points (Equation (7.32))
   - $\boldsymbol{x}_j = \begin{pmatrix} \mathbf{x}_j \\ 0 \end{pmatrix}, \boldsymbol{y}_k = \begin{pmatrix} \mathbf{y}_k \\ 0 \end{pmatrix} \quad \forall j \in \{1,2,\ldots,J\}, k \in \{1,2,\ldots,K\}$
4. Compute matrices with quaternion matrices (Equation (7.31))
   - $\mathbf{C}_1 = -\sum_{j,k} \mathrm{m}_{jk} \mathbf{P}\left(\boldsymbol{y}_k\right)^{\mathrm{T}} \mathbf{W}\left(\boldsymbol{x}_j\right)$
   - $\mathbf{C}_2 = \frac{1}{2} \sum_{j,k} \mathrm{m}_{jk}\, \mathbf{I}$
   - $\mathbf{C}_3 = \sum_{j,k} \mathrm{m}_{jk} \left(\mathbf{W}\left(\boldsymbol{x}_j\right) - \mathbf{P}\left(\boldsymbol{y}_k\right)\right)$
   - $\mathbf{A} = \frac{1}{4}\mathbf{C}_3^{\mathrm{T}}\mathbf{C}_2^{-1}\mathbf{C}_3 - \mathbf{C}_1$ (Equation (7.50))
5. Solve eigenvalue problem with power iteration (Equation (7.58))

   while $\|\bar{\boldsymbol{r}}_n - \bar{\boldsymbol{r}}_{n-1}\| > t_{pow}$ ***and*** $n < n_{max}$ ***or*** $n = 1$ **do**

   $\bar{\mathbf{r}}_{n+1} = \dfrac{\mathbf{A}\bar{\mathbf{r}}_n}{\|\mathbf{A}\bar{\mathbf{r}}_n\|}$

   Increment iteration counter: $n = n + 1$
6. Save dominant eigenvector: $\mathbf{r}_{max} = \bar{\mathbf{r}}_n$
7. Calculate dual quaternion: $\mathbf{s}_{max} = -\frac{1}{2}\mathbf{C}_2^{-1}\mathbf{C}_3\mathbf{r}_{max}$ (Equation (7.45))
8. Compute rotation and translation (Equation (7.30))
   - $\mathbf{R} = \mathbf{W}(\mathbf{r}_{max})^{\mathrm{T}}\,\mathbf{P}(\mathbf{r}_{max})[1\!:\!3, 1\!:\!3]$
   - $\mathbf{t} = \mathbf{W}(\mathbf{r}_{max})^{\mathrm{T}}\,\mathbf{s}_{max}[1\!:\!3]$

Output:

- Pose parameters: $\mathbf{R}$, $\mathbf{t}$

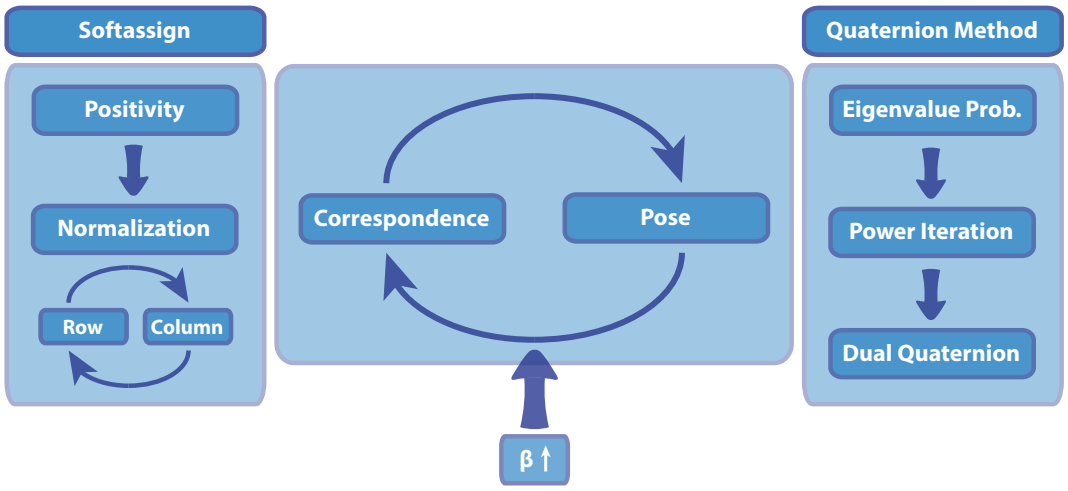


Fig. 7.13. **Incremental fusion of correspondence and pose estimation.** Correspondence and pose are updated as shown in the middle while the confidence parameter β increases. The left side illustrates the correspondence step via softassign as shown in Fig. 7.12. The right side shows the pose update with the dual quaternion method. The eigenvalue problem arising from equation (7.48) is solved with power iteration (7.58) for the dual quaternion representing the current estimate for the rigid transformation.

Algorithm 7.3. Fast and Robust Correspondence and Pose Estimation

---

Input parameters:

- Point sets: $X = \{\mathbf{x}_j \in \mathbb{R}^3 \mid 1 \le j \le J\}$, $Y = \{\mathbf{y}_k \in \mathbb{R}^3 \mid 1 \le k \le K\}$
- Pose parameter guess: $\mathbf{R}$, $\mathbf{t}$
- Scalar parameters: β_0, β_{inc}, β_{max}, t_{soft}
- Iteration parameter: maximal number of loops I_{max}

Computation steps:

1. Initialize iteration counter: $I = 1$
2. Initialize confidence parameter: $\beta = \beta_0$
3. Initialize correspondence matrix: $\bar{\mathbf{M}}_{prev} = \mathbf{0} \in \mathbb{R}^{(J+1)\times(K+1)}$

   while $\beta < \beta_{max}$ **do**
   - **while** $t_{norm} > t_{soft}$ ***and*** $I < I_{max}$ ***or*** $I = 1$ **do**
     - $\bar{\mathbf{M}} \leftarrow$ Update **correspondence matrix** (Algorithm 7.1)
     - $\mathbf{R}$, $\mathbf{t} \leftarrow$ Update **pose** (Algorithm 7.2)
     - // Update loop-break conditions
     - Calculate matrix deviation: $t_{norm} = \left\|\bar{\mathbf{M}} - \bar{\mathbf{M}}_{prev}\right\|$
     - Save previous correspondence matrix: $\bar{\mathbf{M}}_{prev} = \bar{\mathbf{M}}$
     - Increment iteration counter: $I = I + 1$
   - Increment $\beta = \beta\,\beta_{inc}$

4. $\mathbf{M} \leftarrow$ Normalized stochastic correspondence matrix $\bar{\mathbf{M}}$

Output:

- Correspondence matrix $\mathbf{M}$
- Pose parameters: $\mathbf{R}$, $\mathbf{t}$

7.4. Evaluation and Testing

For test purposes, Algorithm 7.3 has been implemented in C++. All tests are made on the same machine with an Intel Core i5-3320M Processor and 8 GB RAM running Windows 7 Professional, 64-bit. The rotation matrix is initialized as the identity and the initial translation vector becomes the vector between the centres of mass of the two clouds. For consecutive fits we use the last pose as an initialization for the rotation matrix and the translation vector.

7.4.1. Accuracy

To test the accuracy of our fit, we neglect the measurement error and calculate the **root mean square (RMS)** of the fitting residuum. This is given for one fit of the clouds $X = \{\mathbf{x}_j \in \mathbb{R}^3 \mid 1 \leq j \leq J\}$ and $Y = \{\mathbf{y}_k \in \mathbb{R}^3 \mid 1 \leq k \leq K\}$ by

$$e_{RMS} := \sqrt{\frac{\sum_{k=1}^{K} \mathrm{m}_{jk} \left(\mathbf{y}_k - \left(\mathbf{R}\mathbf{x}_j + \mathbf{t}\right)\right)^2}{\sum_{k=1}^{K} \mathrm{m}_{jk}}} = \sqrt{\frac{\sum_{\mathbf{x}_j \leftrightarrow \mathbf{y}_k} \mathrm{d}_{jk}^2}{m}} \tag{7.59}$$

with the distance error d_{jk} and the number of matches m that arise from $\mathrm{m}_{jk} \in \{0, 1\}$.

The test scenario is the turning table from OTS prototype design v.1 with the object shown in the demo image (cf. Fig. 7.4) equipped with 6 markers. The distance from the table to the camera system normal to its surface is ca. 70 cm and we use markers with a diameter of 5 mm. We build a test model as described in section 7.5.1 and evaluate a randomly chosen sequence of 100 frames during the movement. The result is shown in Fig. 7.14. The mean error is 22.54 μm with a standard deviation of 11.68 μm. The median is 20.47 μm. We observe 8 outlier measurements during our test. Upon inspection of the calculated point clouds, all of these turn out to include fewer than 6 triangulations due to ambient illumination or part occlusions. The worst three results include only 4 triangulations with the biggest deviation of 88.39 μm which is more accurate than the range of custom pointers tracked by commercial multi-camera medical trackers[23] and below 0.1 mm which is a negligible error for most medical applications[24]. This paves the way to use cases of the proposed system.

Other commercial systems are evaluated in a comparison study of Elfring et al. [96] with a coordinate measurement machine at a distance closer to the camera. The results are summarized in Table 7.1 in terms of RMS error.

[23] Cf. Marinetto et al. [276].
[24] Cf. Elfring, Fuente, and Radermacher [96].

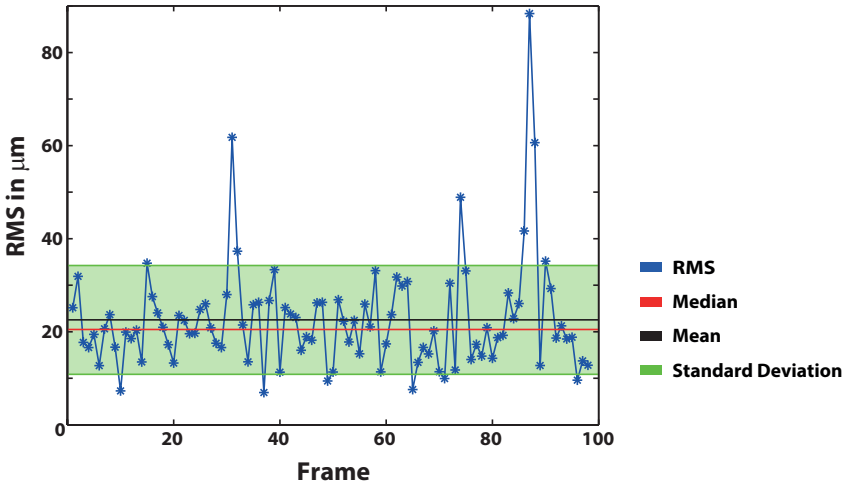


Fig. 7.14. RMS error of OTS. The RMS error (blue) is shown for OTS design v.1 with a 6 markers test object using 5 *mm* diameter markers at a distance of ca. 70 *cm* on a turning table. A random movement is recorded for 100 frames. Additional illustrations show the median (red) and mean value (black) as well as the standard deviation region (green) for the test data.

Optical Tracking System	RMS Error
NDI Polaris P4	0.381 *mm*
NDI Polaris Spectra (passive markers)	0.165 *mm*
NDI Polaris Spectra (active markers)	0.104 *mm*
Stryker Navigation System II	0.077 *mm*

Tab. 7.1. Comparison of commercial optical tracking systems. The evaluation done by Elfring et al. [96] uses clinical pointers at a distance of up to 40 *cm*.

7.4.2. Runtime

The identical setting of section 7.4.1 is used to test the computational efficiency of our approach. The calculation time is shown in Fig. 7.15.

It turns out that the mean calculation time is 5.89 *ms* with a standard deviation of ±5.90 *ms*. The median for the tracking calculation time of 6 points is 2.32 *ms*. For the evaluation, every calculation has been performed 20 times and the mean calculation time is shown here. We discovered that the increased runtime demand for some of the frames depends on the lack of one or several points whereas the calculation times for all the fittings with a recognition of equally many points lie closely around the median with a maximal time deficit of 0.73 *ms* compared to the median. It is worth to note that the runtime stays below 25 *ms* in all cases which allows for **seamless integration into real-time pipelines**.

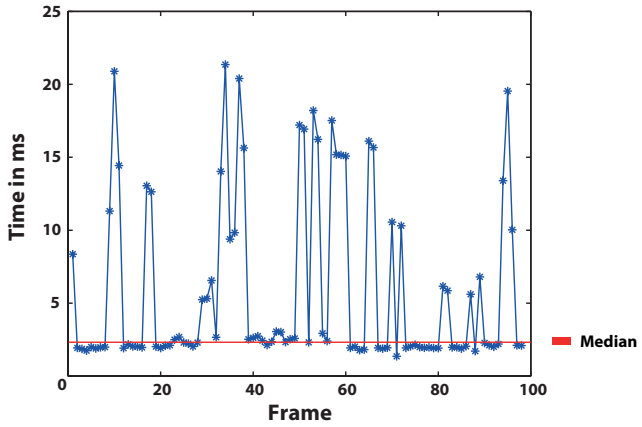


Fig. 7.15. **Fitting algorithm efficiency.** The graph shows the calculation time needed to track an object with 6 markers on a turning table sequence. While all fits with measurements of 6 markers distribute tightly around the median runtime of 2.32 *ms* (the close points), missing triangulations with clouds of as little as 4 points (see the outliers) significantly increase the runtime of the algorithm. Even in these cases, a total runtime of under 25 *ms* still allows for lag-free real-time applications.

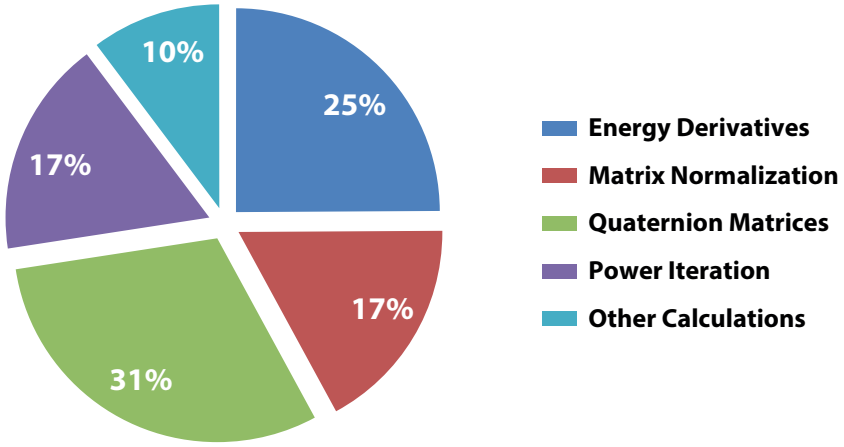


Fig. 7.16. **Relative runtime of fitting algorithm subroutines.** Shown is the computation time for core parts of our fitting algorithm. The calculation of the quaternion matrices takes with 31% the longest. Deriving the energy functional costs 25% followed by the row-column normalization and the power iteration to solve the eigenvalue problem, which both take 17% of the total time. All other processing sums up to the remaining 10%.

The **relative calculation time** of the individual processing components of Algorithm 7.3 thereby distributes as shown in Fig. 7.16. For this evaluation, we measure the total calculation time of every subroutine of the algorithm. If a part runs more than once, we note the total runtime by summation of the individual runtime of all its calls.
It is noticeable, that for both procedures – the correspondence estimation and the pose approximation – the inner loops are in total less time demanding than their precalculation steps. Thus, the calculation of the quaternion matrices takes longer than the actual quaternion estimation by power iteration and the calculation of the partial derivatives of the energy functional is computationally more complex than the alternating row-column matrix normalization.

Let us now analyse the **runtime limitations** of Algorithm 7.3 by running a pose estimation for a point cloud of increasing size. For the evaluation, we artificially create a random point set of increasing cardinality and transform the set with a random pose. The pose consists of a translation vector $\mathbf{t} \in [-5, 5]^3$ and a rotation matrix with Euler-angles α, β, and $\gamma \in [-0.1, 0.1]$. $10 \pm 5\%$ of the points from both sets are deleted and Gaussian noise is added. The whole fitting process is done 10 times for every cloud size taking the identity matrix and the vector $\mathbf{0} \in \mathbb{R}^3$ as an initial guess for the rotation and translation of the pose. The results are shown in Fig. 7.17 where we add a plausibility check by calculation of the RMS error as shown on the right. The residuum is consistent with the artificially inserted noise.

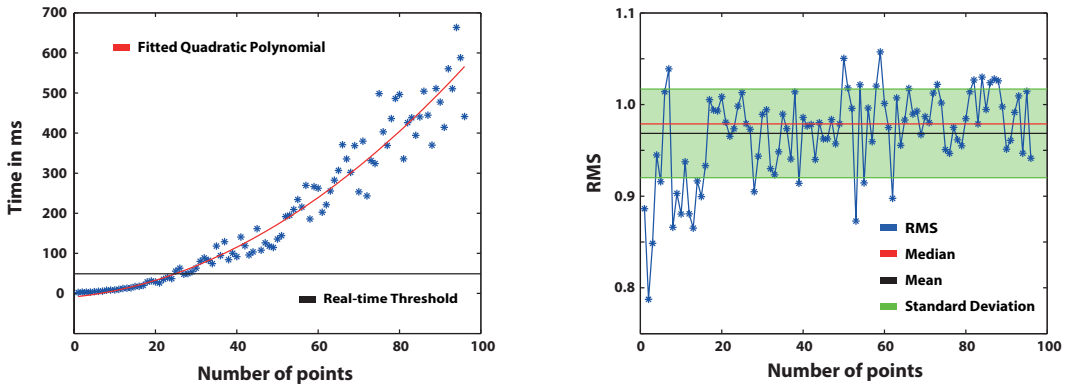


Fig. 7.17. **Time and RMS for pose estimation with different cloud sizes.** Random 3D point sets of increasing cardinality are created and transformed by random poses with added Gaussian noise on the point coordinates. The time for Algorithm 7.3 shown in blue on the left depends quadratically on the number of points (see fitted red polynomial). The RMS error on the right (blue) as well as statistical measures such as median (red), mean (black) and standard deviation (green) are calculated in relation to the number of points. The residuals are consistent with the artificially added noise.

The mean calculation times are recorded and a quadratic polynomial is fitted through the data. This suggests that the **de facto computation complexity** of the fitting algorithm is $\mathcal{O}(n^2)$, where n is the number of points.
We note that without a specific guess for the pose, the given algorithm is theoretically able to track up to 24 points in real time at a frame rate of above 20 fps. This gives us a maximal cloud size for real-time calculation. Larger clouds can still be calculated in real-time with the algorithm, however, the set needs to be split (cf. section 7.5). It is also worth to mention, that the convergence rate eventually increases through better initialization. Nevertheless, further steps such as the single view processing and its image coordinate extraction (cf. Algorithm 3.5) also require computation time, although we empirically determined the fitting procedure as the bottle neck of the approach.

7.4.3. Robustness

In order to evaluate the robustness of the algorithm, we again use simulated data. **Increasing** the **level of noise** for a point cloud with 20 elements from 0.0 *mm* to 0.8 *mm* gives the results shown in Fig. 7.18. The RMS error also increases linearly in the same range as expected while it can be seen that the noise level affects the runtime negatively.

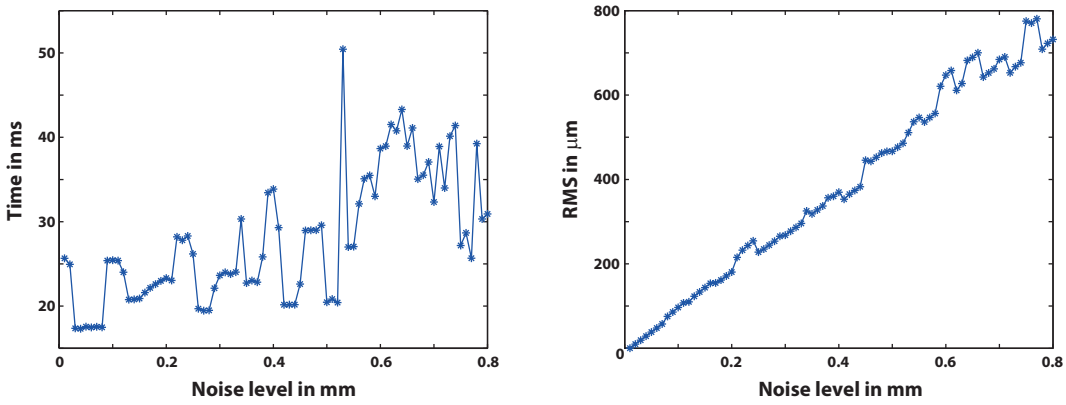


Fig. 7.18. **Impact of noise on fitting algorithm.** Synthetic point sets with increasing noise levels from 0.0 *mm* to 0.8 *mm* are used to test the robustness of Algorithm 7.3. While the accuracy of the estimated pose (right) decreases linearly with the level of noise, the runtime (left) is also increasing.

In a second test, we evaluate the **sensitivity** of the algorithm **to missing points**. For a fixed noise level of 0.1 *mm* standard deviation, we increase the part of deleted points in the target set Y linearly from 0 to 80 points for a configuration of 100 points in total. This has a decisive influence on the computation time as depicted in Fig. 7.19 on the right, since the target set becomes smaller.
The deletion of points together with an improper initial pose given by the identity matrix and the translation vector $\mathbf{0} \in \mathbb{R}^3$ cause the algorithm to fail several times in this test as we can see in the figure on the left. At these points, the minimization of the energy functional converges to a local minimum instead of the global one. This happens mostly if the amount of points within the two sets X and Y differs clearly and local minima become more probable. Using a suitable initial guess and a fitting approach with appropriate subsets as described in section 7.5 solves this issue.

We conclude this evaluation and note that the proposed pose estimation algorithm shows robustness with respect to inter point set motion while maintaining adequate calculation performance. Moreover, missing points and partial occlusion can be handled to some extend where predictions in scenarios with more than 50% occlusion have to be treated with caution. The calculated pose estimates closely reflect the measurement error and are accurate in practice in a range of 0.02 ± 0.01 *mm* at a realistic working distance suitable for applications in medical as well as industrial setups. The algorithm is capable of real-time execution and measurements in small point clouds. However, for clouds of significantly more than 20 points, the calculation time may become critical. An extension for very large point clouds will be described hereafter in section 7.5.

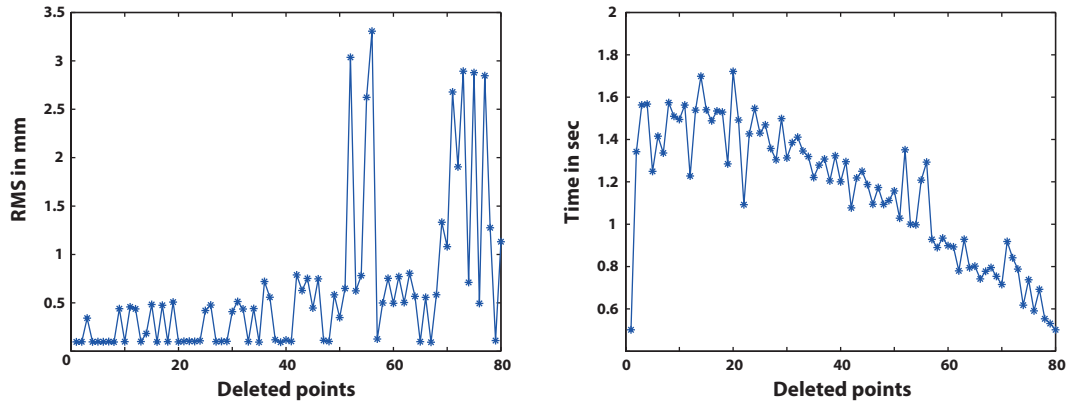


Fig. 7.19. **Impact of missing points on fitting algorithm.** A synthetically created point set of 100 points is taken as source set, while Gaussian noise is added to a repositioned copy of it, the target, while an increasing amount of points are dropped. A fitting from source to target is performed and both residual error (left) as well as calculation time (right) are measured. The illustration on the left shows that the accuracy decreases for larger amounts of missing data while convergence for more than 50% missing points is not always reached. On the right, it can be seen that the calculation time for identical set cardinality is small and larger if the set amount differs. Decreasing set size in the target then again reduces the computation time.

Besides looking into ways to deal with larger point clouds, we consecutively also add some thoughts on procedures that help in practice and offer reliable extraction of the marker models. Moreover, we consider possible changes and show limitations of the proposed tracking system.

7.5. Improvements and Limitations

In this section, we want to discuss two things. Firstly, we focus on the actual point sets for the fitting process, and determine a way to **train a point cloud model** of a marked object. And secondly, we develop ideas for the application of Algorithm 7.3 in case of **real-time problems with large point clouds**.

We assume that a camera system as described in section 7.1 offers a video stream of a scene with a marked object. As mentioned in the beginning of section 7.3.2, we have in principle two different ways of treating the matching problem. Either we match the point sets from two consecutive frame pairs or we match a virtual model cloud onto the current pair.
Due to error propagation and the often required need for an absolute pose, we investigate the latter. Also if we want to formulate a robust algorithm in the presence of occlusion, noise, and other unpredictable image corruptors, we would decrease its accuracy if we take two possibly incorrect, or incomplete images. Thus, we fit a fixed point cloud that represents the known object onto our current measurements. Let us therefore think about a way to retrieve the model cloud.

7.5.1. Model Cloud

If all markers of a target are clearly visible from one viewing position, we take a set of training pictures for a **static training** in the following way.

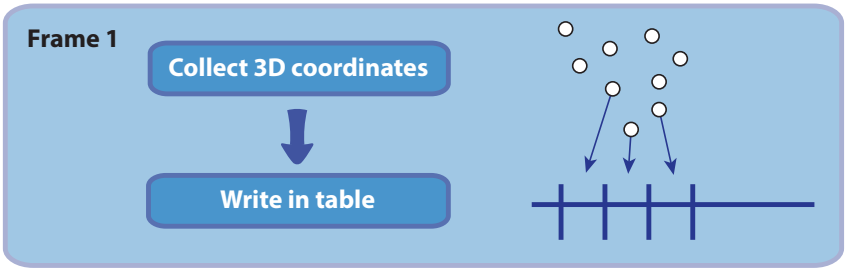


Fig. 7.20. **Collection of coordinates from first frame.** All measured 3D coordinates (right) are collected in one column of a table each.

At first, we move the calibrated camera system to an angle where the markers are in sight for both cameras. After that, we start with the image acquisition and gather the 3D point coordinates for the markers of the first considered frame pair with Algorithm 6.3. The coordinates of every single point are then written in a different column within the first row of a table as shown in Fig. 7.20. Afterwards, we take the second frame into account and obtain the points within the scene (cf. Fig. 7.21). Due to measurement noise, these points are usually not exactly at the same location as in the frame before. Thus, for every coordinate extracted from frame 2, we look for the column whose centroid has the smallest distance to this point. If it is below a certain threshold which represents the noise toleration of the process, we accept the point as the one we have seen already and append the point coordinates to this column.[25] If, however, the distance is above the threshold, we add the coordinates to a new column. This can be the case, if a point has not been recognized within the previous frame or if we retrieve a wrongly detected point.
We repeat this procedure for every frame pair of our training set.

After this, we have a table with potentially different sized columns. In order to remove incorrect point measurements and errors, we only accept columns with entries of at least half the size of the number of our training frames. This means they must appear at least in half of the acquired images. For all of the remaining columns we then exclude outliers and calculate the centroid for the rest of the points as illustrated in Fig. 7.22. These centroids are taken as the model point cloud for a static training case. Even though more advanced outlier and noise statistics can be investigated to clean the measurements, this simple procedure has shown to work well in practice.

In some scenarios, the markers of an object cannot be seen from a single viewpoint or model training needs to be performed in a dynamic environment. This can be the case, if self-occlusion appears, the motion of the object is somehow restricted or the scan is performed during object motion. It is still possible to calculate a model cloud or extend an existing one with **dynamic training**.

[25] The noise toleration depends on the resolution of the images, the marker size, the distance from the object to the camera system, and other setup parameters. For our test with an object distance to the cameras of approximately 50 cm and a marker size of 5 mm in diameter, we choose this to be 0.1 mm.

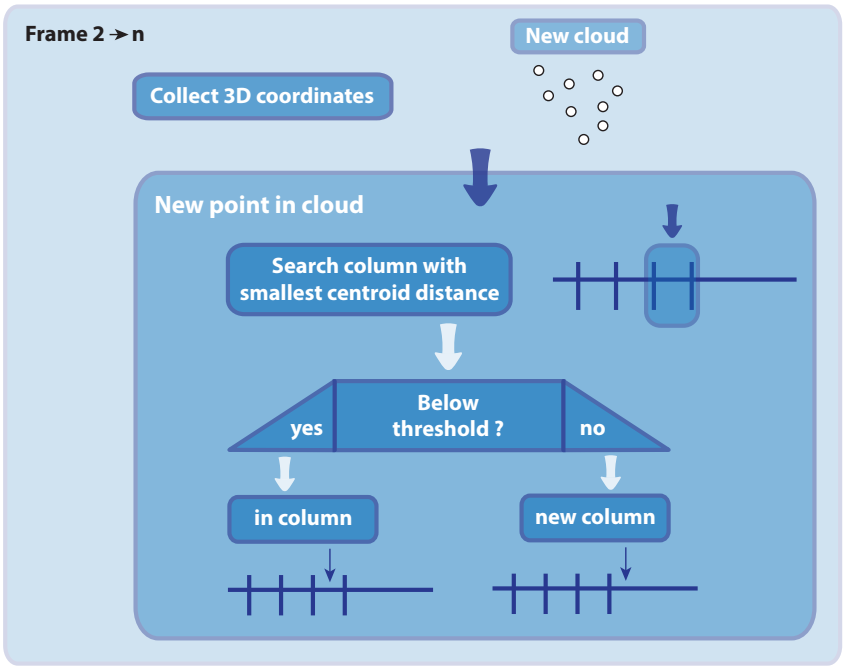


Fig. 7.21. **Clustering of points in model table for scene without movement.** The newly triangulated point cloud (top) is processed point by point. We first scan the existing table for the closest column and add a new row if it is close enough. Otherwise, the candidate is appended to a new column.

We can adapt the procedure described beforehand and fill the table in a different way. What we do is to keep a dynamically changing current model with all the information we gathered so far. In the first frame, for instance, this is just the position of all measured points. If we now acquire a new frame pair with its point cloud, we use the fitting Algorithm 7.3 to fit the current model points onto the new measurements. The underlying idea is that whenever there are some markers visible on the object and the frame rate is high enough, there is no detection of a completely new set of points with a new frame since real motion is continuous. Thus, we always see a subset of already known points and can perform a fitting.

With the matching comes a correspondence matrix $\mathbf{M}$. We use this correspondence to decide whether a point is already known or new. In the first case, we utilize the inverse pose $\mathbf{P}^{-1}$ to get the point in the same coordinate frame as the model cloud and append it to the corresponding column. In the second case, we transform the newly detected point with $\mathbf{P}^{-1}$ and add it to a

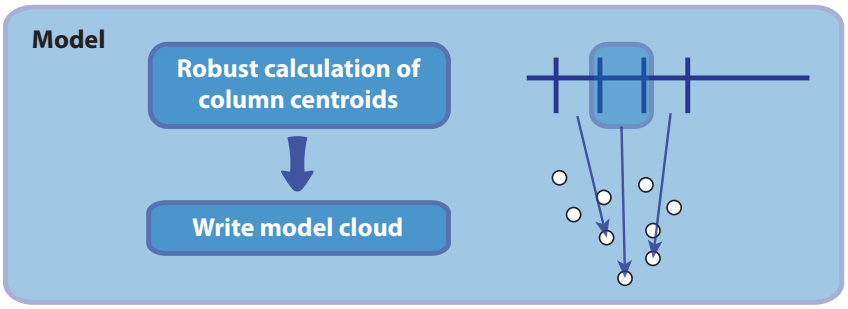


Fig. 7.22. **Calculation of model from table.** If the model table is built (top right), the source model is set up by calculating the centroids from the individual columns after outlier removal.

new column of the table. The whole process is shown in Fig. 7.23. The model extraction from the table content thereafter remains the same as before where the observation thresholds may be adjusted.
The computational complexity of this approach is higher as several fittings are necessary. We can also collect the point coordinates in each frame in an online mode and perform the robust model cloud calculation offline.

Even though the proposed algorithm can handle real-time tracking tasks, it has limitations. We now focus on the natural restrictions of the process and describe ways to overcome them.

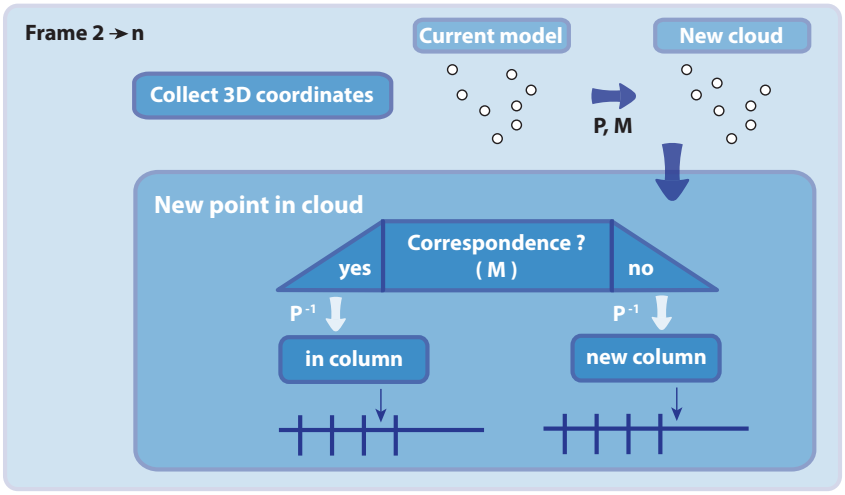


Fig. 7.23. **Clustering of points in model table for scene with movement.** In a dynamic environment, a pose **P** and a correspondence matrix **M** between the current model (top left) and the observation (top right) exist. These are calculated with the fitting algorithm and used to analyse (lower box) whether a point within the current observation has a partner in the model cloud. Depending on this, the point is transformed into the model reference frame by $\mathbf{P}^{-1}$ and appended to the existing observations or added as a new model candidate.

7.5.2. Handling Large Clouds

The immanent limit of fitting Algorithm 7.3 for real-time applications is given by the size of the point sets. In section 7.4.2 we have seen that real-time processing with a cloud size of considerably more than 25 points is critical for a standard hardware setup. Notwithstanding, exactly this can be essential for certain applications in interactive environments. A high marker density on an object is mostly preferable since it also increases the robustness of the fitting process against occlusion as Fig. 7.24 shows. For the first setup, the occluding rectangle impedes a pose detection whereas the second denser setup offers enough visible points even with the occluded part.

In this section we finish our algorithmic modeling and describe a way to **fit large point clouds** onto each other in real time. For this to be feasible, we focus on subsets of the model cloud and use **relevant subsets** within the actual measurement for a fit. Since we know that the point cloud itself is static, the pose of a subset with a reasonably large cardinality represents

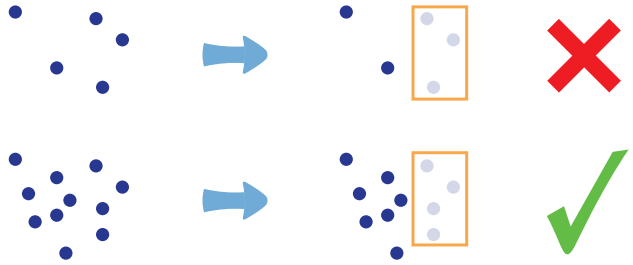


Fig. 7.24. **Marker density and robustness against occlusion.** The upper (small) point setup prevents a calculation of the pose in presence of the occluding box while the lower (larger) point setup provides enough measurements to perform a fit even under occlusion.

the pose of the entire cloud and can provide redundancy if multiple sets are used.
Let us explain how we adroitly choose the points for the two relevant subsets.

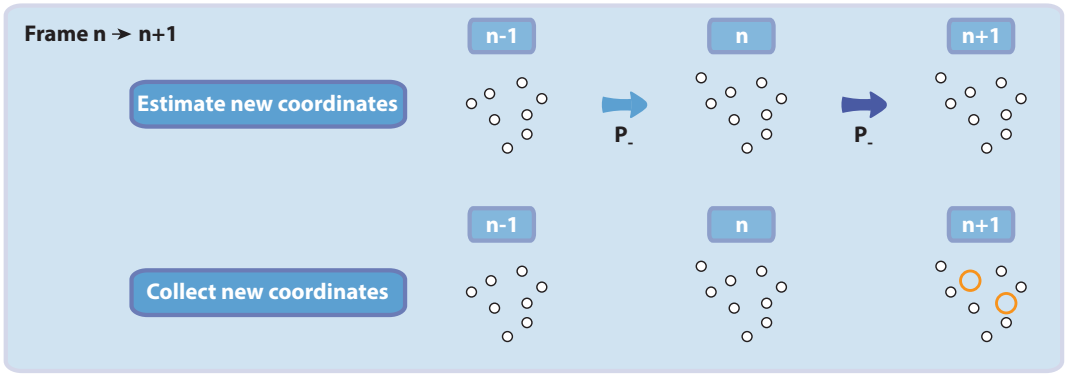


Fig. 7.25. **Difference between estimated point set and measurement for new frame.** The pose $\mathbf{P}_{-}$ that fits the points from frame $n-1$ to n can be used to estimate the location of the points in frame $n+1$ (upper row). A comparison of the estimates with the points in frame $n+1$ reveals missing points (lower row). The set of matching partners reduces.

For our further thoughts, we follow the illustration in Fig. 7.25.
Let us assume we have already fitted frame pair $n-1$ to n and look for a fit of frame pair n to $n+1$. Let the relative pose from $n-1$ to n be given by $\mathbf{P}_{-}$.[26] A motion in space is continuous. With an adequately high frame rate, the transformation of the points from frame n by $\mathbf{P}_{-}$ is therefore a good estimate for the position of the points in the next frame $n+1$. As a further step, we get the point coordinates of frame pair $n+1$ with the stereo correspondence Algorithm 6.3. Comparing their neighbourhoods with our estimates can now give either a match or a missing point. The idea is to find a subset within the newly acquired point cloud that can be used for a fit to a special subset of the model cloud. The representations $\mathbf{S}_{-}$ and $\mathbf{S}$ of such subsets within the measurements for both, frame $n-1$ and frame n are given in Fig. 7.26. It can be seen that the same subset is not a good choice for the next frame in this example since it contains one point less in the current cloud.
The size of these subsets represents the quality of the algorithm and is directly correlated to its runtime. We look for a way to create a new subset $\mathbf{S}_{+}$ of the same size as $\mathbf{S}$.

[26] In the beginning of our tracking procedure we initiate this with an identity rotation matrix $\mathbf{R} = \mathbf{I}$ and a zero translation vector $\mathbf{t} = \mathbf{0}$.

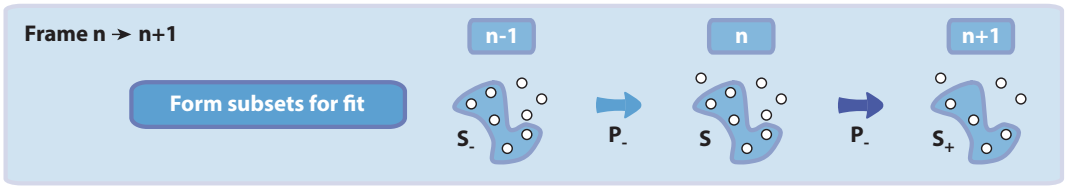


Fig. 7.26. **Representation of subsets in frame $n-1$, n and subset guess for $n+1$.** The pose estimation between frame $n-1$ and n gives two sets of corresponding points $\mathbf{S}_-$ and $\mathbf{S}$ (highlighted in the middle). The set $\mathbf{S}$ can be used to estimate a set $\mathbf{S}_+$ of points in frame $n+1$ (right). The cardinality of the sets may not be the same as shown here.

Which points are potential candidates? As a start, we look in the neighbourhood of the missing points arising from set $\mathbf{S}$ and pose $\mathbf{P}_-$. We include all points of frame $n+1$ that are in close vicinity to these points. If it holds that $|\mathbf{S}_+| = |\mathbf{S}|$, we are done. Due to errors and occlusion it is possible that not every point of $\mathbf{S}$ has a partner in frame $n+1$. One such example is visualized in Fig. 7.27. We then adapt the set $\mathbf{S}$ by deleting the points without partners and adding new ones. If the transformation of an added point is in the neighbourhood of some point within the current measurement, we add this one to $\mathbf{S}_+$. The process continues as long as we find potential partners in frame $n+1$ to the newly added points of $\mathbf{S}$. If this always happens, we end up with two equally sized sets ready to use for a fit.

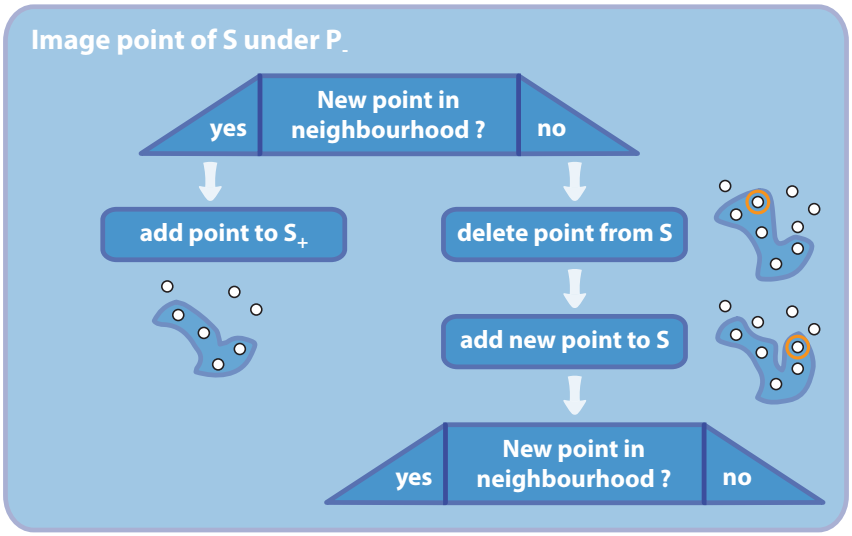


Fig. 7.27. **Creation of subsets $\mathbf{S}$ and $\mathbf{S}_+$.** The estimated points from set $\mathbf{S}$ under the transformation $\mathbf{P}_-$ are investigated iteratively to create the a new set $\mathbf{S}_+$. If they have a partner in their neighbourhood (left), this point is added to $\mathbf{S}_+$. If this is not the case (right), the point is deleted from $\mathbf{S}$ and a new point is added for which the process repeats.

If, however, it appears that a newly added point from $\mathbf{S}$ does not have a partner in the image pair $n+1$, we delete it once again from the set and add a new point which has not been considered so far. This procedure is illustrated in Fig. 7.28 and recurs until its estimated neighbourhood matches some current coordinates of frame $n+1$. These coordinates are added to the set $\mathbf{S}_+$. The algorithm finally terminates when both sets $\mathbf{S}$ and $\mathbf{S}_+$ are of equal cardinality or no triangulated points are left to be considered.

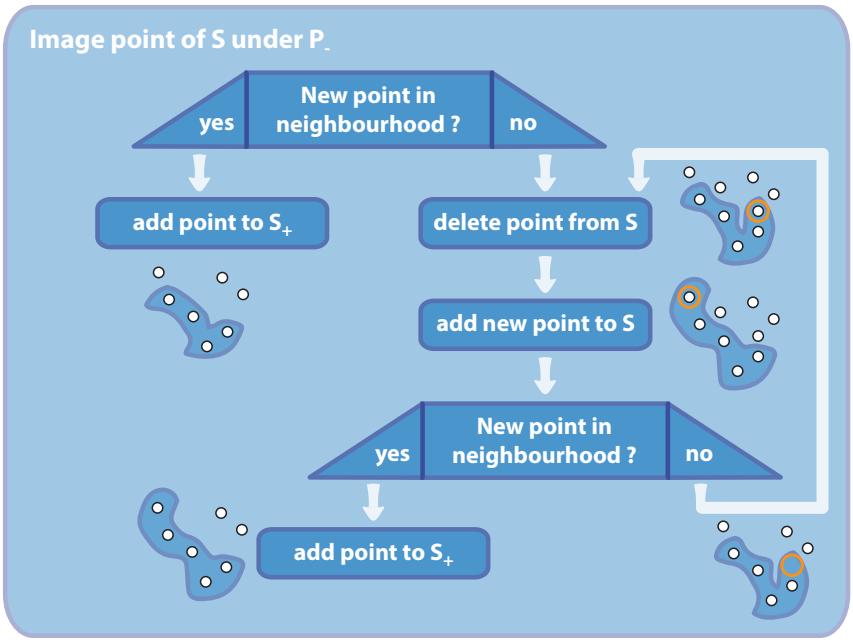


Fig. 7.28. Modification of subsets S and $\mathbf{S}_+$. The process from Fig. 7.27 repeats (right) for all the points in set **S** until a set $\mathbf{S}_+$ of equal cardinality is reached (lower left) or all points are processed.

We note that the actual fitting is still done from a model cloud to the actual frame and not relative from one frame pair to the next. The relative pose $\mathbf{P}_-$ is solely given by the two poses $\mathbf{P}_{n-1}$ and $\mathbf{P}_n$ that fit the model (i.e. a subset of it) to the points of frame $n-1$ and n respectively. The model cloud itself is not shown in the illustrations, but we see the representation of a part of it and we only marked the subsets of them within the single frames for visual simplicity. **Multiple** evenly distributed dynamic model **subsets add redundancy and robustness** to the overall process in practice and pose averaging can be used to integrate the individual pose estimates to a final result.[27] While Algorithm 7.3 is thereby used with a relaxed error tolerance to robustly find a solution, a fast consecutive processing with ICP (cf. section 7.3.2) iteratively refines the pose. In total this procedure makes the fitting approach applicable to large clouds by using smaller ones.

Now that we are able to robustly estimate the pose of an object, we investigate how to communicate this pose to other machines and algorithms.

7.6. Communication Interface

The optical tracker as presented in sections 7.1 is set up on an independent machine that connects to the tracking cameras and other machines via Ethernet. In order to **control the tracker** and **transmit the pose** of the OTS, we implement an interface that communicates **via TCP/IP over Ethernet**. Over the interface, the OTS machine can receive commands from the client and transmit measurement results.

[27] Cf. Hartley, Aftab, and Trumpf [164].

7.6.1. OpenIGTLink

For both control and pose data, the **OpenIGTLink** protocol as proposed by Tokuda et al. [413] is used in Version 2. OpenIGTLink is a multi-platform network communication interface used for image-guided and robot-assisted medical interventions. The OTS machine runs an OpenIGTLink server and every connected machine implements an OpenIGTLink client.

7.6.1.1. Message Header

The standard message header contains information about the transmitted data and specifies its format. Following the standard protocol, the header is set up as shown in Fig. 7.29 where the header fields follow Table 7.2.

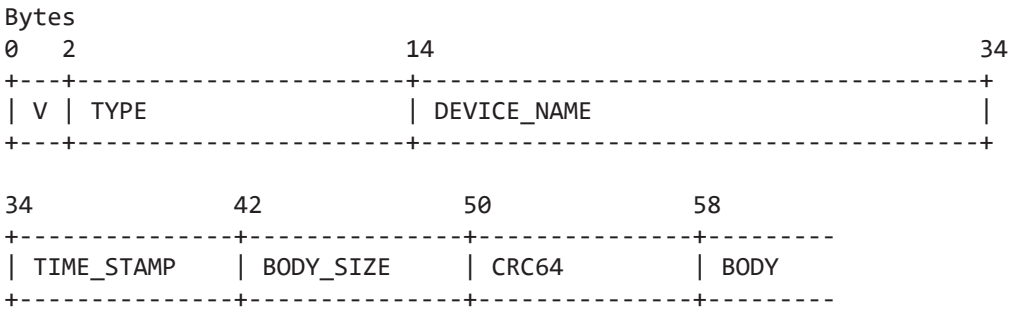


Fig. 7.29. **OpenIGTLink message header structure.** Every message sent follows the depicted format where the first 58 bytes are used as a header to specify the message BODY.

Data	Type	Description
V	Unsigned short (16bit)	Version number (2)
TYPE	char[12]	Type name of data
DEVICE_NAME	char[12]	Unique object name (ObjXY)
TIME_STAMP	64-bit unsigned int	Timestamp of OTS system
BODY_SIZE	64-bit unsigned int	Size of message body in bytes
CRC	64-bit unsigned int	64 bit CRC for body data

Tab. 7.2. **OpenIGTLink message header.** The different parts of the message header contain meta information about the message.

7.6.1.2. Message Body

To communicate with the OTS two types of messages are used: the **TRANSFORM message** sends the upper 3 rows of a homogeneous linear transformation in 4×4 matrix form and specifies the current measurements for the objects (ObjXY) that are tracked while the **STRING message** is used to control the OTS. Their message bodies are shown in Tables 7.3 and 7.4.

Data	Type	Description
R11	32-bit float	Element (1,1) in 4-by-4 linear transformation matrix
R21	32-bit float	Element (2,1) in 4-by-4 linear transformation matrix
R31	32-bit float	Element (3,1) in 4-by-4 linear transformation matrix
R12	32-bit float	Element (1,2) in 4-by-4 linear transformation matrix
R22	32-bit float	Element (2,2) in 4-by-4 linear transformation matrix
R32	32-bit float	Element (3,2) in 4-by-4 linear transformation matrix
R13	32-bit float	Element (1,3) in 4-by-4 linear transformation matrix
R23	32-bit float	Element (2,3) in 4-by-4 linear transformation matrix
R33	32-bit float	Element (3,3) in 4-by-4 linear transformation matrix
TX	32-bit float	Element (1,4) in 4-by-4 linear transformation matrix
TY	32-bit float	Element (2,4) in 4-by-4 linear transformation matrix
TZ	32-bit float	Element (3,4) in 4-by-4 linear transformation matrix

Tab. 7.3. **OpenIGTLink TRANSFORM message body.** The message contains the first 3 rows of a 4×4 homogeneous linear transform. The translation units are in millimeters.

Data	Type	Description
ENCODING	uint16	Character encoding type as MIBenum
LENGTH	uint16	Length of string (bytes)
STRING	uint8[LENGTH]	Byte array of the string

Tab. 7.4. **OpenIGTLink STRING message body.** Encoding and length are defined before the string content of the message.

7.6.2. OTS Control

A tracking of up to three objects at the same time is set as standard and the DEVICE_NAME in the message header (cf. Table 7.2) encodes which object pose is sent. Unique object names are specified during training as “ObjXY”, where XY depicts a zero padded number between 00 and 20. While the information flow for TRANSFORM messages is from the OTS to the client, STRING messages are used bidirectionally to control the optical tracker and to receive confirmation as well as error messages. The standard set of messages is shown in Table 7.5. This allows the client to control the OTS and to train new objects specified with index “XY” $\in \{00, 01, \ldots, 20\}$. For instance, if “Obj07” should be retrained, the client would sent “TRA07” which sets the OTS in training mode. After successful training of this configuration, the OTS sends “TRF07” and tracking of the new object can be started with “STA07”.
While the successful transmission of all commands from the client is confirmed with “ACK”, errors such as unsuccessful training, hardware failures, problematic exterior light conditions etc. can be specified in particular error codes depending on the application.

Byte Array	Message Description
STAXY	Start tracking of object XY
STOXY	Stop tracking of object XY
TRAXY	Start training of object XY
TRFXY	Training finished for object XY
STOAL	Stop tracking of all objects
RST	Reset optical tracking system
ACK	Acknowledge message to confirm command
ERRXY	Error code with number XY

Tab. 7.5. **String messages for bidirectional communication between client and OTS.** The client can send various command to control the OTS while the OTS confirms the controls and can send error codes.

7.6.2.1. Start & Stop of Tracking

The OTS is set to track up to three objects at the same time. If the client sends a tracking request (“STAXY”), the index XY of the previously trained object is specified. The server immediately starts tracking and sends an acknowledge message (“ACK”). As long as the object pose is determined, transform messages are sent. If the object is out of sight, no transformation message is sent. The process can either be stopped for specific objects (“STOXY”) or for all objects at once (“STOAL”). The system can also perform a soft reset with the command “RST” which resets the cameras and the image processing pipeline. The command will be acknowledged, too.

7.6.2.2. Model Training

A new model can be taught by setting the OTS in training mode for index XY ("TRAXY"). The server acknowledges the command ("ACK") and starts the training. During training, the model can be slowly moved within the working volume of the system (cf. section 7.5) to improve the model accuracy. A successful training of model XY is confirmed by the server with ("TRFXY"). The maximum training time is set to 30 sec and a training error can be specified as an error code XY ("ERRXY") that is send back by the OTS. In case that a model is already saved under a given index, the previous model is overridden.

Now that we can communicate poses to other systems, we want to investigate methods of interaction and collaboration. For this we take a look at co-calibration.

7.7. Tool Calibration

Tracking and communicating object poses over the network allows to calibrate multiple sensors or cameras with respect to manipulators, tools and other sensing systems. In order to enable such applications, we first detail the **calibration of** a marked **stylus** and consecutively discuss **robot-camera calibration** procedures.

7.7.1. Pivot Calibration

In medical as well as industrial setups, the use of a tracked stylus can be beneficial to point at some target to save its coordinates or to determine a fixed trajectory over time. Some medical tools such as drillers naturally come with a pointy end whose location is essential in surgery. Instead of tracking the tool tip, an OTS recognizes a marker setup somewhere rigidly attached on the object. In order to estimate the tip location relative to the tool coordinate system and thus relative to the marker system, a **pivot** or hot-spot **calibration** can be performed where the tip is pointed at some fixed location as shown in Fig. 7.30 while the tool is rotated around this pivot point.

During the process, the poses $\mathbf{R}_i, \mathbf{t}_i$ of the tool coordinates with $i \in \{1, \ldots, n\}$ can be recorded with an optical tracking system. Any point $\mathbf{p}$ from the tool coordinate system TCS can be calculated with respect to the world coordinate system WCS by

$$\mathbf{p}_{\text{WCS}} = \mathbf{R}\mathbf{p}_{\text{TCS}} + \mathbf{t} \tag{7.60}$$

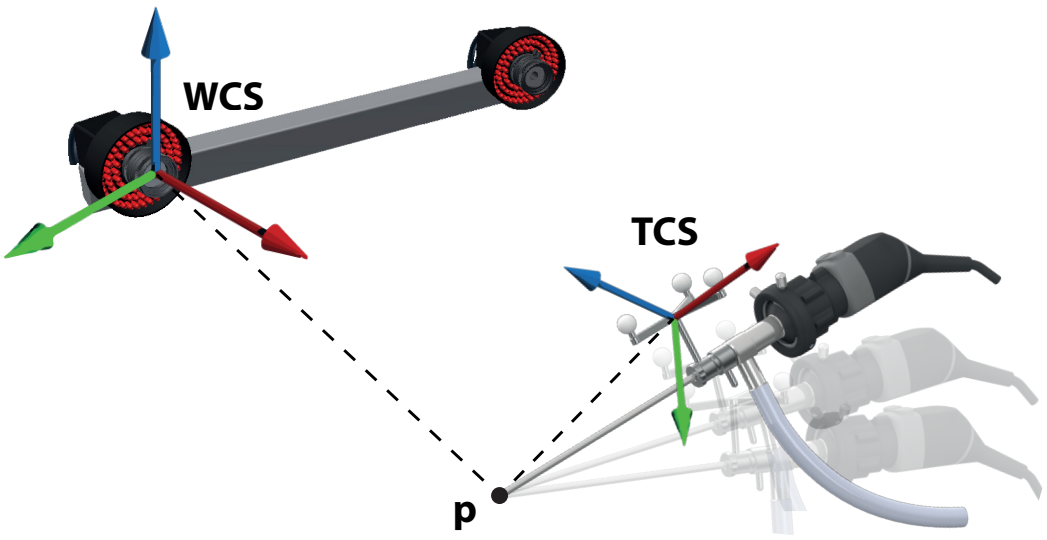


Fig. 7.30. **Pivot calibration setup.** A tool (right) is tracked from an optical tracking system (top left). The pose of the tool is given by the relative pose between the world coordinate system (WCS) and the tool coordinates (TCS). The tip coordinates are stable during a pivot movement where the tool tip $\mathbf{p}$ is fixed while the tool pivots around the point (semi-transparent illustration) with changing relative poses between the reference systems.

where the pose of the tool is given by $\mathbf{R}$ and $\mathbf{t}$.
Since the pivot point remains the same both in tool as well as world coordinates, we can write

$$\begin{pmatrix} \mathbf{R}_1 & -\mathbf{I} \\ \vdots & \vdots \\ \mathbf{R}_n & -\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{p}_{\text{TCS}} \\ \mathbf{p}_{\text{WCS}} \end{pmatrix} = \begin{pmatrix} -\mathbf{t}_1 \\ \vdots \\ -\mathbf{t}_n \end{pmatrix} \tag{7.61}$$

which gives an overdetermined linear system that can be solved as a least squares problem for $\mathbf{p}_{\text{TCS}}$ ($\mathbf{p}_{\text{WCS}}$). A solution can be found for instance with a singular value decomposition or some robust optimization method.
This allows us to use hand-held stylus-like tools with precise knowledge of their tips. We make practical use of pivot calibration in the collaborative medical application described in chapter 10.3.

However, for various reasons such as precision, accuracy and repeatability as well as for collaborative support, sensors or tools may also be held by a robotic manipulator for which further calibration methods are needed to determine the position and orientation of all involved components and devices in a common coordinate frame.

7.7.2. Hand-Eye Calibration

To **calibrate a camera and a robotic arm** such that their location is known in a common coordinate reference, a hand-eye calibration is performed. We utilize the work of Tsai and

Lenz [421] with its implementation in the Visual Servoing Platform (ViSP library)[28] and calibrate depending on the location of the vision sensor as shown in Fig. 7.31.
In the **eye-in-hand** case, the camera is moving with the end effector of the robotic manipulator. We calibrate the static transformation between both coordinate frames by placing a marker pattern in the field of view and performing a series of movements with the robot to determine the pose.
Our **eye-on-base** routine provides the information for the relative pose between robot base and an external optical tracking system. For this process we attach a marker on the robot end effector and run through a series of different positions to calculate the pose.

In chapter 10.3 we make use of these calibration techniques to calibrate the different components of a collaborative robotic arm to support a medical procedure and enable multi-modal sensor fusion. In chapter 8.1, we leverage the hand-eye calibration routines to produce robotic ground truth for volumetric 3D ultrasound scans. Furthermore, in chapter 7.9, we use a calibrated robotic arm to setup a prototype for cooperative rehabilitation therapy.

While additional sensors and co-calibrated tools are beneficial in many applications, an individually placed static tracking system to observe all involved objects and tools is often problematic in practice. The reason for this lies in the intrinsic requirement of the **outside-in tracking system** to have a clear line of sight to observe all involved instruments and a considerable amount of markers for precise tracking from the camera viewpoint. We discuss possible relaxations for this requirement hereafter and explain an example application where more flexibility is essential.

7.8. Line of Sight

In practical applications with various object for tracking, a single fixed optical tracking system that observes the scene from the outside is not always sufficient to see enough details such that a tracking of every involved component can be realized at all times. Oftentimes a stereo tracker is put on a flexible tripod or mounted at the ceiling. Especially when humans interact in the scene the **line of sight** of a single or both cameras of the stereo system **can be obstructed** and tracking is lost. One solution for this problem is to co-calibrate **multiple trackers** that can be placed at different locations. However, the problem remains if occluders occur close to the object of interest which is often the case if the tracked tools are objects for interaction.
An example for such a case can be an industrial factory where hand-held devices are used for manufacturing. And in a clinical environment, the surgery room is often populated with multiple people that interact close to the patient while **outside-in** trackers are further away. In order to ensure minimal distraction and to support with pose information in these cases, we consider solutions to the line-of-sight issue.

Extending our thoughts from the previous chapters, we hereafter construct a **dynamic camera-in-hand** marker-based stereo tracking system that is placed within the scene and observes the objects as shown in Fig. 7.32 while being able to move. We then test this system in a medical

[28] Cf. Marchand, Spindler, and Chaumette [275].

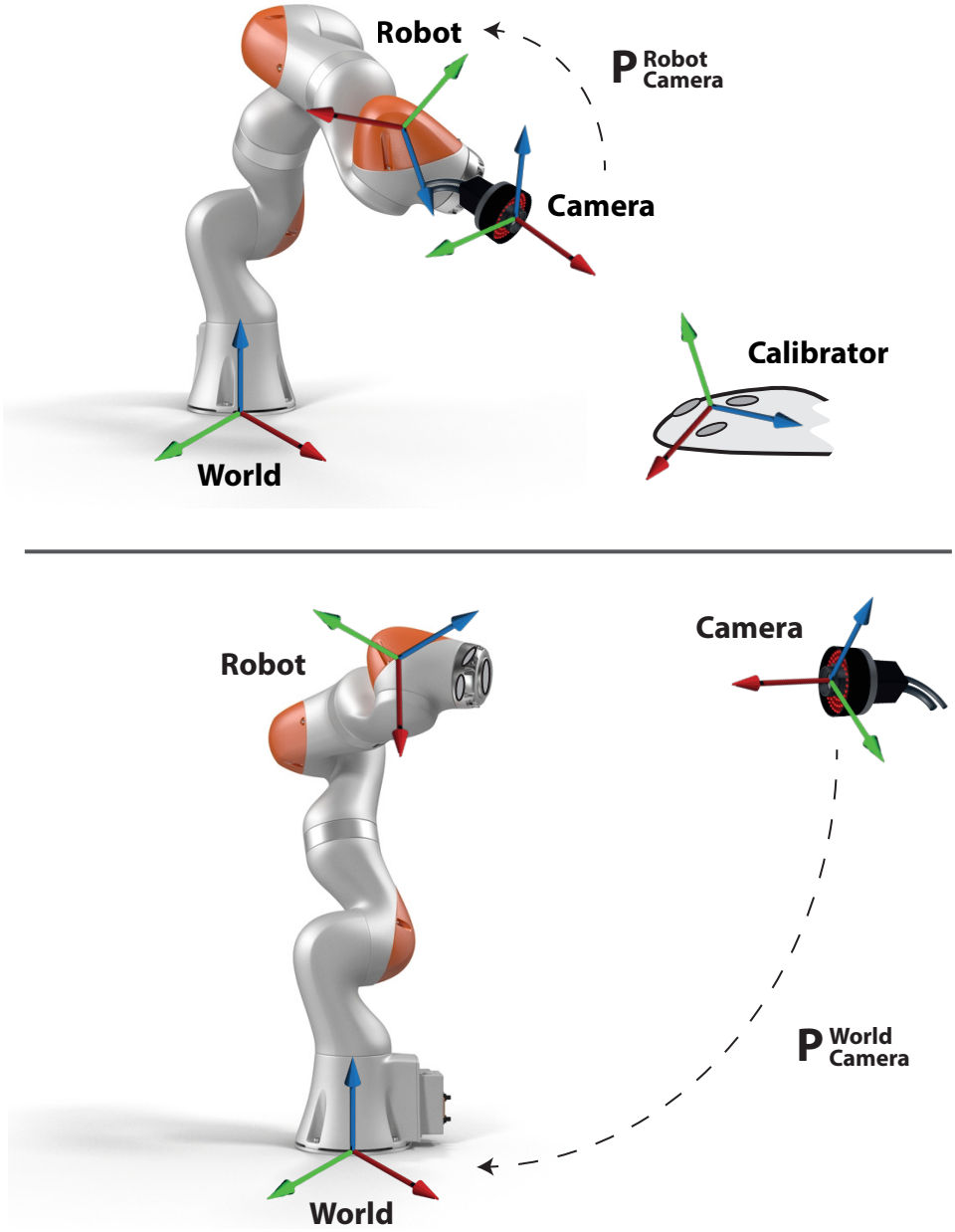


Fig. 7.31. **Hand-eye calibration setups.** The top part illustrates the *eye-in-hand* case where a camera system (camera) is rigidly attached to the robot. The camera system observes a static calibration target marked with retro-reflective circular markers and the calibration routine determines the poses between camera and robot end effector ($\mathbf{P}^{\text{Robot}}_{\text{Camera}}$) while the pose between the end effector and the world reference frame is provided through forward kinematics. The lower part depicts the *eye-on-base* case where a static optical system (camera) observes the marked end effector of the robot and the pose of the system relative to a world coordinate frame ($\mathbf{P}^{\text{World}}_{\text{Camera}}$) is determined while the robot pose is given. The robot frame and the rigidly attached object reference frame do not necessarily coincide.

setup.
In chapter 8 we consecutively extend the ideas and describe a markerless approach for **inside-out** stereo tracking before we focus on the monocular case.

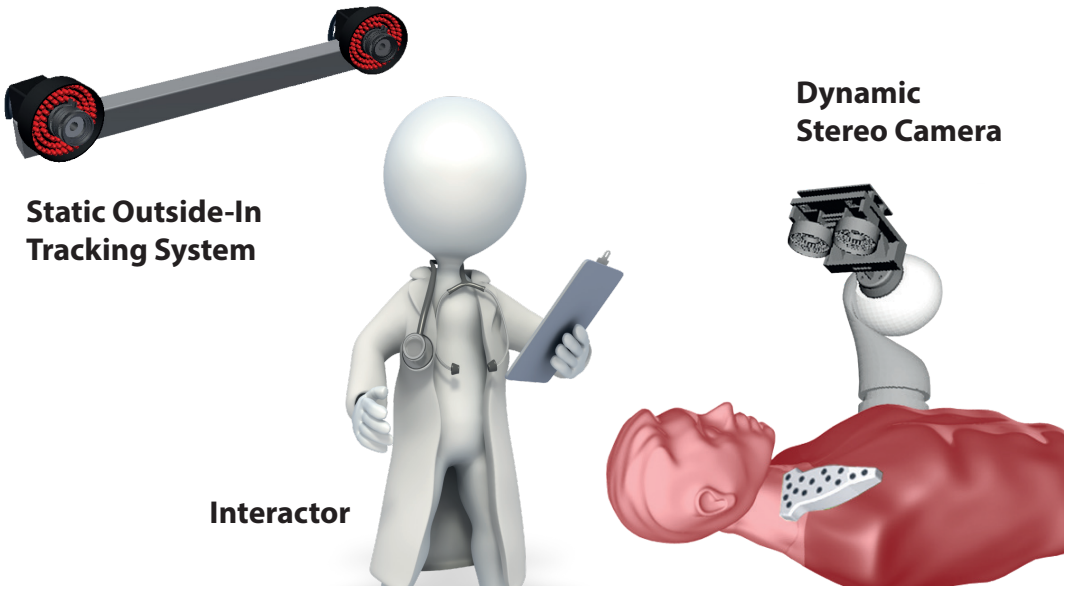


Fig. 7.32. **Static outside-in and dynamic camera-in-hand tracking system.** A typical outside-in stereo tracking system (top left) is statically mounted on a tripod or at the ceiling. In dynamic environments, the line of sight can be obstructed in various ways. One common factor are people interacting in the scene (middle). A dynamic camera system illustrated on a robotic arm (top right) allows to observe poses from within the scene and can change its viewing angle to continuously observe the marked object depicted as an ultrasound transducer (lower right).

7.8.1. Virtual & Dynamic Cameras

Aside of the physical extension of the tracking setup by using multiple optical systems, an indirect extension for outside-in trackers with **mirror tracking** is debatable. In this case, one or multiple mirrors that are also tracked, virtually extend the amount of optical viewing angles as shown in Fig. 7.33. However, due to the error propagation through the pose estimation of the planar mirror itself, the tracking quality decreases. Moreover, the physical mirror is also not a perfect plane and manufacturing differences can lead to considerable pose quality differences[29] or require a calibration of the mirror plane.

In the following, we therefore consider another solution and turn the problem at its head by moving the camera into the scene. The prototype evolution is shown in Fig. 7.34. A direct translation from the marker-based outside-in concept constitutes a **moveable camera-in-hand OTS** stereo variant with active ring-lights. We apply the system in practice in chapter 7.9 and in chapter 10.3.
This first stereo setup uses two board-level GC1291M-BL cameras (SMARTEK Vision, Croatia) equipped with miniature DSL315B-NIR fisheye lenses (Sunex, USA). The individual monocular field of view with 135° allows for a wide sight coverage and two IF 093 NIR band-pass

[29] Cf. Liu, Wu, and Wu [259].

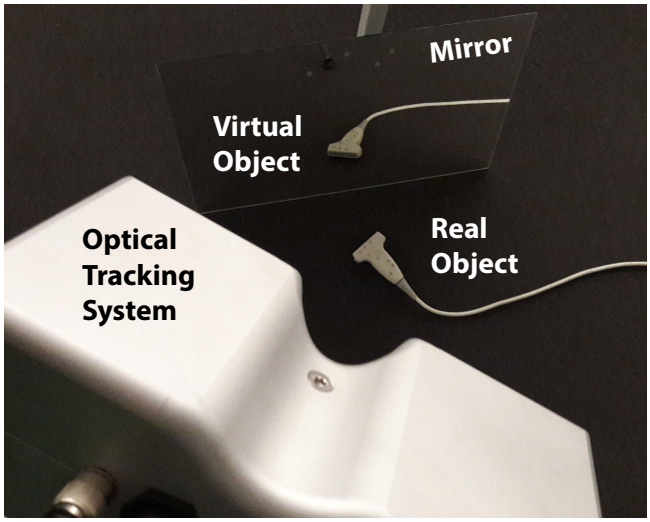


Fig. 7.33. **Mirror tracking and virtual viewpoint.** The amount of viewpoints for an optical tracking system (lower part) can be virtually extended by placing a marked planar mirror (top) in the scene. Besides the direct line of sight for a real object observation, an indirect view through the mirror is possible. Knowing the pose of the mirror, it is possible to track the objects through the mirror while both the object and the mirror can change their position and orientation.

filters (Schneider-Kreuznach, Germany) are inserted into each optical path. The direct illumination is done with two FLDR-i70A LED rings (FALCON Illumination, Malaysia) that flash at a 875 nm wavelength. The illuminators are triggered during camera exposure by an IPSC2 strobe controller (SMARTEK Vision, Croatia) with 750 mA at 24 VDC. Both cameras acquire synchronized images due to a hardware trigger at a frame rate of 24 Hz with an exposure time of 1.5 ms.
To improve the energy efficiency, the hardware size and the need for strobe controlling to steer the active illumination, we further develop a passive RGB stereo-system which can be equally mounted on a robotic arm. To enable also flexible hand-held use, the system is then miniaturized with a mountable and boxed RGBD sensor for **inside-out tracking** and further improved in size with a board level sensor and a custom mount.

Miniature inside-out or camera-in-hand vision system such as the ones presented here can improve the visibility restrictions often caused by line of sight loss. We provide an example for robot supported movement therapy in chapter 7.9 and use similar hardware for robust sensor fusion in part 10.3. However, the constant requirement for a free line of sight remains an inherent part of any vision system and at the moment in which the vision system looses sight, the source for the tracking information is lost. Various scholars therefore propose hybrid modality solutions leveraging inertial measurement units (IMUs) which we briefly mention here.

7.8.2. Visual-inertial Tracking

In case of occlusion, a vision system is unable to reliably track the object of interest. While it is possible to interpolate and extrapolate poses (cf. section 9.1), measurements without

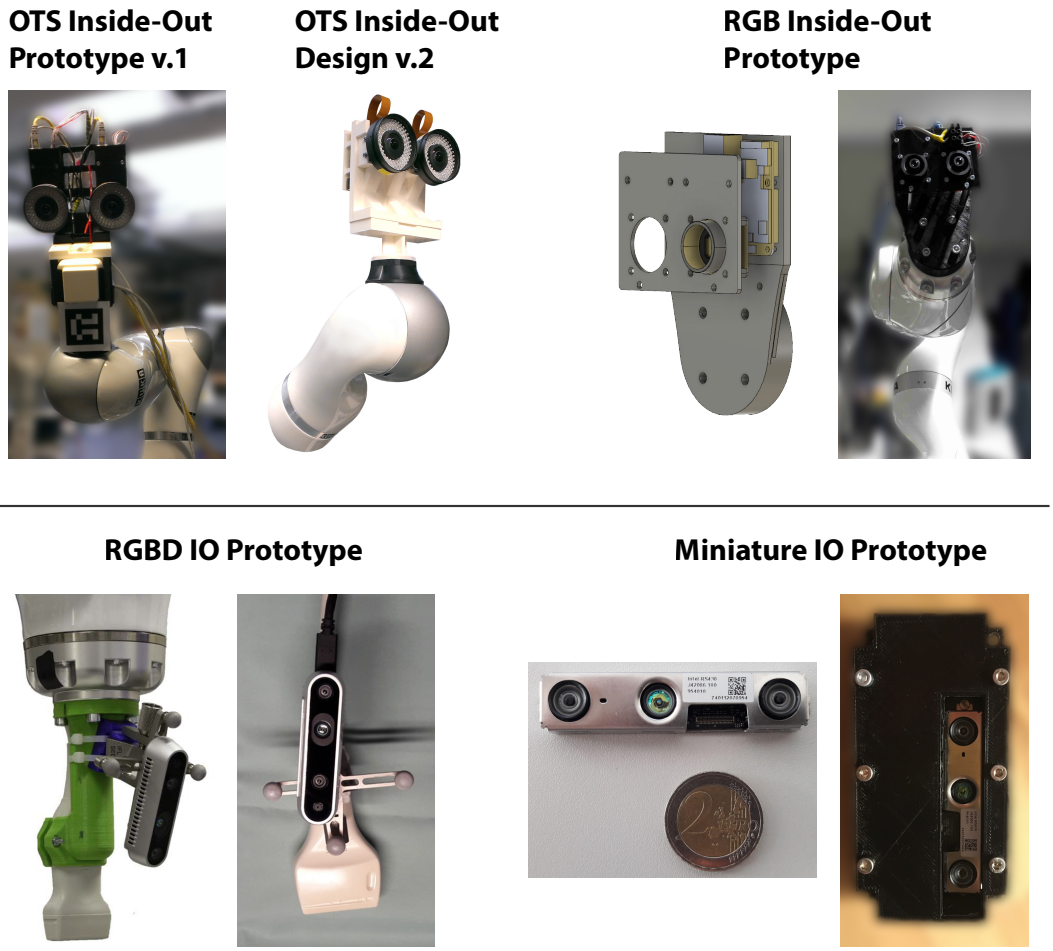


Fig. 7.34. **Dynamic OTS prototype evolution.** The first dynamic OTS systems (top left) are a direct integration of the marker-based outside-in concept with a significantly smaller size such that they can be mounted on a robotic arm. A second iteration (top right) leaves out the ring LED lights to enable a passive stereo system with similar geometry. In order to miniaturize and enable mobile hand-held applications for inside-out usage, an RGBD sensor is used (bottom). A boxed version based on an intel RealSense D435 camera with the dimension of 90 mm x 25 mm x 25 mm is used with a flexible mount (bottom left). It is further miniaturized with a board-level version of the intel RealSense D430 (bottom right).

observations are limited.
One way of dealing with this is the use of non-vision hardware. Aside of EM tracking devices as discussed in chapter 7.2.2, the hybrid use of **inertial measurement units** (**IMUs**) together with vision system can be mutually beneficial. An IMU measures acceleration (accelerometer) as well as angular velocity (gyroscope) which can be integrated to help the measurements of orientation and translation in the absence of visual data or enrich the input signal in a joint hybrid pose estimation framework.

The first visual-inertial fusion systems are used in inside-out scenarios where the requirement is **simultaneous localization and mapping** (**SLAM**) or **visual-inertial odometry** (**VIO**).
Early works are combined with keypoint extraction pipelines to minimize the data input. The scholars Mourikis et al. [291] propose an efficient EKF-based fusion algorithm to integrate IMU and vision data. And Jones et al. [198] focus on robustifying the fusion to maintain scalability. They describe experiments with correct trajectories in sequences with up to 30 km

trajectory length. Vanishing points from parallel lines in urban scenes are used as additional constraints for stabilization by Camposeco et al. [59]. And Leutenegger et al. [247] propose a probabilistic cost function to optimize the fusion in a keyframe-driven nonlinear optimization scheme. VINS-mono[30] combines monocular vision and IMU tightly in a single framework that is equipped with loop detection and pose graph optimization for long term stability while Mur-Artal et al. [299] establish a real-time VIO system with loop closure where the already established map can be reused.

Direct methods leverage photo-consistencies and perform visual-inertial odometry without keypoint and feature extraction. The work of Forster et al. [119] uses pixel-intensities leveraging the manifold structure of the rotational group while the full geometry is estimated with a stereo vision plus IMU setup by Usenko et al. [430] where the authors build a semi-dense map from the joint pixel-IMU measurements. The formulation is put into sensor-centric perspective by Bloesch et al. [32]. They optimize a photometric loss with an iterative extended Kalman filter (EKF) and Von Stumberg et al. [437] extend a vision-only SLAM system[31] to a hybrid fusion method and add the IMU into the coupled optimization. Previous information is integrated with a marginalization strategy.

The vast majority of SLAM and odometry approaches that help to estimate the camera pose with vision and IMU data do utilize the image as a static data source. This static assumption is true for global shutter sensors. However, from a cost perspective as well as to utilize existing hardware with pre-integrated IMUs such as mobile phones, it can be convenient to also use low-budget vision sensors. These are **rolling shutter cameras** in most cases. Interpolation techniques with specific models for rolling shutter sensors[32] can be used for appropriate VIO integration and continuous-time models. Splines[33], for instance, can help to fuse rolling shutter camera acquisitions with high-frequency accelerometer and gyroscope data. Efficient implementations such as the EKF-based fusion pipeline from Li et al. [248] enable runtime of these approaches on mobile devices, too.

The community for hybrid approaches grows significantly in the last years and newer **datasets** help to compare approaches objectively and quantify their performance on various target domains with real and synthetic data. To specifically tackle real noise and data corruption, the dataset of Chen et al. [65] can help to analyse robustness of VIO fusion approaches and the fully synthetic data from Cao et al. [61] aims to test above mentioned rolling shutter pipelines with tests on the Wuhan University Rolling Shutter Visual-Inertial (WHU-RSVI) data.
The synthetic indoor dataset from Kirsanov et al. [220] aims to provide insights into pipelines targeted at content-awareness by focusing around semantic and panoptic segmentation challenges in the presence of IMU and vision data while the NEAR dataset[34] is built to investigate the performance of VIO approaches in the context of indoor augmented (AR) and mixed reality (MR) applications.
Indoor and outdoor scenes are provided by the TUM VI benchmark of Schubert et al. [370]. The dataset contains visual and inertial data from wide-angle stereo and IMU of real scenes

[30] Cf. Qin, Li, and Shen [336].
[31] Cf. Engel, Koltun, and Cremers [97].
[32] Cf. Guo et al. [155].
[33] Cf. Lovegrove, Patron-Perez, and Sibley [263].
[34] Cf. Wang et al. [442].

in varying scenarios with highly accurate pose information from a motion capture system at the start and end of the scenes.

While this gives a brief overview over the direction in this field, we decide to focus on the vision-only aspects first and integrate other sensor modalities in chapter 10. We exemplify the advantage of line-of-sight aware approaches with a prototype of a cooperative robot that can be used in medical therapy.

7.9. Cooperative Robotic Movement Therapy

Movement therapy plays an integral part in stroke rehabilitation where repetitive motion with active patient participation is crucial for the positive outcome. Intense similar motion steered by the patient triggered the use of collaborative robots as therapy assistants. In this chapter, we discuss how to use **camera-in-hand tracking for training of upper limb movements in hemiparetic patients after stroke with the help of a light-weight robotic arm**. The robotic assistant equipped with the stereo tracker supports the movement of the deficient arm while the **patient steers the robot with natural movements of the healthy arm** which wears a sleeve with retro-reflective markers.
Online model training and adjustments help to ease the interaction while the optical tracker updates the pose of the arm within 9 ms and with a precision of 0.5 mm. A series of tests with healthy subjects show the applicability of the approach to accurately mirror the movements from the healthy to the potentially impaired arm.

7.9.1. Medical Background & Therapy Forms

In the year 2010, a total of 17 million people were affected by strokes globally while 33 million post stroke patients were alive.[35] Hemiparesis is affecting 80% of stroke patients.[36] They suffer from not being able to properly move one side of their upper or lower limbs. Therapy strategies and muscle rehabilitation thus have an immediate impact on the quality of their everyday life. There are various medical and therapeutic approaches towards limb recovery which have been summarized in the works of Pulman et al. [334] and Basteris et al. [15]. These include electrical stimulation, neurological therapy, mental practice and imagery, constraint-induced movement therapy as well as mirror therapy, and repetitive task practice. Given the movements and processes, the latter two fields can benefit from robotic support. In approaches that include supported motion, **patient participation** turns out to be **of significant importance both for the immediate as well as the longer term outcomes of stroke rehabilitation**, in particular for upper-limb exercises with intensive movement.[37] Current robot-mediated therapy forms are categorized depending on the type of human-robot interaction as defined by Basteris et al. [15]. We present a passive mirror-therapy system in which the patient controls a robotic arm with his unimpaired arm and the robot transfers the motion onto the affected arm.

[35] Cf. Feigin et al. [107].
[36] Cf. Pulman and Buckley [334].
[37] Cf. Blank et al. [31].

Previous systems such as the MIME, BFIAMT, and Bi-Manu-Track are controlled with a joystick while U-EX07 requires the use of an exoskeleton. To the best of our knowledge, the recent advent of flexible 3D computer vision systems has not been leveraged for robot-assisted movement therapy. We therefore present the first passive mirrored upper limb rehabilitation system that is able to transfer the movements from a guiding arm in real-time onto training moves for the impaired arm.

7.9.2. System and Setup

This application demonstrates a **first prototype** as a step towards **contact-free robot manipulation for upper limb rehabilitation**. The system is shown in Fig. 7.35 where the natural arm movement of the healthy side is observed by the stereo camera system mounted at the end effector of the light-weight robotic arm which observes the marked sleeve of the impaired side to mirror the movements and train the affected limb of the hemiparetic patient which is fixed to the robot. One difficulty lies in the robust tracking of the flexible healthy arm in real-time. The system is not only required to work accurately but also safely and robustly. Even if the marker is not fully in sight, the movement needs to be adequately reflected onto the impaired arm. To realize this, the system benefits from the developed algorithms and the marker-based tracking described before.

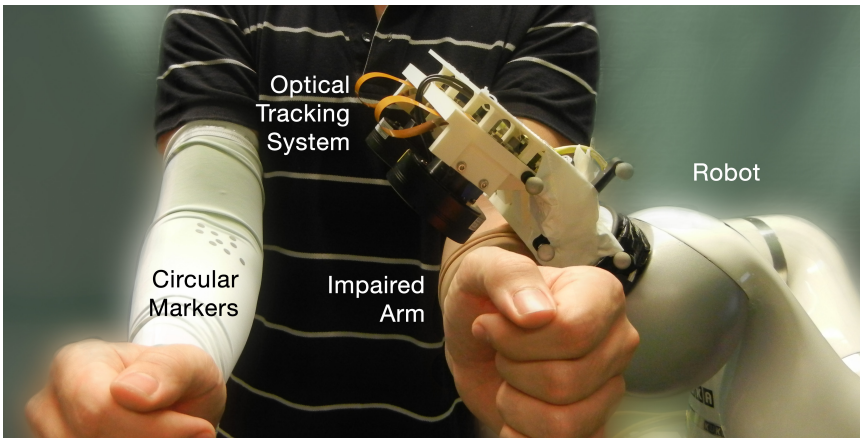


Fig. 7.35. **Cooperative robotic movement therapy system prototype.** An optical tracker (top) is attached to a robot (right). A hemiparetic patient is able to steer the motion of the impaired arm (right) attached to the robot end effector by movements of the healthy arm in a sleeve with retro-reflective markers (left). The observed motion of the healthy side is mirrored by the robot in real-time which moves the hemiparetic side.

7.9.2.1. Vision System and Robot Hardware

The hardware is depicted in Fig. 7.36. A 3D-printed custom mount with a double active illumination LED-ring for a stereoscopic camera is attached to the robotic arm of an LBR 4+ (KUKA, Germany). A sling binds the impaired arm to the end effector of the light-weight robot. A hand-eye calibration (cf. chapter 7.7.2) determines the transformation between camera

coordinates and robot end effector. The used vision system hardware is the second iteration of the dynamic OTS design as discussed in chapter 7.8.1.

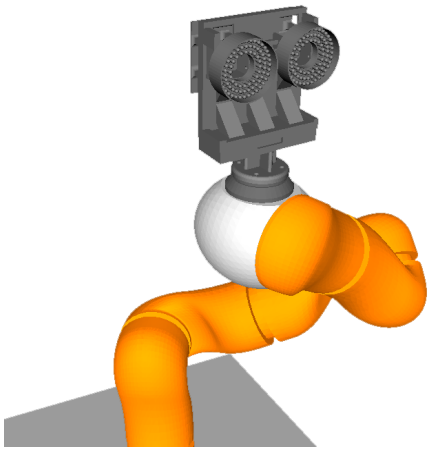


Fig. 7.36. **Robot with stereo camera system.** A custom 3D-printed mount is attached to the robot end effector. The mount holds the two cameras with their active ring-light illuminators. This allows the optical tracker to view the scene from a perspective close to the robot while moving with the light-weight arm.

The potential patient wears a tightly fitted sleeve on the healthy arm which is equipped with flat circular markers of 7 mm diameter. The retro-reflective film helps generating sub-pixel precise measurements to accurately calculate the relative pose between the robot and the healthy arm in order to move the robot and the impaired arm.
The robot control is run on a machine with an Intel Core i5 4690K at 3.5 GHz while the image processing and tracking is executed within the FRAMOS Application Framework (FRAMOS Imaging Systems, Germany) on an Intel Core i7 960 at 3.2 GHz. In order to determine the overall system accuracy, we use the external optical tracking system Polaris Vicra (NDI, Canada) as ground truth in all the consecutive experiments. To test the latency of the prototype, a second robot UR-6-85-5-A (Universal Robots, Denmark) is leveraged.

7.9.2.2. Tracking & Robot Control

The tracking algorithm runs two image processing pipelines for the marker extraction of the individual camera images in parallel threads and performs Algorithm 7.3 for pose estimation. The marker ensemble attached to the sleeve of the patient is trained with a fast one-to-multi-shot learning procedure (cf. section 7.5.1) to flexibly use patient-specific sleeves and ensure consistent tracking quality. For this, the patient is asked to place the healthy arm in a comfortable base position. Once **teaching mode** is active, all sleeve points are transformed into a position-independent reference frame that is initiated from a translation of the camera coordinates into the centroid of the first observation and adjusted to the patient needs. All consecutive measurements are registered to the initial point cloud while robustly enlarging the model with outlier rejection and parameter adjustments. The patient or medical expert that explains the system can therefore individually decide the training duration while neither arm movement nor marker occlusions effect the teaching result before the start of the robotic manipulator.

Robotic control is based around the Fast Research Interface (FRI) provided by the robot manufacturer (KUKA, Germany). We extend this for the path planning by the open-source robot operating system ROS[38] and communicate via OpenIGTLink between the two machines as discussed in chapter 7.6. The system uses the continuously updated TF library[39] to determine the reference frames in real-time for both the healthy arm and the robot. The control loop plans a trajectory that aligns the end effector with the virtual reference frame from the training stage, thus maintaining a static pose between the impaired and healthy arm. We use OMPL[40] and MoveIt![41] for self-collision avoidance and path planning where a variant of RRTConnect[42] is used as a planning backbone. The stochastic nature of the algorithm could impose a safety issue in proximity to the patient. Thus, we control the planned trajectories of each joint and reject path planning if a joint moves beyond a threshold close to the user. The final **safe trajectory** is then asynchronously pushed to the joint-trajectory-action controller in ROS and forwarded to FRI where trajectory updates happen every 40 ms. To ensure acceleration continuity, a quintic spline interpolation with the current state is performed by the controller. As an additional safety measure, both the acceleration and velocity of the robotic arm are limited to 0.8 m/s^2 and 0.24 m/s respectively. The overall process control is depicted in Fig. 7.37 and enables a smooth dynamic motion closely reflecting the natural movement of the patient's healthy arm.

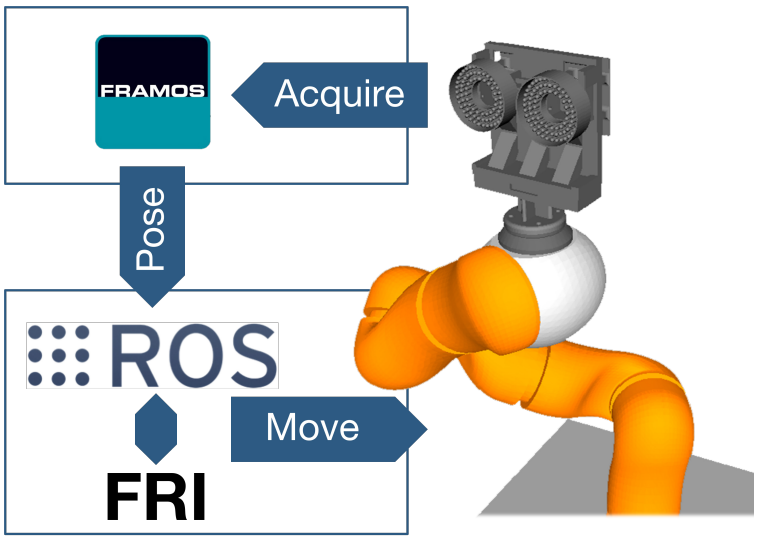


Fig. 7.37. **Process control loop.** The calibrated stereo vision system attached to the robot end effector (top right) acquires synchronized images that are processed by the FRAMOS software framework (top left). The calculated pose is used for the path planning and robot control (bottom left) within ROS and FRI which determines the movements.

7.9.3. Experimental Validation

The **parameters for the visual tracking** algorithm are determined empirically as described in chapter 7.3.8. A pose estimate is considered successful if at least 50% of the marker points are

[38] Cf. Quigley et al. [338].
[39] Cf. Foote [118].
[40] Cf. Sucan, Moll, and Kavraki [398].
[41] Cf. Chitta, Sucan, and Cousins [67].
[42] Cf. Kuffner and LaValle [231].

detected and the pose error between estimation and observation is below a certain threshold. To allow for slight deformation of the sleeve, we relax this constraint to 3 mm RMSE. The initial learning stage that defines the model cloud is extended by an online learning stage which is triggered once the constraints are violated for a longer period.

The **accuracy of the system** is evaluated in two steps. At first, we perform a stereo calibration for the intrinsic and extrinisic parameters of the stereo system. The final mean reprojection error is measured as 0.29 pixels.
In a second step, the overall system accuracy (i.e. stereo calibration, pose estimation and hand-eye calibration) is calculated. An extended working volume of 600×600×400 mm^3 is chosen for this test. Since the repeatability for the forward kinematics of the robot pose according to the manufacturer is ±0.05 mm[43], we use this as ground truth to validate the system error. While observing the sleeve target with 10 markers in a static position on a table, we move the robotic arm along a planned trajectory within the working volume in steps of 20 mm and record the estimated poses. The RMSE for the pose fitting is 0.21 mm ±0.25 mm while the standard deviation for the translation error with respect to the robotic ground truth is calculated as 0.23 mm, 0.23 mm, 0.42 mm in x, y, and z direction of the camera: A precision not noticeable for the user during dynamic motion.
To measure the overall **robustness**, we firstly repeat the experiment and occlude up to 50% of the markers without significant deviation from these results. We then train the system in an example scenario with a user and record the RMSE of the pose estimates with respect to the learned sleeve configuration in various robot poses while keeping the robot static. The mean results are shown in Fig. 7.38. It can be noted that the error is below 0.5 mm within a 200 mm window around the training distance.

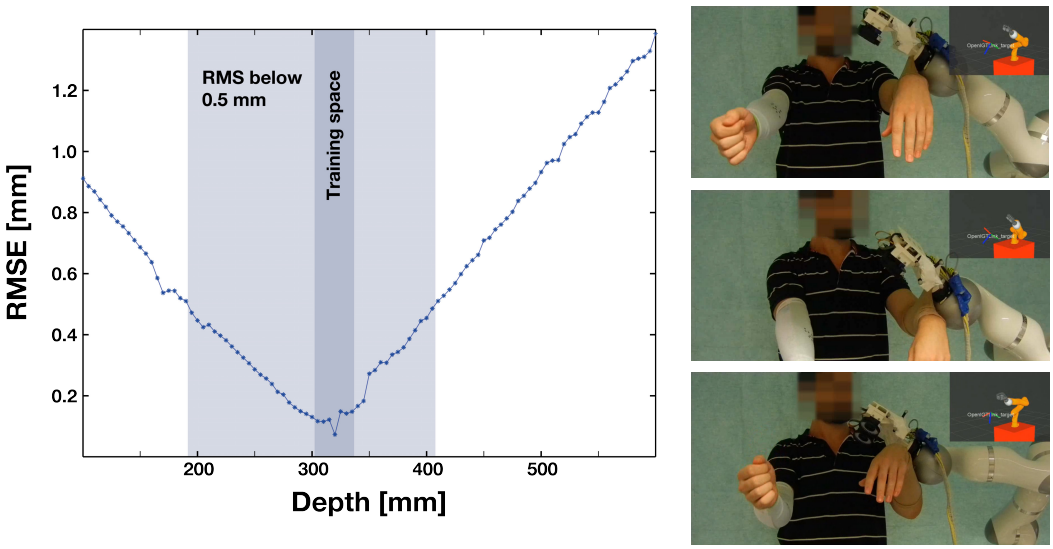


Fig. 7.38. **Robot accuracy evaluation and practical test.** The mean accuracy for the pose RMSE during a user test is shown on the left. The robot moves to several poses following the arm while the RMSE at different distances between 100 mm and 600 mm is recorded. The distance during marker training is highlighted together with the area where the error is below 0.5 mm. On the right, the robot arm is shown in various poses in a test with a medical expert where the robot is following the user movements.

[43] Cf. KUKA GmbH [232].

The **calculation time** for the pose estimation with the 10 test markers on the sleeve is 9.42 ms ±1.44 ms. To measure the **system latency**, we utilize a second (reference) robot holding a target which is observed by the application robot. Both robots are synchronized and co-calibrated. The application robot follows the reference robot while we record the trajectories. The setup and translational component of the motion is shown in Fig. 7.39 during a move. The delay is measured as the relative time difference between the start of the robot motion indicated with a vertical dashed line. We calculate an overall average latency of 318.70 ms.

A final experiment with two healthy medical experts targets the **usability of the system in movement therapy**. The test persons rest the left arm in the sling and are asked to relax it fully. Fig. 7.40 illustrates a possible test scenario. Both users then execute a series of movements with the other arm in all possible directions and are asked for feedback on their experience.[44] The testers were able to freely control the robotic arm without prior training and agree that the robot mimics the motion of the right arm smoothly. The control mechanism has been reported to be intuitive and steering the robot in this way found to be easy.

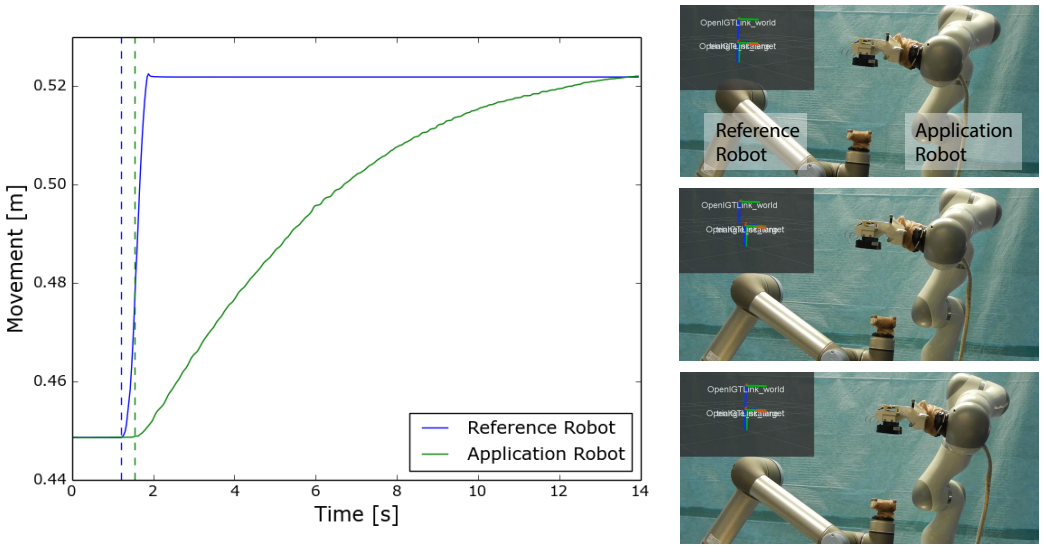


Fig. 7.39. **Robot movement latency test.** Two robotic arms are deployed to measure the latency of the proposed system. The reference robot holds a marker and moves to a set of new poses and the application robot follows. The recorded motion is illustrated on the left for one translation component. The reference robot is shown in blue between one of these movements. The application robot motion is illustrated in green. The delay of the beginning of the motion (vertical bars) shows the system delay. The slow smooth motion is due to the safety measures. The photos on the right show the robots in static positions (top), after the motion of the reference arm (middle) and after the motion of the application robot (bottom).

7.9.3.1. Discussion and Retrospective

The presented marker-based camera-in-hand tracking system for collaborative robotic movement is capable of steering an impaired arm with a healthy arm, thus enabling a first step in the direction of touch-free movement therapy. Previous robotic rehabilitation systems either use joystick control or exoskeletons[45] while our prototype is a **first demonstrator for natural**

[44] A video of the experiments is available for download at `http://campar.in.tum.de/Chair/PublicationDetail?pub=busam2015acvr`.
[45] Cf. Basteris et al. [15].

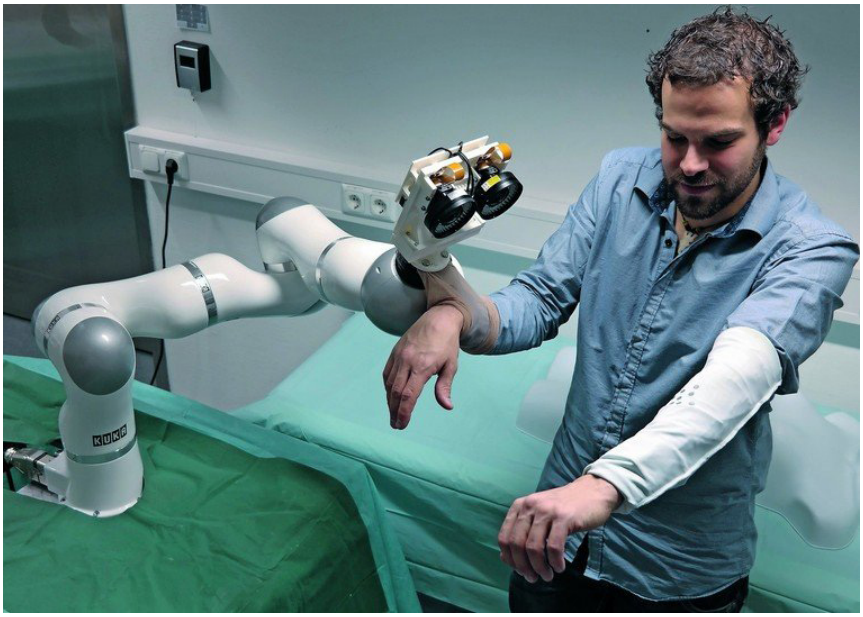


Fig. 7.40. **Usability test of movement therapy robot.** Illustrated is a configuration similar to the usability study. The healthy subject is attached to the robot with one arm and steers the movement with the other side.

motion steering without these constraints. The touchless real-time pipeline acts as an interface for the interaction between human and machine and thus directly addresses the inherent problem of robust recognition of the live human intent as pointed out by Rautaray et al. [343]. While the markers ensure a robust and accurate pose estimation and a safe robot control with a latency of only 318.70 ms and a precision above 0.5 mm in a considerably large working volume, they are also the main disadvantage of the current pipeline. However, similar performance for human-robot interaction is not achievable with marker-free methods.[46] Even though there are similar performing marker-based tracking methods such as the ones presented by Garrido-Jurado et al. [137], our self-adhesive fiducial system with rapid online training and controllable rejection basin is more flexible than commonly used marker systems such as the ones from Naimark et al. [303] or Zhang et al. [476] and calculates poses faster than other systems.[47] The ability to learn on the fly and re-adjust for sleeve motion or new users also helps to quickly adapt the configuration.

While paving the way for more natural collaborative systems, this first concept demonstrator allows only for simple movements and needs necessary extensions in order to compete with available rehabilitation robots as described by Basteris et al. [15]. The current movements are restricted to three directions of the arm that fully steer the hand movement while leaving some slack for the rotational part with a loose fixation of the hand at the end effector. Also no extra model is utilized for joints such as the elbow. Moreover, the force applied to the impaired arm is currently not considered and the speed is limited for safety reasons. The tracking algorithm is capable of tracking multiple targets at once and thus can be extended with joint-specific marker ensembles in a multiple target tracking scenario. Furthermore, the current manipulator is a standard robotic arm and would need some customization in order to function similar to an exoskeleton where more complex individual muscle movements can be mirrored.

[46] Cf. Siddiqui and Medioni [376] as well as Buehler et al. [44].
[47] Cf. Fiala [112].

The camera-in-hand solution of our tracker circumvents the line-of-sight issues commonly appearing for static outside-in systems and the two-camera design may eventually contribute towards patient acceptance as the stereo-setup gives a more humanoid appearance to the robot than a single monocular camera.
A clinical study for upper limb movement therapy can help to eventually understand the clinical impact of such a system with the above mentioned extensions and a comparison to existing robotic and non-robotic approaches can lead to a conclusion for the impact of natural gesture-control and user participation for active rehabilitation in impaired hemiparetic patients.

With this example, we conclude the description of our marker-based pose estimation pipeline and consider the next logical constraint relaxation by moving into the domain of marker-free approaches.

8

Markerless Pose Estimation

> *Jedes enthält etwas als Objekt in sich,*
> *obwohl nicht jedes in gleicher Weise.*
>
> – **Franz Brentano**
> (Psychologie vom empirischen Standpunkte)[1]

The previous chapter described a 3D computer vision pipeline that can be used to accurately measure rigid body motions of marked objects in space. We have seen how to utilize modern camera hardware to build such a system and how we can co-calibrate it with other sensors to eventually apply the system in a medical use case. Even though the camera-in-hand setup described in chapter 7.9 moves together with the robot and accurately measures the relative pose between camera and marked object, a free line of sight between vision system and markers is essential.

In this chapter, we focus on both these challenges, the line of sight and the markers. We first turn our previous approach at its head and instead of pointing the cameras towards the object of interest, we fix the vision system rigidly to the object and point the cameras away from it into the surrounding environment (cf. section 8.1). In our medical scenario, for instance, this can be an operation theatre. Making use of the miniature IO design developed in section 7.8.1, we design an **inside-out tracking** approach that allows for **camera pose** estimation relative to the surrounding. Since the vision system is fixed to the object of interest, the object motion is identical with the camera movements.

For reasons such as space, environment and hardware wiring, it is, however, not always possible to fix a miniature camera onto the object. While markers can facilitate high accuracy (cf. chapter 7.4.1), their placement or appearance may also be impractical in scenarios such as robot grasping in production lines or in certain augmented reality (AR) setups. Depending on the use case, more relaxed accuracy requirements allow even for outside-in approaches without markers. Consecutive thoughts address **markerless object pose** estimation where marker properties are replaced by object properties (cf. section 8.2) such as geometrical information or natural features arising from the object appearance and the help of 3D models.

[1] "Each includes something as object within itself, although they do not all do so in the same way.", Franz Brentano: Psychologie vom empirischen Standpunkte. Leipzig: Duncker and Humboldt, 1874, S.124f.

8.1. Inside-Out Tracking

We first analyse the possibility of using a miniature hardware vision system for object pose estimation without markers in a medical use case scenario to show its practical relevance and exemplify the advantages for 3D ultrasound compounding.

Continuous pose estimation for medical instruments and surgical sensors is the de facto standard of modern interventions and an essential tool for 3D ultrasound compounding where 2D ultrasound slices are combined with their spatial pose information over time in order to computationally reconstruct the underlying 3D anatomy. External optical trackers as commonly used suffer from line-of-sight issues as discussed in chapter 7.8. This problem becomes evident in particular if the region of interest is difficult to access. The **inside-out tracker** we propose here aims to circumvent these obstacles and provides a practical solution. Simultaneously localizing the camera while reconstructing the operating room allows for markerless tracking where the camera pose is determined by its surroundings. We enable ultrasound probe tracking with the help of **visual SLAM** in an interventional scenario. A miniature vision sensor with multiple modalities (monocular, stereo, active depth sensing) is mounted on an ultrasound probe and the cameras point into the room. It is used to relocalize its position and orientation within an adaptive map of the operating room and tested in the interventional context of transrectal 3D ultrasound (TRUS). State-of-the-art algorithmic pipelines for direct and feature-based visual SLAM as well as a commercial optical tracker are compared to each other both qualitatively and quantitatively regarding their relevant performance for anatomical volume reconstruction and pose accuracy. A robotic manipulator serves as a ground truth pose generator to identify pose variations and to compare all approaches.
Indirect binocular SLAM based on feature tracking shows the most promising results. The system is tested in extensive experiments that reflect the challenging clinical environment present during prostate ultrasound biopsies.

8.1.1. Status Quo & Medical Motivation

Tracking systems provide the rigid body transformation of one (or multiple) targets with respect to a common reference frame which can be the tracker itself, the patient or any pre-calibrated coordinate frame. Medical instruments and tools are tracked to enable certain forms of medical imaging and computer aided interventions. Accurate tracking information is a crucial element for reliable diagnostic analysis in medical applications such as **3D ultrasound compounding**.
Mechanical trackers use the physical description of their kinematic chain to provide high precision measurements.[2] Due to their expensive components and bulky setup, they are unsuitable for dynamic clinical usage where flexibility is required.
In contrast to mechanical solutions, **electromagnetic tracking** systems are more flexible to use, but they provide a limited accuracy and are known to interfere with metallic objects in proximity to the working volume which by itself is comparably small which restricts the user to movements in a limited space.[3]

[2] Cf. Hennersperger et al. [170].
[3] Cf. Kral et al. [227].

Optical tracking systems (OTS) do not suffer from these disadvantages and enjoy widespread use. Usually, a set of active or passive infrared markers is attached to the target and tracked by a static external stereo camera. While the accuracy is relatively high (cf. section 7.4) and the working volume considerably large under optimal conditions, these systems require a free line of sight to the target. In cases with critical line-of-sight restrictions, the typical outside-in systems are repositioned multiple times during an intervention in case of tracking loss as occlusions of markers can occur while they are in use. This impacts the medical workflow severely and can critically increase intervention time and stress on both patient and physician.
More robust systems such as the tracker presented by Busam et al. [49] work also for incomplete target visibility with partly occluded marker setups. Marker-visibility can be radically reduced in different practical cases where prominent examples are freehand SPECT [173] and freehand 3D ultrasound [109] which critically demand robust and reliable tracking solutions. The combined requirement for both accuracy and flexibility in medical 3D imaging are addressed by a series of ideas. The system presented by Esposito et al. [101] uses a collaborative robotic arm to assist in a medical intervention under ultrasound guidance with a camera-in-hand system. The approach offers the advantage to automatically reposition the marker-based tracker in a dynamical setup, however, it does not resolve the line-of-sight issue.
To reduce the need for markers in 3D ultrasound, Sun et al. [399] attempt to leverage specific localized skin features instead. While these features can be a promising guidance, the system is designed for a specific anatomy that provides rich enough information.

Our approach is pragmatic. Compared to previous ideas, we aim to provide a generalizable tracking system that works autonomously without manual or case-specific feature selection while being simple to setup and use even for novice users. We leverage the hardware iterations from section 7.8 and make use of a **miniaturized camera system** that can be placed on the object of interest where the image and video data provides us with the foundation to extract features in the room that can be used to track the camera and estimate the object pose.

8.1.2. Inside-Out Object Tracking

We discuss the inside-out tracking system in the use case of transrectal 3D ultrasound compounding as illustrated in Fig. 8.1 which shows the typical setup for a prostate fusion biopsy. The ultrasound device is equipped with a miniature camera system whose line of sight points away from the patient towards the rich features of the clinical environment.

8.1.2.1. Inside-Out Tracking System

To establish a generic inside-out tracking approach that is robust to different environments, the ad hoc extracted geometric information from the scenery is vital for its pose estimation quality. In order to realize such a system, we utilize a visual method to simultaneously map the surrounding and localize the camera within the adaptive reconstruction (SLAM).[4] We propose to rely on characteristic features of surrounding structures to enable a cumulative construction

[4] Cf. Mur-Artal and Tardós [298].

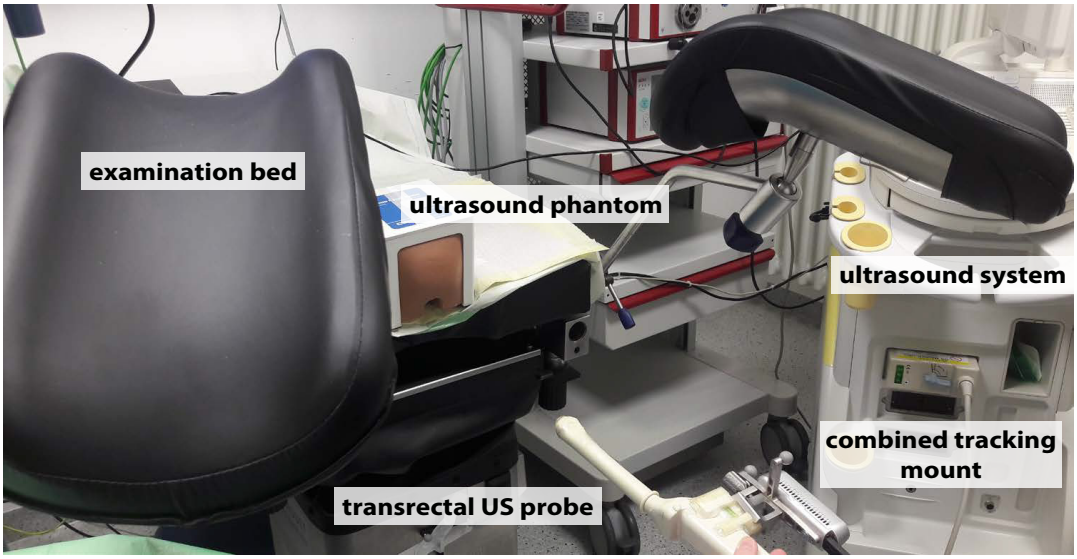


Fig. 8.1. **Interventional setup for transrectal ultrasound fusion biopsy.** The figure shows the examination bed with a transrectal ultrasound phantom. The ultrasound system (right) is placed such that the cabling allows for the intervention. We propose the combined tracking mount (lower right) on the handle of the ultrasound probe that adds a miniature camera to the traditional rigid markers used for classical outside-in tracking.

of the map within a previously unknown environment observed by the sensor.
SLAM methods can be separated in direct and feature-based methods, both with its characteristic drawbacks and benefits. For **direct SLAM** approaches, the entire image information is taken into account. The pioneering systems LSD-SLAM[5] and DSO[6] are prominent examples of this kind of approach. While starting from the direct sensor measurements, the pipelines may lead to erroneous poses under changing lighting conditions, require good initialization and are not able to recover poses correctly for rolling shutter cameras. In contrast, **feature-based methods** rely on extracted feature points. Prominent examples such as ORB-SLAM[7] and its extensions from Mur-Artal et al. [298] and Campos et al. [58] lead to more stable tracking behaviour during illumination changes, but require a minimum amount of structure within the scene.
Structure from motion (SfM) pipelines such as SfM learner[8] and the monodepth pipeline[9] as well as their successors are also able to determine the poses between consecutive frames in a video sequence to incrementally build a pose trajectory. However, full SLAM pipelines such as the ones mentioned above are different. They provide an entire framework including methods for loop closing, pose graph optimization and re-initialization which are necessary to establish poses also after sight-loss, occlusion or incremental drift.

While different additional modalities can be used to extend visual SLAM for instance with the inclusion of an IMU (cf. section 7.8.2), we rely here on **binocular stereo** vision. This has several advantages over monocular or active depth sensing methods.[10]
The **inside-out optical tracking system** (IO-OTS) proposed here is thus relying on marker-

[5] Cf. Engel, Schöps, and Cremers [98].
[6] Cf. Engel, Koltun, and Cremers [97].
[7] Cf. Mur-Artal, Montiel, and Tardós [297].
[8] Cf. Zhou et al. [485].
[9] Cf. monodepth by Godard, Mac Aodha, and Brostow [144] and monodepth2 by Godard et al. [145].
[10] Cf. Mur-Artal and Tardós [298].

free image data and its poses are retrieved from the SLAM backbone which allows for flexible use in unknown environments without the need for markers. To assess the quality of the IO-OTS, we experimentally analyse the tracking accuracy both quantitatively and qualitatively in comparison to a commercial tracking solution with the help of robotic ground truth movements and investigate the sample use case of freehand 3D ultrasound imaging.

To the best of our knowledge, the proposed prototype is the first system that is using SLAM for inside-out tracking to estimate the object pose in an interventional setup which is exemplified with 3D TRUS. Pointing the vision system away from the patient into the quasi-static room turns the tracking idea at its head and enables a constant update of the OR map while being more robust to partial occlusions. The line-of-sight restriction from outside-in trackers is thereby tackled through the use of wide-angle lenses.
Aside of an improved rotational accuracy, the system also omits time-consuming intraoperative hardware repositioning which is necessary if external outside-in tracking systems are used and a marker visibility during the entire medical procedure is not possible. While the method dynamically adjusts the map to environmental changes it also paves the way for seamless multi-sensor fusion where a common map can be shared amongst multiple sensors or edge devices to enable fast replacement and information combination.

8.1.2.2. Camera & Object Pose Estimation

Pose estimation with a moving camera in SLAM setups usually determines the camera reference frame relative to a world anchor. We, however, are interested in the use case of freehand ultrasound where the US sensor gives the coordinate system of interest and thus need a calibration step in form of a rigid transformation as the camera is rigidly attached to the ultrasound transducer. Hereafter, we discuss our inside-out setup for object pose estimation including the calibration chain, detail and justify pipeline choices and finally describe the processes for evaluation.

In interventional 3D ultrasound imaging, we seek the rigid transformation of the transducer with respect to a common world reference frame. Let us therefore denote with ${}^{B}\mathbf{T}_{A}$ the transformation from A to B and let the desired world reference frame be called W. Thus, the transformation ${}^{W}\mathbf{T}_{US}$ indicates the ultrasound image (US) in world coordinates. In contrast to an outside-in tracker, the inside-out camera system is rigidly attached to the ultrasound probe and provides a direct relation to the world reference frame with

$$ {}^{W}\mathbf{T}_{US} = {}^{W}\mathbf{T}_{RGB} \cdot {}^{RGB}\mathbf{T}_{US} \tag{8.1} $$

where ${}^{W}\mathbf{T}_{RGB}$ is the transformation which needs to be determined by the SLAM-based tracking algorithm while a conventional **3D ultrasound calibration** method[11] can be leveraged to determine the static transformation ${}^{RGB}\mathbf{T}_{US}$.

Our **miniature camera setup** is shown in Fig. 8.2 and described in detail in sections 8.1.3 and 8.1.4. The hardware provides several image modalities commonly used for visual SLAM. Even though requiring the least amount of input data, monocular SLAM approaches need a

[11] Cf. Hsu et al. [185].

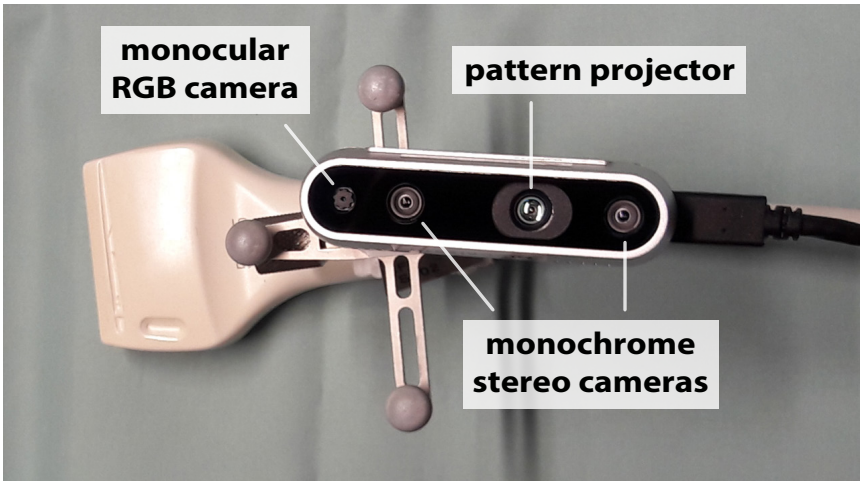


Fig. 8.2. **Combined miniature camera mount on ultrasound transducer.** The picture shows the outside-in rigid body marker with its metal frame which can be partly seen with the three mounted retro-reflective marker spheres. Additional to it, a miniature multi-modal RGB-D sensor is mounted to the frame of the example ultrasound probe. The sensor provides a monocular colour camera with rolling shutter (left), a binocular monochrome greyscale global shutter stereo camera pair which is sensitive to the near infrared (IR) spectrum as well as an IR pattern projector that is able to project a static dot pattern into the scene for additional texture.

considerable amount of translation without much rotation as an initialization in the beginning of the movement which is a restriction we want to omit to ease the use for medical users. Additional to this, it accumulates drift errors over time which affect the pose estimation quality and the absolute scale of the reconstruction as well as the pose trajectory remain unknown as the necessary pairing of two video frames used for triangulation induce a non-deterministic baseline initialization in practice for freehand motion. The latter makes it unsuitable for metric measurements and the composition with the ultrasound calibration which ultimately prevents the system from providing metric object poses.
The camera provides an additional depth map in real-time with an efficient implementation of SGBM[12]. The depth accuracy of this RGB-D camera, however, is too noisy for high accuracy measurements. Thus, we rely on a stereo setup and calibrate the cameras prior to our use to determine their fixed baseline and intrinsic parameters as described in chapter 4.2. This additionally allows for feature triangulations in rotation-only motions such as the ones common in 3D TRUS.

To understand and **compare** the suitability of **different SLAM-methods** for the IO tracker, we run our evaluation with publically available implementations of prominent pipelines. As a representative of a feature methods, we apply the popular ORB-SLAM2[13] pipeline in its stereo version while we rely on stereo DSO[14] as a direct method.
The transrectal ultrasound setup with a phantom is shown in the clinical environment in Fig. 8.3 together with the inside-out camera view of the left monochrome greyscale sensor. The detected features of ORB-SLAM and reprojected sparse map points from DSO are augmented for visualization purposes to show that a standard operation theatre provides sufficient visual

[12] Cf. Hirschmüller [177].
[13] Cf. Mur-Artal and Tardós [298].
[14] Cf. Wang, Schworer, and Cremers [444] who present the stereo version of the popular Direct Sparse Odometry (DSO) pipeline from Engel et al. [97]. Note that the standard implementations of DSO and LSD-SLAM from Engel et al. [98] cannot be used here as they are restricted to the monocular camera case.

information for pose estimation. Surgical environments are scenes with multiple devices and medical tools that provide visual features for camera solutions. The equipment around the examination bed is ideal for inside-out tracking systems as feature response maps and edge detectors provide stable responses while the same medical devices are challenging occluders and obstacles for outside-in trackers.

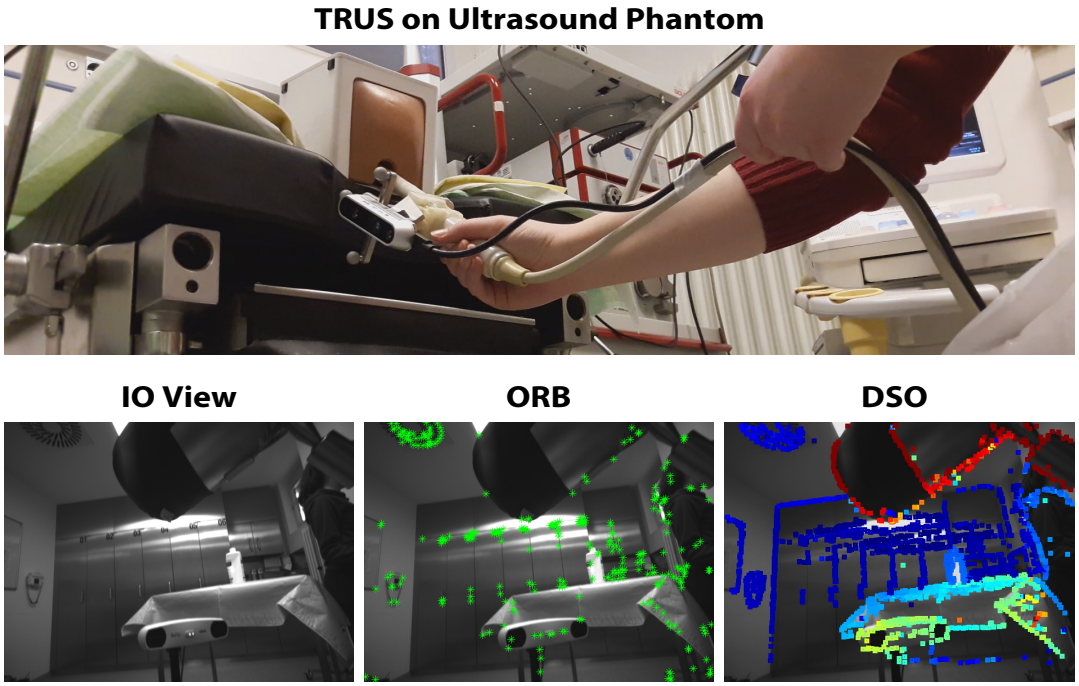


Fig. 8.3. **3D TRUS phantom acquisition.** Top: An ultrasound volume scan of a prostate phantom is performed in the prostate biopsy operating room. The miniature camera is mounted together with the rigid body marker of an outside-in system on a transrectal ultrasound probe. The images in the lower row show the input and extracted data from the two SLAM methods. Bottom left: The image shows the left view of the stereo pair from the inside-out tracker pointing away from the patient situs into the room. The response map of ORB-SLAM (bottom, centre) shows the detected ORB features augmented in green. Bottom right: The reprojected sparse map points from DSO are colour coded depending on their distance to the camera (warmer colours are closer).

8.1.3. Tracker Validation

We now **evaluate the accuracy** of the inside-out tracking approach in comparison to a commercial medical outside-in tracker and assess the advantages of feature-based SLAM against a direct approach. A robotic manipulator thereby serves as ground truth. In a consecutive qualitative analysis we then further exploit the clinical benefit of the system for 3D ultrasound compounding.

8.1.3.1. Evaluation Setup

To quantify the tracking accuracy, we **attach** our combined **tracking mount** rigidly **to a robot** end effector as shown in Fig. 8.4 together with the associated reference frames. The complete setup of all coordinate frames relevant for our evaluation is illustrated in Fig. 8.5.

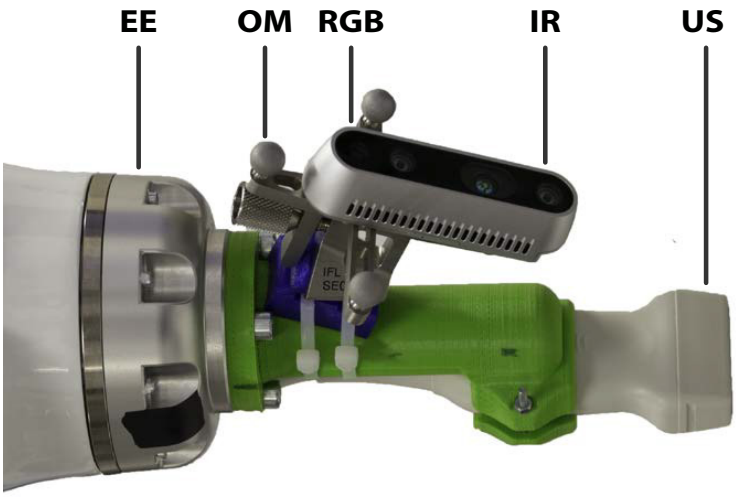


Fig. 8.4. **Pose accuracy evaluation mount.** The miniature camera is attached on the combined mount with the optical rigid body marker (OM) at the robot end effector (EE). The mount additionally attaches both inside-out camera and ultrasound transducer to one another to perform also qualitative evaluation with 3D US scans. Shown are the two camera reference frames for RGB (RGB) and infrared stereo pair (IR). The latter is incident with the right IR camera. We also show the coordinate frame notation for the ultrasound probe (US).

The camera model[15] and **calibration** method[16] follow the ideas of Zhang [481] with which we calibrate both the monocular RGB (RGB) as well as the stereo cameras (IR1, IR2). We use a pinhole camera with two radial distortion coefficients and calculate the stereo geometry with OpenCV.[17] The hand-eye calibration[18] between inside-out tracking camera (IR) and robotic end effector (EE) is performed using the algorithm of Tsai et al. [421] in the implementation within the framework of Marchand et al. [275] in eye-in-hand mode. Eye-on-base is used to calibrate the external optical tracking system (OTS) to the robot base (RB).

We initially evaluate the tracking accuracy of various inside-out methods in comparison with the external optical tracker. A 7 DoF robotic arm KUKA iiwa (KUKA Roboter GmbH, Augsburg, Germany) serves as the tool to generate a ground truth motion with which we can compare the different systems. This robotic manipulator guarantees for a positional reproducibility of ±0.1 mm which will turn out to be one to two orders of magnitude more precise than the optical algorithms we test. With special attention to the medical use case, we compare to a commercially available optical tracking system (Polaris Vicra, Northern Digital Inc., Waterloo, Canada) based on spherical markers and infrared illumination that is commonly used in medical applications. As inside-out tracking sensor hardware, we utilize an intel RealSense Depth

[15] Cf. chapter 4.1.
[16] Cf. chapter 4.2.
[17] Cf. Bradski [38].
[18] Cf. chapter 7.7.2.

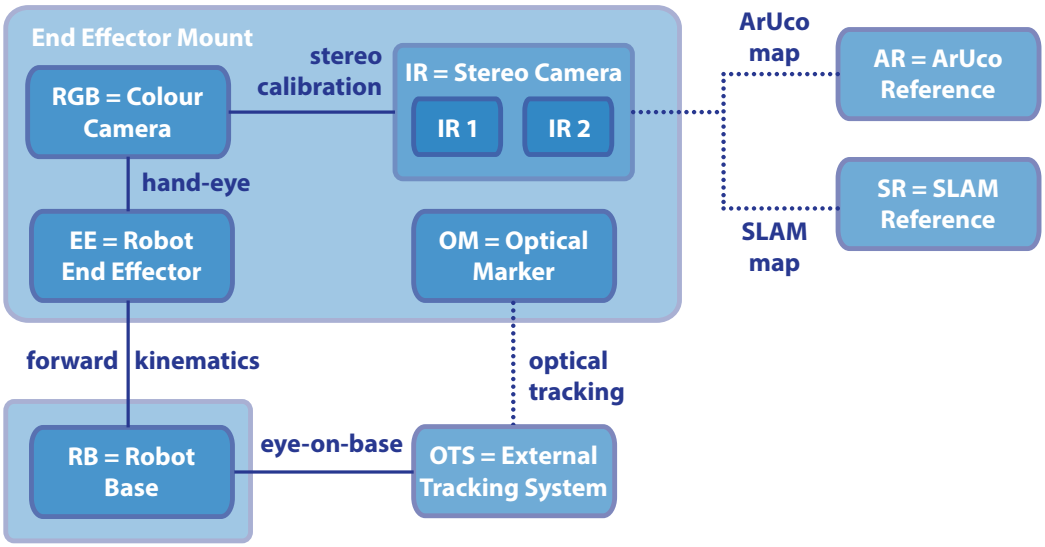


Fig. 8.5. **Tree of reference frames for accuracy evaluation.** The different reference frames are all co-calibrated or connected via pose estimation approaches. The robot base (RB) calculates its end effector (EE) position with the robot forward kinematics. The monocular colour camera (RGB) is calibrated with the end effector via hand-eye calibration (eye-in-hand variant) and itself co-calibrated via a stereo calibration with the binocular monochrome infrared stereo setup (IR) whose reference frame is incident with the right IR camera (IR 1). All three cameras are calibrated for their intrinsic parameters and corrected for distortion effects. The ArUco (AR) and SLAM (SR) methods map the room and allocate their world reference coordinate frames. An external optical tracking system (OTS) is calibrated with the robot base via hand-eye calibration (eye-on-base variant) and provides itself the pose for the optical marker (OM) additionally attached to the end effector mount.

Camera D435 (Intel, Santa Clara, CA, USA) which provides synchronized images with a global shutter binocular greyscale stereo system and a rolling shutter RGB camera in miniature format.

We evaluate tracking systems for inside-out tracking with both feature-based and direct SLAM methods against an ArUco[19] marker-based inside-out method with 16 × 16 cm markers and a classical outside-in system. The miniature marker with both the optical outside-in target as well as the miniature camera is attached to the robot as shown in Fig. 8.4 and the robot is controlled via the Robot Operating System (ROS) while a separate machine acquires the vision data with the intel RealSense SDK[20]. All systems are synchronized to a common NTP-server with a temporal offset of $t < 1$ ms and images are acquired with a resolution of 640 × 480 pixels at a frame rate of 30 Hz. Poses for RGB camera and tracking target are communicated via TCP/IP over Ethernet with the publicly available library S.I.M.P.L.E.[21] while image processing and acquisition is done on a machine with an intel Core i7-6700 CPU, 64bit, 8GB RAM which runs Ubuntu 14.04.
The surrounding conditions are equivalent to a conventional TRUS. We therefore cover the same scanning volume and tracking time while the distance to the optical systems is identical to the medical procedure. All evaluation results hereafter therefore directly reflect this medical

[19] Cf. Garrido-Jurado et al. [137].
[20] Cf. https://github.com/IntelRealSense/librealsense.
[21] Cf. https://github.com/IFL-CAMP/simple.

use case and the error analysis provides insight into all involved components. The entire setup is shown in Fig. 8.6.

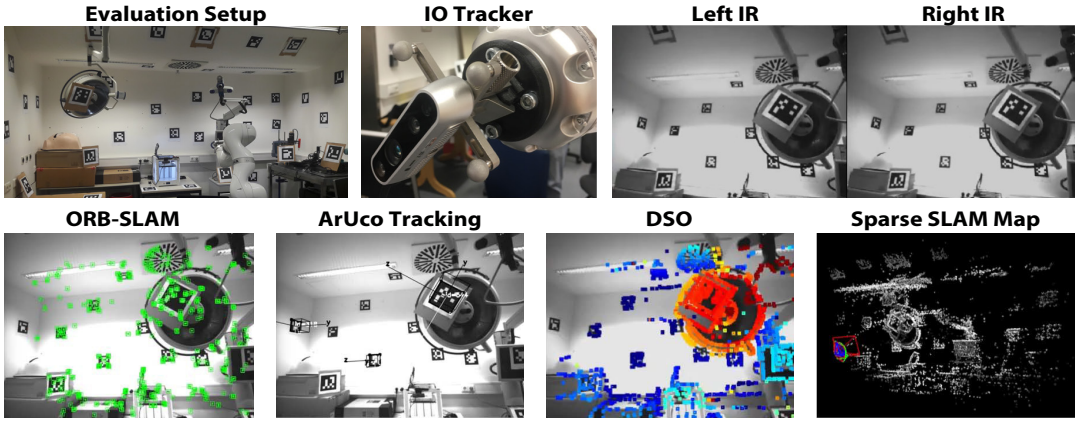


Fig. 8.6. **Evaluation setup and response maps.** The accuracy test evaluation is done in an operation room (top left), where the combined mount of an IO Tracker and rigid body marker with spherical IR markers (top centre) is attached to the robot end effector. The inside-out stereo view (top right) illustrates the field of view of the monochrome camera pair. The responses from ORB-SLAM, ArUco marker tracking and DSO necessary to run the different SLAM algorithms are shown projected onto the image plane at the bottom from left to right together with a sparse reconstruction.

8.1.3.2. Tracking Accuracy

The hardware setup described before is now used to **evaluate the tracking accuracy quantitatively**. The robotic manipulator is put into gravity compensation mode to serve as a ground truth pose acquisition system while the end effector is manipulated by a medical expert. The test person performs a series of motions for which the pose sequences are recorded with all systems and the forward kinematics of the robotic arm is used to generate a ground truth (GT) sequence of the movements.

In order to compare the tracking error, we transform all pose sequences in a common reference coordinate system. We choose the RGB coordinate system of the miniature camera at the end effector as our common reference (see Fig. 8.5). The transformations are given by

$$
\begin{aligned}
{}^{\mathrm{RGB}}\mathbf{T}_{\mathrm{GT}} &= {}^{\mathrm{RGB}}\mathbf{T}_{\mathrm{EE}} \cdot {}^{\mathrm{EE}}\mathbf{T}_{\mathrm{RB}} && (8.2)\\
{}^{\mathrm{RGB}}\mathbf{T}_{\mathrm{SR}} &= {}^{\mathrm{RGB}}\mathbf{T}_{\mathrm{EE}} \cdot {}^{\mathrm{EE}}\mathbf{T}_{\mathrm{RB}} \cdot {}^{\mathrm{RB}}\mathbf{T}_{\mathrm{IR}} \cdot {}^{\mathrm{IR}}\mathbf{T}_{\mathrm{SR}} && (8.3)\\
{}^{\mathrm{RGB}}\mathbf{T}_{\mathrm{AR}} &= {}^{\mathrm{RGB}}\mathbf{T}_{\mathrm{EE}} \cdot {}^{\mathrm{EE}}\mathbf{T}_{\mathrm{RB}} \cdot {}^{\mathrm{RB}}\mathbf{T}_{\mathrm{IR}} \cdot {}^{\mathrm{IR}}\mathbf{T}_{\mathrm{AR}} && (8.4)\\
{}^{\mathrm{RGB}}\mathbf{T}_{\mathrm{OTS}} &= {}^{\mathrm{RGB}}\mathbf{T}_{\mathrm{EE}} \cdot {}^{\mathrm{EE}}\mathbf{T}_{\mathrm{RB}} \cdot {}^{\mathrm{RB}}\mathbf{T}_{\mathrm{OTS}} \cdot {}^{\mathrm{OTS}}\mathbf{T}_{\mathrm{OM}}. && (8.5)
\end{aligned}
$$

With these, the accuracy of the tested systems such as the optical tracking system (OTS), the ArUco tracking (AR) as well as the two SLAM-methods (SR), namely ORB-SLAM and DSO, can be directly compared. In order to minimize other influences, a series of hand-eye calibrations has been done which has proven that the residuals from the robotic hand-eye calibration is negligible in comparison to the overall tracking error.

We acquire a series of 5 sequences with a total of 8'698 poses and summarize the resulting pose errors in Fig. 8.7. The translation error is measured in millimeters and the angle error indicate the deviation of the rotation axis in degrees compared to the robotic ground truth.

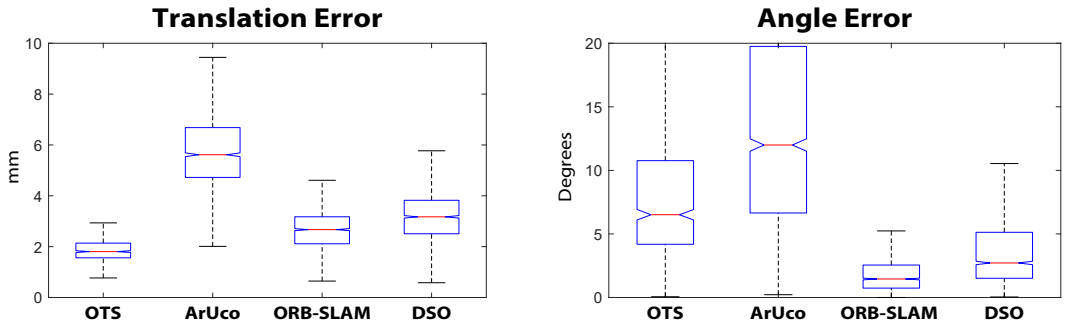


Fig. 8.7. **Tracking error comparison.** The box plots show the error spread for translation and angle error of the different tracking methods (minimal and maximal measured error indicated by the black whiskers) together with the median error (red) and the first and third quartiles (blue). The outside-in optical tracking system (OTS) is compared to inside-out ArUco marker tracking (ArUco), ORB-SLAM and DSO. The left plot shows the translation error distribution in millimeters while the right plot illustrates the angle error in degrees.

The external optical tracking system provides the best translational accuracy with an error of 1.90 ± 0.53 mm. We observe that the SLAM methods are comparable with errors of 2.65 ± 0.74 mm for ORB-SLAM and 3.20 ± 0.96 for DSO. Inside-out tracking with ArUco markers gives less accurate translation results with a residual error of 5.73 ± 1.44 mm. Analysing the rotational result gives an interesting insight. The markerless inside-out methods provide better results than the optical outside-in tracker with errors of 1.99 ± 1.99° for ORB-SLAM, followed by 3.99 ± 3.99° for DSO while the OTS angle error is measured as 8.43 ± 6.35°. The angular results of ArUco tracking are rather noisy with a residual of 29.75 ± 48.92°.

The evaluation suggest that the ArUco approach is viable for approximate pose estimation, but not suitable for accurate tracking while the proposed **IO-OTS solutions outperform** the commercial **outside-in optical tracking system in terms of rotational accuracy** and show valuable results also for translational motion. An explanation for the accuracy gain in rotation lies in the miniature system design of the inside-out tracking cameras where small rotations around an arbitrary axis close to the optical centre lead to severe changes of the field of view as illustrated in Fig. 8.8. This **inside-out rotation leveraging effect** is of particular interest in applications that require a significant amount of rotational motion such as 3D TRUS where an improvement of rotational accuracy directly translates to better 3D ultrasound reconstructions. We therefore consecutively replace the robot with an ultrasound transducer and perform a qualitative analysis of the system for manual 3D ultrasound scans comparing also the resulting ultrasound compounding.

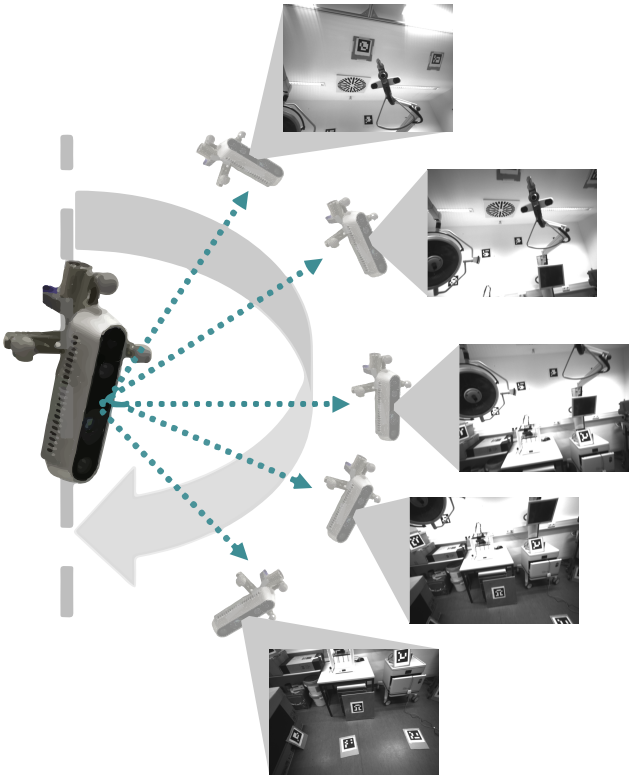


Fig. 8.8. **Rotation leveraging effect of inside-out tracking.** A small motion such as a rotation close to the optical centre of the miniature inside-out sensor (left) results in severe changes of the field of view and the image content as shown on the right. This eases the estimation of the pose for the sensor and thus the object rigidly attached to it. At the same time, the motion of the spherical optical markers co-attached to the sensor is minor. These are used for pose estimation by an optical outside-in tracking system.

8.1.4. Inside-Out 3D Ultrasound

To evaluate the practical use of the tracking method, **we compare the quality of ultrasound compoundings** based on poses of our markerless inside-out tracking and the commercial outside-in optical system using retro-reflective spherical markers. The ultrasound image plane is co-calibrated to the other tracking reference frames shown in Fig. 8.4 with the open source ultrasound toolkit Plus[22] for which a series of correspondence pairs are provided using a tracked stylus pointer to retrieve the desired rigid displacement.

Based on the favourable tracking characteristics in our accuracy tests, we choose the ORB-SLAM method to run the markerless inside-out tracking and assess the reconstruction quality for a 3D ultrasound compounding of a phantom with a spherical structure inside. For imaging purposes, we integrate a 128 element linear transducer (CPLA12875, 7 MHz) on the combined tracking mount and connect it to a cQuest Cicada scanner (Cephasonics, CA, USA). The publicly available real-time data acquisition framework SUPRA[23] is deployed in conjunction with ROS, and the calibration is performed using a stylus that is calibrated with a pivot calibration as described in chapter 7.7.1.
We evaluate and compare the results of a sweep acquisition between the outside-in optical

[22] Cf. Lasso et al. [236].
[23] Cf. Göbl, Navab, and Hennersperger [143].

tracker (OI) and the markerless inside-out method (IO) while we synchronize the tracker poses of both systems to the image acquisitions with the interpolation techniques provided by Busam et al. [50] which we detail in section 9.1.
A qualitative comparison of the compounding results as calculated with the ImFusion Suite (ImFusion GmbH, Munich, Germany) is shown in Fig. 8.9. It can be clearly seen that the rotational accuracy advantages of the inside-out tracking approach improves the boundary quality of the spherical structure inside the phantom which results in a smoother surface of the reconstructed 3D compounding. A video analysis[24] with a temporal slicing of the compounding results emphasizes this point and helps to understand the importance of rotational motion by showing the rotational motion of a sweep for a prostate phantom with the typical transducer used.

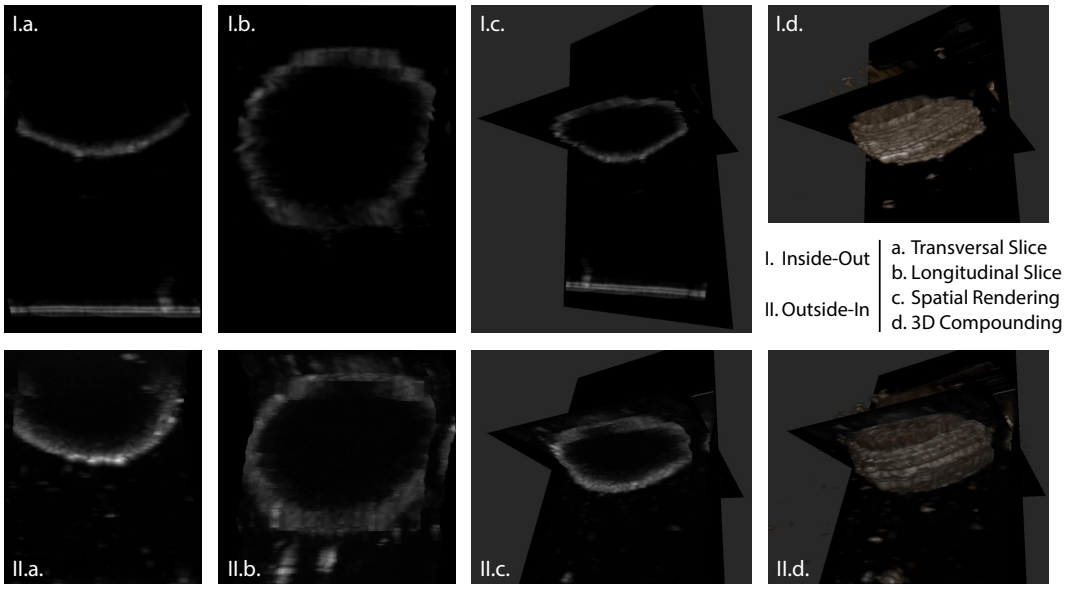


Fig. 8.9. **Qualitative tracking comparison for 3D ultrasound.** A direct comparison of tracking-based 3D ultrasound scans is shown for a scan of a phantom with a spherical probe inside. The top row shows the result of our inside-out tracking compared to the baseline outside-in scan shown in the bottom row. The different images (from left to right) show computational results after compounding for a transversal slice (a), a longitudinal slice (b), a two-plane spatial rendering (c) and the final 3D compounding. The reconstruction in both calculations recovers the spherical nature of the phantom while the rotational accuracy advantage of the inside-out tracker results in a more accurately defined boundary in the slices and a smoother rendering surface structure in the final compounding.

Aside of the improved compounding quality, **another advantage of the system lies in its practical use**. Without the need for spherical markers that are required to be visible from the external tracking system, the process of installing and adjusting the tracking cameras becomes obsolete for any use in a medical procedure. Line-of-sight problems arising from rotational motion of the ultrasound transducer are no longer problematic and even scans with complete rotations are feasible with the inside-out system without tracking loss which is not possible with an outside-in tracker that needs visibility of the full rigid body marker during the entire scan. Such procedures no longer require readjustment of the tracker or an additional reference target if repositioning of the cameras is required. The improved rotational accuracy for such procedures is further improved by the elimination of error propagation that are part of common outside-in setups when repositioning is required as an additional rigid world reference frame

[24] The video can be found under `http://campar.in.tum.de/Chair/PublicationDetail?pub=busam2018_pocus`.

in the form of a marker is typically co-calibrated to the system for instance during a 3D TRUS procedure. Such calibrations between different markers which are necessary for a consistent transformation chain into a common coordinate reference system introduce inaccuracies which propagate through all measurements and ultimately hamper the pose quality.

Overall, the markerless inside-out tracking method based on visual SLAM has demonstrated its accuracy advantage for general tracking as well as 3D ultrasound imaging. In the end, this can benefit medical procedures such as 3D transrectal prostate fusion biopsy and other procedures that primarily include rotational motion of the ultrasound probe. Reasoned by the accuracy and versatility, we believe that this can pave the way to more detailed investigation of the proposed markerless inside-out tracking method for the use in various medical procedures also by other research groups.

8.1.5. Mobile Tracking Systems

Through the experiments, we show the advantages in term of rotational accuracy and its benefits for the clinical use as a freehand 3D ultrasound tool are apparent. The fact that the tracking system itself is capable to use its surrounding for orientation even in unknown environments adds an orthogonal dimension to possible exploitation of the method. While the inside-out tracker is shown to work in the operating room (OR), it is not bound to it. The map that is incrementally built from the SLAM system can be initialized anywhere and does not require additional calibration. With the miniature dimension of the sensor, a **new mobility is given to medical tracking** and we are no longer bound to the use in just indoor environments, but can also imagine the system to run outdoors where it can be used for instance by emergency physicians as illustrated in Fig. 8.10.

Improvements of the internal image matching pipeline and speed ups using spatio-temporal video cues as investigated in the work of Ruhkamp et al. [358] can further enhance the ease of use of the tracking system and accelerate its acceptance in the medical domain by reducing the computational processing load.
With its limited spatial needs, the miniature sensor can further also be put on several devices that can cooperate with each other simultaneously for sensor fusion and collaborative interaction scenarios where the common reference frame is the computed global map itself. Investigations towards multiple systems with a common pose optimization or incremental map updates on the edge may also benefit medical purposes beyond freehand 3D ultrasound scans.

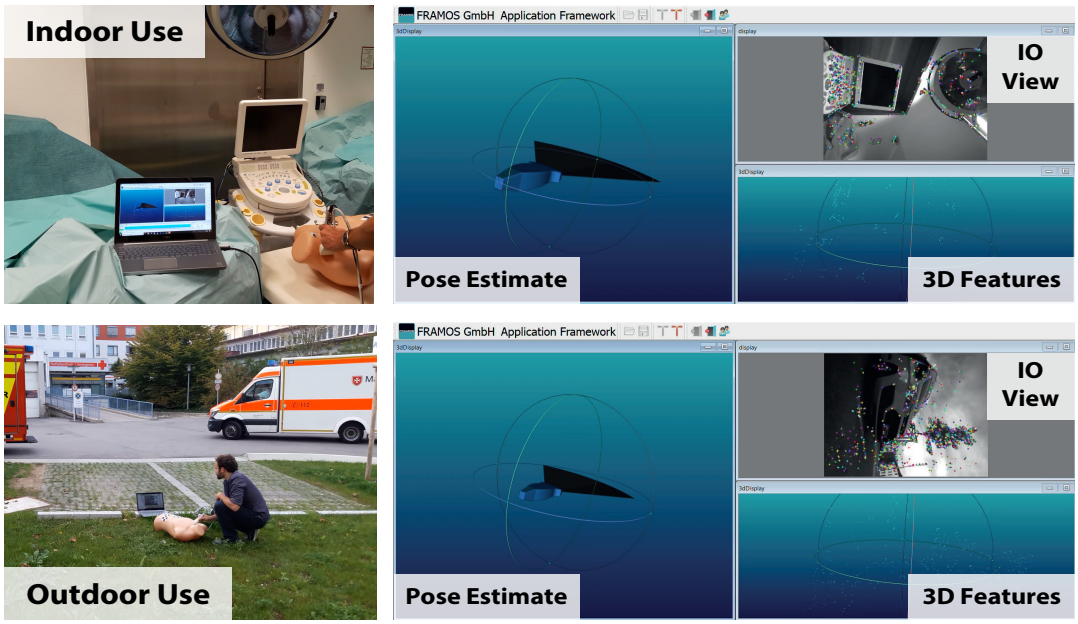


Fig. 8.10. **Mobile use of medical inside-out tracking.** The inside-out tracker for ultrasound pose tracking can be used in various environments. The top row shows an indoor use in an operating theatre where the features are found on medical equipment and within the room. The pose is illustrated here as a rendering of the ultrasound probe (centre column) and the feature points found in the sequences are visualized in 3D (right). The lower row illustrates the use of the portable tracker outside in a setup where it is run solely with a mobile laptop. Features in the inside-out view are found on surrounding objects which allow to built a 3D map.

8.2. Markerless Object Poses

The human brain is very efficient when it comes to rough estimates of position and orientation of objects in the field of view of a person without the need of specific markers or additional sensing other than pure sight. The degree of accuracy to which this is possible is in many cases sufficient for observations of fast motions, for complex interactions and even for manipulations of the visible object. This motivates us to mimic a similar approach computationally.
The aforementioned inside-out approach (see section 8.1) offers highly accurate pose estimates under more flexible conditions compared to marker-based outside-in trackers through improved mobility of miniature cameras. However, it suffers from the need of fixing the sensor to the object. Such a preparation is not always possible in particular if the object is not known before or many instances exist for example in a manufacturing process where multiple parts of the same kind may be used frequently by a robotic manipulator. Thus, we focus on building a **robust vision-based object pose estimation** pipeline that can be used for object pose estimation **in outside-in scenarios** without the need of markers or additionally mounted hardware.
For the purpose of visual pose estimation, prior object information of different form can be vital: May it be the knowledge of an observer about a similar part, a specific shape characteristic or a visually apparent feature. We want to exploit such cues in the consecutive discussions and start with geometric similarities and the observation that hand-made objects often share parametrizable 3D structures in section 8.2.1. We continue with an analysis of visual possibilities to create 3D object models in section 8.2.2 that can be used for comparison and learning purposes as detailed in section 8.2.3. In a final step, we propose an automated decision process to progressively estimate 6D object poses in section 8.2.4.

8.2.1. Geometric Parametrization

Primitive parametric shapes are oftentimes the building blocks of industrially manufactured goods. As a result, it is in many cases possible to describe the geometric structure of everyday objects as a **composition** of these simple forms. Poses can be estimated by surface reconstruction and parameter fitting. This allows for a description with adequate shapes that deform under parameter changes. While object geometry is frequently a result of computer aided design (CAD) whose models can be used for the purpose of pose estimation from 3D sensor data[25], also biological structures follow such forms approximately. In contrast to the scenario where the best 6D pose parameters are estimated to fit a known shape onto some measurement, **geometric primitives** function as a basis to gradually setup a 3D model in its own pose more generically.
Besides low-parametric 1D and 2D primitives such as points, lines and ellipses, many simple shapes in 3D space such as planes, spheres, cylinders can be described as 3D quadrics. These forms have attracted many researchers in the past[26] and are still a branch of active investigations.[27] The difficulty lies in the task of solving a combined detection and fitting problem where the goal is to detect the primitive in a cluttered 3D scene and fit its parameters to best match measurements from a 3D sensor even under occlusions and partial visibility.

[25] Cf. Birdal and Ilic [29].
[26] Cf. Miller [285], Cross and Zisserman [75], and Andrews and Séquin [6].
[27] Cf. Birdal et al. [26].

We consecutively give a brief overview over the related work in the area of compositionality and fitting with parametric shapes.

8.2.1.1. Literature Overview

The literature of instance-agnostic pose retrieval methods can be separated into parametric fitting methods, primitive detectors and their shape decomposition counterparts, and more recently also class-level shape and pose estimators.

Parametric fitting methods use a given parametric model and approximate its parameters to best match the observation. Prominent generic examples are second order surfaces such as quadrics[28] and superquadrics.[29] For 3D data, their characteristics help to estimate surface normals and curvature[30] and iterative methods for local surface fitting[31] can be used for mesh segmentation. The knowledge of quadric parameters can also be applied to practical vision tasks such as robotic grasping[32] and feature extraction from face images.[33] Fitting algorithms for quadrics are the focus of different investigations since the 1990s. The pioneering work of Taubin [407] approximates the geometric distance necessary for the parameter optimization with a Taylor approximation in order to efficiently fit a quadric. Through implicit use of local surface information, the method could be improved by Blane et al. [30]. Tasdizen [404] robustified the approach with a regularizer based on surface normals. A probabilistic approach using a Bayesian prior is chosen by Beale et al. [17]. All these methods, however, require nine or more points for the parameter fit. More recently, in the works of Birdal et al. [26] and Birdal et al. [27], we propose a solution that only requires four oriented points using tangential surface information as an additional constraint rather than a regularizer. With the novel construction, it is further possible to develop a voting strategy for parameter estimation that only requires three such points while being robust to higher levels of sensor noise. This approach also enables to detect the quadric primitive in a cluttered point cloud which was traditionally considered a separate task.

Shapes are often considered to be built out of atomic primitives.[34] While **shape decomposition** methods abstract complex 3D shapes into simple volumetric primitives such as cuboids[35] or learn to decompose into atomic superquadric elements[36], **primitive detection** approaches follow the other direction to find the primitves in sensor data.
Aside of cuboids and general (super-)quadrics, such primitive units can be planes, spheres, cylinders, cones, and many more. Planes can be robustly detected via Hough voting.[37] However, for more complex shapes, RANSAC-based ideas such as Globfit[38] were used early on. Plane detection can be improved with region growing and regularization enhances robustness.[39] Re-

[28] Cf. Cross and Zisserman [75].
[29] Cf. Leonardis, Jaklic, and Solina [243].
[30] Cf. Zhao et al. [483].
[31] Cf. Yan, Liu, and Wang [458].
[32] Cf. Uto et al. [431] as well as Pas and Platt [322].
[33] Cf. You and Zhang [465].
[34] Cf. Fidler, Boben, and Leonardis [113].
[35] Cf. Tulsiani et al. [424].
[36] Cf. Paschalidou, Ulusoy, and Geiger [323].
[37] Cf. Borrmann et al. [34].
[38] Cf. Li et al. [250].
[39] Cf. Oesau, Lafarge, and Alliez [314].

cent methods also investigate plane detection in large scenes and incremental measurements.[40] Cylinder detection is investigated by Qiu et al. [337] and the team of López-Rubio et al. [262] target ellipsoids with a non-linear optimization approach. Also hybrid methods that investigate paraboloid and hyperboloid[41] as well as cylinder and sphere[42] together are investigated. While formulations can be efficient, these pipelines are type-specific and consider objects of particular properties or deconstruct an unknown object into specific atomic units.

Pose estimation of such generic objects provides an interesting research direction. Information on the object of interest is, however, commonly available in practical scenarios before the acquisition. Oftentimes, even a prior video and offline calculations can be realized before object poses need to be estimated. From an application side, this is even more interesting as the addition of a depth camera or point cloud extraction hardware at test time may be more difficult or costly to add. In contrast to the classical geometric methods as mentioned before, we consequently want to focus on the case, where there is no explicit depth information present at test time and analyse a scenario with monocular video data only. To still be able to leverage prior information, we rely on 3D models of the object before we start our pose estimation routines. To understand the model extraction process with the help of vision sensors and to see how this has been addressed so far, we examine existing pose datasets and their 3D models.

8.2.2. 3D Models & Pose Datasets

To compare different tracking and detection methods for rigid pose estimation, various datasets have been proposed. Usually a **reference frame** is provided either **by markers** or hardware tools such as a turning table or a robotic manipulator. While marker boards are the most commonly used choice, this can also be done with more precise tracking systems as shown in Fig. 8.11 which illustrates a typical setup. The reference frame is typically used to register a **manually adjusted 6D pose** to later acquisitions while the object remains in the same position relative to this reference. Extrinsic co-calibration of different sensors enables the use of multiple modalities.

LineMOD[43] and its occlusion extension[44] are arguably the most widely used **datasets for 6D pose detection**. They provide **3D models** from 15 household objects acquired with a PrimeSense RGB-D Carmine sensor (PrimeSense, Israel) and their 3D meshes that are recovered with multi-view reconstruction. A common reference frame is extracted from a static marker board on which the objects are placed. It can be used to propagate manually annotated ground truth poses across a sequence.
More recently HomebrewedDB[45] uses three of LineMOD objects and adds 30 higher quality 3D models that are extracted with a commercial structured light 3D scanner (Artec Eva, Artec3D, Luxembourg). While the reference is also done with a planar marker board,[46] the scenes are

[40] Cf. Fang, Lafarge, and Desbrun [103] as well as Czerniawski et al. [77].
[41] Cf. Andrews et al. [6].
[42] Cf. Sveier et al. [401].
[43] Cf. Hinterstoisser et al. [174].
[44] Cf. Brachmann et al. [37].
[45] Cf. Kaskman et al. [203].
[46] Cf. Garrido-Jurado et al. [137].

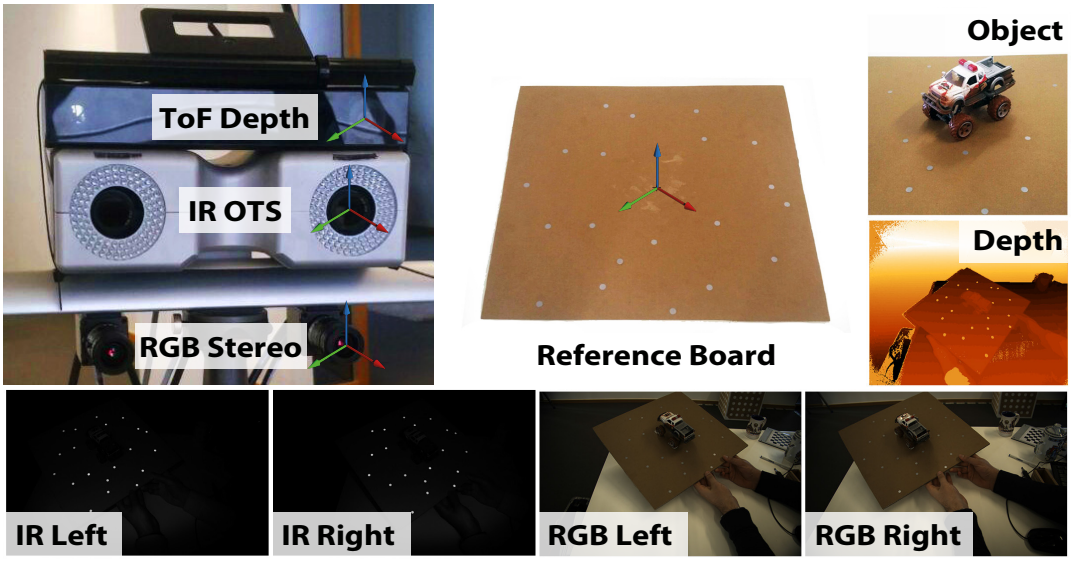


Fig. 8.11. **Hardware and acquisition setup for 6D pose dataset.** A typical setup to acquire a pose dataset is shown. Multiple sensors acquire the same scene: A depth camera (black sensor on top left) provides distance measures. In this case, a Kinect 2 (Microsoft, Redmond, US) uses indirect time-of-flight (ToF) sensing to acquire a depth map (colour coded on the right). The developed optical tracking system (OTS, see chapter 7.1) uses two infrared ring LEDs and a band-pass filter to acquire a stereo image pair in the infrared (IR) spectrum (bottom left). It provides the pose of the marker setup in the current frame. An additional binocular RGB stereo camera pair acquires RGB images (bottom right). All cameras are co-calibrated. The scene consists of a reference board (centre) which provides reference coordinates that are static with respect to diverse objects placed on top while the relative pose to the camera is changed. The toy truck is shown as an example object here in detail (top right). While the retro-reflective markers help to provide reliable and accurate pose measurements, they appear as artifacts with incorrect depth values in the ToF depth map.

more cluttered and acquired under different illumination conditions.

Other datasets focus on textured as well as texture-less objects in slight clutter[47] and under heavy occlusion.[48] A more industrial setup of a warehouse is used in the dataset of Rennie et al. [345] whose 3D models are extracted with photogrammetry and a monocular camera. The pose ground truth is annotated in a semi-manual process from a human annotator and with the help of depth data similar to the process done by Hodaň et al. [179] who propagate information with ICP alignment in their TUD Light and Toyota Light setups. The latter work also summarizes a series of datasets in the BOP 6D Pose Benchmark.

More industrial objects are proposed in the T-Less dataset[49] where the authors present 30 partly symmetric and texture-less objects together with manually created 3D CAD models and 3D scans. Their local reference frame is defined with the help of markers from the ARToolKitPlus.[50] Manually created 3D models are also used for the industrial objects in the MVTec ITODD dataset from Drost et al. [92] such that errors from model reconstruction are fully eliminated at the cost of realism. They also leverage a semi-manual annotation approach with ICP refinement on their high quality depth data extracted for objects on a turning table that provides a reference system.

[47] Cf. Tejani et al. [408].
[48] Cf. Doumanoglou et al. [91].
[49] Cf. Hodaň et al. [178].
[50] Cf. Wagner [439].

While these setups show large variability and the ground truth labels are often of high quality, the objects are all acquired from individual camera viewpoints that are not temporally connected through a video sequence.

The team of PoseCNN,[51] however, presents a dataset that includes 92 video sequences including 21 household objects from the YCB set[52] ready to evaluate also **6D pose tracking** methods. The YCB models are acquired with five RGB-D sensors and five high resolution RGB cameras that are placed in optimal positions around a turntable.[53] This allows for very accurate and high resolution **3D model** that are manually **aligned to the initial video frame**. The authors of PoseCNN do not use a marker board as a reference and propagate the initial pose automatically through the short video sequences with the help of a depth sensor and some fitting pipeline. Not always does this result in visually accurate ground truth positions and orientations as we discuss in section 8.2.4. Pixel-perfect ground truth poses and depth measurements without markers are given for the YCB objects in the synthetic data from Tremblay et al. [418] and with many additional occlusion cases in the extension by Jalal et al. [193] who use an elaborate rendering pipeline for high quality images and depth maps. The **domain gap between synthetic and real data**, however, provides additional challenges for 6D pose estimation pipelines[54] and even clever domain adaptation and randomization techniques[55] fall short behind training on real sensor data.

The real video RGB-D data, on the other side, provides a possibility to leverage additional spatio-temporal constraints, but suffers from ground truth accuracy drawbacks and error propagation of automatic annotation methods. Reference marker boards prevent the images from having natural surroundings and free interactions with a human. Moreover, they need to be excluded in order to not be recognized during training. A way to circumvent the shortcomings of real acquisitions without the need for a marker board is to not provide a stable reference frame on a board, but to use high quality marker-based pose annotations directly on the objects.

The team of Garon et al. [136] proposes to use miniature retro-reflective spherical markers with the **motion capture system** Vicon MX-T40 (Vicon, Oxford, UK). This allows them to improve their previous video dataset[56] which is using flat markers on a board as reference and ICP for propagating pose annotations in temporal sequences. Additional high quality 3D models are acquired with a handheld 3D scanner of 1 mm voxel resolution (Creaform GoScan, Creaform Inc., Lévis, Canada), and a manual cleaning in a post-processing step provides high resolution models with no visible artifacts. The poses are acquired with eight cameras and 3 mm markers on the objects serve as reference frame to which the models are registered. The footprint of the markers (see also Fig. 8.11) is removed from the depth images in an elaborate process to not provide artificial signals that can be picked up by learning based methods trained on this dataset. Aside of plain views, the so acquired RGB-D videos include severe occlusions and clutter.

[51] Cf. Xiang et al. [454].
[52] Cf. Calli et al. [56].
[53] The acquisition hardware setup is described by Singh et al. [380].
[54] Cf. Zakharov et al. [470].
[55] Cf. Zakharov, Kehl, and Ilic [469].
[56] Cf. Garon and Lalonde [135].

8.2.3. Visual Pose Estimation

With 3D object models and datasets to test pose estimation pipelines, one can evaluate the versatile approaches presented in this domain. The capability of interacting with objects in our 3D world and to understand the surrounding geometry correctly only from pixel data is a crucial element for successful vision systems. At their core lies relative position and orientation estimation. As this process is immanent for every real 3D camera application, many different solutions have been proposed to estimate rigid object and camera poses. Before we suggest a novel solution to monocular 6D pose estimation, we give a detailed overview of the field and discuss relevant challenges and put open questions in context.

8.2.3.1. From Markers to Features

Early works as discussed in section 7.2 directly apply **marker-based systems** to track objects. Typical augmented reality applications are driven by markers such as AR-Tag[57], ArUcO[58], AR-Toolkit[59] or AprilTag[60]. Similar to our proposed OTS from chapter 7, they can provide reliable and accurate tracking performance which makes these systems an attractive choice to calculate world anchors in dataset acquisition pipelines for marker-free methods (see section 8.2.2). Such systems are also used for sensor fusion as for example by Esposito et al. [101] and they can be extended to high accuracy systems.[61] Reliable and robust detection is of particular interest in the medical domain,[62] where our self-adhesive markers allow flexible usage.[63]

Object-marker calibration can be intricate and time-consuming in practice and **feature extractors** are a practicable alternative. In an ideal scenario, markers can then be fully replaced by automatically extracted keypoint structures encoded by natural object features in images. The goal of extraction pipelines is twofold, to detect points that describe significant and salient structures within the image on one hand while being invariant to image changes on the other hand. Such changes can be of geometric nature such as image translation, rotation as well as scale and viewpoint changes of the camera, but also influenced by exterior factors such as illumination changes, sensor noise or caused by seasonal variations of the surrounding. An **ideal feature** is a local structure that is highly distinctive and repeatable in order to be accurately and reliably detected under many image perturbations to enable a robust retrieval without confusion. Moreover, the quantity of features in an image is adequate to guarantee such properties and it can be described in a compact fashion that is efficient to compute.

[57] Cf. Fiala [111].
[58] Cf. Garrido-Jurado et al. [137].
[59] Cf. Kato and Billinghurst [204].
[60] Cf. Olson [316].
[61] Cf. Birdal, Dobryden, and Ilic [28].
[62] Cf. Esposito et al. [102].
[63] Cf. Busam et al. [49]

8.2.3.2. Feature Extraction Pipeline

The feature extraction pipeline commonly consists of a **detection** stage to find keypoints to which a consecutive **description** stage assigns a feature value. The same keypoints in different images are then matched in a **matching** stage. A saliency function typically assigns a reliability score to each pixel to highlight significant structures such a corners or blobs. Its local maxima define the keypoint locations after non-maximum-suppression. They can usually be characterized in terms of location, orientation, characteristic scale and their reliability score.

Early **detection methods** rely on gradient-based techniques to find corner structures.[64] Template-based methods such as SUSAN[65], FAST(ER)[66], and AGAST[67] improve upon the early corner detectors by utilizing machine learning and binary classifiers for efficient detection. Scale-space[68] analysis paved the way to blob detectors that calculate extrema in image pyramids with differential operators and scale-normalized extensions.[69] Such operators are efficiently approximated in pipelines such as SIFT[70] and SURF.[71]
Plenty of different detection methods have been suggested. A categorization of different methods is done in the survey of Li et al. [249] while Schmid et al. [369] evaluate different interest point detectors.[72] TILDE[73] tackles the problem of illumination changes through learning on real data while patch-based CNNs are used within the encoder-decoder network MagicPoint[74] which is fully trained on synthetic noisy primitives. The recent Key.Net[75] combines the strengths of both hand-crafted and learning pipelines in a unified hybrid detector.

Prominent **descriptors** include the descriptor part of the feature extraction pipeline from SIFT and binary descriptors such as BRIEF[76], BRISK[77], ORB[78], and FREAK[79] that can be matched efficiently using Hamming distances. The efficiency of these methods make them a common backbone for camera pose estimation[80] where they are used to match against a precomputed feature database.[81] Tracking applications such as the one described in section 8.1 benefit from the rotation accuracy of such systems in inside-out camera setups.
Deep learning based descriptors can be trained as soft binary classifiers and show advantages over hand-crafted pipelines.[82] Metric learning with triplet margin loss and hard negative min-

[64] Cf. Moravec [290] as well as Harris and Stephens [162].
[65] Cf. Smith and Brady [383].
[66] Cf. Rosten and Drummond [354] for FAST and Rosten, Porter, and Drummond [355] for FASTER.
[67] Cf. Mair et al. [271].
[68] Cf. Lindeberg [257].
[69] Cf. Lindeberg [258].
[70] Cf. Lowe [264].
[71] Cf. Bay, Tuytelaars, and Van Gool [16].
[72] Further evaluations are done also in the more recent works by Tuytelaars and Mikolajczyk [425] as well as in the works of Mikolajczyk et al. [283], Miksik and Mikolajczyk [284], Lee and Park [242], Salahat and Qasaimeh [361].
[73] Cf. Verdie et al. [434].
[74] Cf. DeTone, Malisiewicz, and Rabinovich [85].
[75] Cf. Barroso-Laguna et al. [13].
[76] Cf. Calonder et al. [57].
[77] Cf. Leutenegger, Chli, and Siegwart [246].
[78] Cf. Rublee et al. [356].
[79] Cf. Alahi, Ortiz, and Vandergheynst [5].
[80] Cf. Mur-Artal, Montiel, and Tardós [297], Mur-Artal and Tardós [298] as well as Campos et al. [58].
[81] Cf. Wu et al. [453] as well as Li, Snavely, and Huttenlocher [252].
[82] One of the first methods in this direction was L2-net that was presented by Tian, Fan, and Wu [410].

ing[83] is a common training strategy. Recent pipelines such as R2D2[84] train dense descriptors to estimate additionally both repeatability and reliability of the features.
The rise of modern RGB-D sensors also triggered the design of 3D descriptors[85] aside of image-only methods. While these methods can help for accurate object retrieval even in cluttered scenes[86] and recent data-based approaches also work on large point sets,[87] we focus here on 2D methods.

Joint **descriptor-detector pipelines** can also be learnt[88] and inclusion of structure from motion and depth sensing allows for unsupervised training[89]. SuperPoint[90] extends MagicPoint to joint descriptor-detector estimation and self-supervised learning through homography warping of input images. More robustness can be won by using a single feature map for both description and detection[91] and separate interaction between both stages can help during training.[92]

In the **matching stage**, the task is to compare the feature descriptors and find corresponding keypoints in different images or from model renderings in a pose estimation scenario. While k nearest neighbours in descriptor space can be found with a brute force searching strategy or via radius-search within a hypersphere of a specific radius, more efficient methods usually involve Kd-trees. There are various approximate nearest neighbour search strategies[93] many of which are integrated in the Flann library.[94]
In order to robustly detect the matches in the presence of outliers, simple tests exists. David Lowe proposed in the SIFT pipeline a simple ratio test to withdraw ambiguous matches if the score of the second best match is above a certain threshold. Cross check validation where the matches need to be mutually agreeing by matching from image A to B and vice versa can be an additional outlier removal strategy. More sophisticated robust matchers make use of grid-based motion statistics (GMS)[95] to enable high precision matches with low recall. DynaMiTe[96] deploys a dynamic model with temporal constraints to efficiently extend GMS in time. The advent of graph convolutional neural networks (GNNs) enables also learnt matching approaches. SuperGlue[97] for instance solves an optimal transport problem and leverages a GNN that can be directly connected to learning based detector and descriptor stages to solve a matching problem end-to-end in the presence of heavy noise and image variations.

Once the matches are found between object and observation, the pose can be calculated. The Perspective-n-Point (PnP) algorithm or one of its efficient variants (e.g. P3P or EPnP)[98] can be used to recover the **6D pose from** multiple **2D-3D correspondences**. RANSAC[99] helps to

[83] Cf. Mishchuk et al. [288].
[84] Cf. Revaud et al. [346].
[85] Cf. Rusu et al. [360] as well as Tombari, Salti, and Di Stefano [414].
[86] Cf. Mian, Bennamoun, and Owens [281].
[87] Cf. Saleh et al. [362].
[88] Cf. Yi et al. [463].
[89] Cf. Ono et al. [317].
[90] Cf. DeTone, Malisiewicz, and Rabinovich [86].
[91] Cf. Dusmanu et al. [93].
[92] Cf. Barroso-Laguna et al. [14].
[93] Cf. Fukunaga and Narendra [126], Beis and Lowe [18] as well as Silpa-Anan and Hartley [379].
[94] Cf. Muja and Lowe [292].
[95] Cf. Bian et al. [24].
[96] Cf. Ruhkamp et al. [358].
[97] Cf. Sarlin et al. [363].
[98] Cf. Ke and Roumeliotis [209] for P3P and for n points cf. Hesch and Roumeliotis [172] as well as Lepetit, Moreno-Noguer, and Fua [245].
[99] Cf. Fischler and Bolles [115].

estimate the pose in the presence of noise by random subset sampling of putative matches. The estimated pose is supported by an increasing subset of matches within an iterative process that aims to find an inlier consensus set. Robust metrics[100] such as the Huber loss or the Tukey biweight function also improve convergence and robustness in learning setups and can help to solve the pose estimation problem when formulated in an optimization framework even when the calculated features are ambiguous and lack descriptiveness. There exist various strategies to train a network within such an optimization framework in order to estimate the pose directly from input images. We briefly categorize them.

8.2.3.3. Pose Classification & Regression

Rotations densely populate a non-Euclidean space and there are multiple parametrization for the Riemannian manifold described by them as we have seen in section 6.2.[101] On the unit quaternion hypersphere for instance, the geodesic distance is not compliant with the Euclidean L-p norm in its 4D-embedding and the parametrization constitutes a double cover of the rotation group SO(3) impeding 6D pose regression networks.[102] Some works therefore discretize the problem and learn a **classifier**. The work of Kehl et al. [210] for instance discretizes the space in angular intervals and treat the rotation estimation as a classification problem. Template matching strategies are used by Hinterstoisser et al. [174] for viewpoint estimation and improvements[103] achieve a sub-linear matching complexity in the number of objects by hashing.

Different learning based methods are used to train a **regressor** for pose estimation. The scholars Brachmann et al. [37] as well as Tejani et al. [408] use random forests. CNNs for RGB-D based pose estimation are applied by both Kehl et al. [211] as well as Wang et al. [441]. While these methods need additional depth information, some works report results solely using RGB images[104] without the need for additional depth. To realize a 6D pose estimation pipeline, these methods are usually separated into three stages similar to the earlier feature extraction pipelines[105]: 2D detection, 2D keypoint extraction, 6D pose estimation. The work of Tekin et al. [409] is based on YOLO[106] and thus provides a single shot method. After bounding box corner or keypoint detection, the 6D pose is generally estimated with P*n*P as mentioned above.

Additional information in the form of semantic segmentation priors is used by Hu et al. [187] as well as Xiang et al. [454] while Do et al. [89] extend Mask-RCNN[107] with an extra branch for pose estimation.

[100] Cf. Barron [11].
[101] Cf. also Busam et al. [50].
[102] Cf. Zhou et al. [487].
[103] Cf. Cai, Werner, and Matas [55], Kehl et al. [213] as well as Hodaň et al. [180].
[104] Cf. Crivellaro et al. [74], Kehl et al. [210], Rad and Lepetit [339], Xiang et al. [454], Sundermeyer et al. [400].
[105] Cf. Kehl et al. [210], Sundermeyer et al. [400], Rad and Lepetit [339].
[106] Cf. Redmon et al. [344].
[107] Cf. He et al. [166].

8.2.3.4. Robust Pose Refinement & Limitations

For **robust pose detection** from pixel-wise object correspondences, P*n*P can be used within a RANSAC loop. This is combined with dense correspondences from Zakharov et al. [471] while Peng et al. [326] design a pixel-wise voting network to vote for specific keypoints and Li et al. [254] treat translation and rotation estimation separately. Further works specifically target the problem of 6D pose estimation under severe occlusion.[108] To improve upon the estimated pose, Li et al. [251] propose an RGB-based **refinement** strategy. Many methods, however, refine their RGB results with additional depth information using ICP.[109]

While these methods all improve the accuracy of pose estimators, the data is a bottleneck. Decently sized training sets with reliable pose annotations are difficult to produce (see section 8.2.2) while **synthetic renderings** seem a rather time-efficient solution. Thus, it is interesting to capitalise on synthetic data, however, with the given domain adaptation issues. The model performance is mostly hampered by the domain gap created through synthetic-only data training which is addressed with the help of depth maps by Rad et al. [340] whileRad et al. [341] applies a learnt feature transformation.

Independent of the amount of training data, additional issues arise by the nature of the problem. The projection operator that projects the 3D scene onto the 2D image plane is a surjection and not injective. Thus, the **projection is not invertible** and we loose information when observing an object as exemplified in Fig. 8.12. While the image content is enough to estimate the 6D pose of the object in some cases, a natural **pose ambiguity** arises in other cases from the missing information. To give an example: We can clearly define a rigid pose to a cup with a handle from a given image when the handle is visible (see Fig. 8.12). The moment, the handle becomes occluded by the cup itself and is not visible in the image anymore its pose becomes inherently ambiguous. If the handle is not visible from the camera perspective an infinite amount of 6D poses provide a solution to the estimation problem as they all are indistinguishable under projection. One way to deal with this is to replace a unique pose estimation with a probability estimation of possible pose hypotheses. We propose an estimation of a likely pose distribution given a single RGB image in our work of Manhardt et al. [272] where we explicitly deal with the presence of a single instantiation during training rather than the knowledge of the distribution itself. Solving the case through estimation of multiple likely hypothesis, it is possible to retrieve a visual ambiguity score to detect the presence of ambiguous cases. As a side product, we further retrieve a viewpoint dependent estimate for the axis of ambiguity. While this is a general limitation of the 6D pose estimation problem from a single camera view, we do not always observe pose ambiguities in practice as object texture and more complex geometry often resolve the issue.

8.2.3.5. Model-agnostic Poses

The main training strategy for 6D pose estimators is to **train one network per object**. Transferring pose knowledge to **unseen objects** even when they look similar is intricate and training

[108] Cf. Oberweger, Rad, and Lepetit [312] as well as Fu and Zhou [125].
[109] Cf. section 7.3.2.

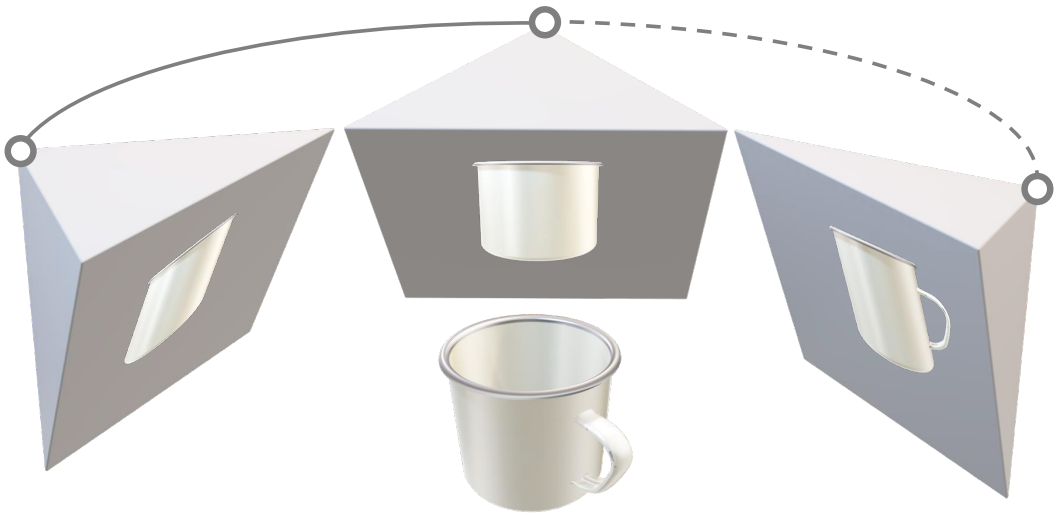


Fig. 8.12. **Ambiguous and unique object poses under projection.** The cup object (front) is here illustrated under three different perspectives that are shown with images acquired from different perspectives (left, centre, right). The frustums illustrate the camera position and the image plane. The left and centre image are indistinguishable without the handle of the cup in sight; so are all images taken from similar camera poses along the connecting grey line. The right most image defines a unique pose of the object which is possible due to the handle being in sight. The same is true for all similar camera poses along the dashed line for which the handle is visible in the images.

multiple objects together with the same network leads to resulting predictions that are unreliable.[110] Approaches to circumvent this downside involve the use of deformable shape models and annotations in the training dataset which can generalize to some extend across object instances.[111]

A new branch of works to bridge the gap between instance-based pose estimation and generic shape retrieval has just started recently. **Class-level 6D pose estimation** considers objects of a specific category, such as cups, cameras, laptops and so on. The task is to estimate the metric pose and shape of the object in sight in the absence of a 3D model. A non-parametric data-driven way leveraging recent advances in deep learning is thereby used to approximate both 6D pose and object geometry for a specific category. The first approach from Wang et al. [443] in this direction uses a monocular RGB image and a depth map to recover metric pose and shape. The CPS pipeline[112] goes one step further and uses only the RGB data for the same task.
It is correct that these methods are not agnostic to the object class. However, instead of heavily relying on the depth information or point cloud data as required by most of the more generic methods, this can pave the way to more general methods that do not require additional depth sensing or other prior shape information for the object of interest.

On another end, CorNet[113] focuses on the objects geometry instead and tries to detect model-agnostic corners. While this is more robust, it is in spirit similar to early pose estimation approaches that detect significant points. We follow a different path and learn a discrete

[110] Cf. Kaskman et al. [203].
[111] Cf. Pavlakos et al. [325].
[112] Cf. Manhardt et al. [274].
[113] Cf. Pitteri, Ilic, and Lepetit [330].

set of decisions that lead to the correct pose in section 8.2.4. This provides the flexibility of object-specific training as well as object-agnostic decisions trained with a heterogeneous dataset which we want to analyse specifically on video data where temporal information can be leveraged.

8.2.3.6. Temporal Tracking

Tracking of **3D object poses using temporal information** has been presented with the help of depth maps and point clouds. It can be realized with ICP and its variants.[114] These methods are very **sensitive to** their **initialization**.[115] They rely on an the initial pose which is required to be close to the correct prediction and often fail in the presence of heavy noise and clutter.[116] To stabilise tracking with depth maps, additional intensity information[117] or a robust learning procedure[118] can help. The current methods need one CNN trained per objects[119] or are bound to specific geometrical constraints such as planar objects.[120] PoseRBPF[121] is an efficient RGB-only tracker using a particle filter that helped to achieve state-of-the-art results on the YCB dataset.[122]

We want to consecutively focus on a novel, model-agnostic approach to monocular pose estimation that can benefit from temporal data. Inspired by classical temporal trackers whose optimization procedure usually include incremental pose updates, we formulate iterative updates as an action decision process. We will see that this formulation allows to reach a wide convergence basin independent of the object model. While we largely benefit from temporal information in terms of computation time, our method can also be used to detect the pose with multiple seeds intuitively.

8.2.4. Pose Estimation as Action Decision Process

Object pose estimation from monocular RGB images is an integral part of robot vision and augmented reality. Robust and accurate pose prediction of both object rotation and translation is a crucial element to enable precise and safe human-machine interactions and to allow visualization in mixed reality.
Previous 6D pose estimation methods treat the problem either as a regression task or discretize the pose space to classify. We reformulate the problem as an **action decision process** where an initial pose is updated in incremental discrete steps that sequentially **move a virtual 3D rendering towards the correct solution**. A neural network estimates likely moves from a single RGB image iteratively and determines so an acceptable final pose. In comparison to previous approaches that learn an object-specific pose embedding, a decision process allows

[114] Cf. Rusinkiewicz and Levoy [359] as well as Segal, Haehnel, and Thrun [372].
[115] Cf. Zhang [479].
[116] Cf. Garon, Laurendeau, and Lalonde [136].
[117] Cf. Held et al. [169], Yuheng Ren et al. [466], Joseph Tan et al. [200] as well as Kehl et al. [212].
[118] Cf. Tan and Ilic [403].
[119] Cf. Garon and Lalonde [135].
[120] Cf. Wang and Ling [445].
[121] Cf. Deng et al. [82].
[122] Cf. Xiang et al. [454].

for a lightweight architecture while it naturally **generalizes to unseen objects**. Moreover, the coherent action for process termination enables dynamic reduction of the computation cost if there are insignificant changes in a video sequence. While other methods only provide a static inference time, we can thereby automatically increase the runtime depending on the object motion. We fully train and test the lightweight network on a consumer laptop using only synthetic data with pixel-perfect annotations and evaluate robustness and accuracy of our action decision network on real video scenes with known and unknown objects and show how this can improve the state-of-the-art on YCB videos[123] significantly.[124]

8.2.4.1. Motivation

We live in a 3D world. Every object with which we interact has six degrees of freedom to move freely in space, three for its orientation and three for its translation. Thus, the question to determine these parameters naturally arises whenever we include a vision system observing the scene. A single camera will only observe a projection of this world. Thus, recovering such 3D information constitutes an inherently ill-posed problem which has drawn attention of many vision experts in the past.[125] The motives for this can be different: One may want to extract scene content for accurate measurements[126], camera localization[127] or 3D reconstruction.[128] Another driver can be geometric image manipulation[129] or sensor fusion.[130] Also human-robot interaction[131] and robot grasping[132] require estimation of 6D poses.

The rise of low-cost RGBD sensors helped development of 6D pose detectors[133] and trackers.[134] More recently, the field also considers methods with single RGB image input as discussed in the literature overview before. The best performing methods for this task[135] are all data-driven and thus require a certain amount of training images. Annotating a large body of data for this kind of task is cumbersome and time-intensive which yields to either complex acquisition setups or diverse annotation quality.[136] A variety of reconstruction methods allow to provide high quality 3D models of the objects with the datasets.[137] The majority of pose estimation pipelines such as the work of Tekin et al. [409], Xiang et al. [454] as well as Peng et al. [326] are all trained on real data. Besides **difficult and time-consuming pose annotations**, this brings two further drawbacks. On one hand, the networks adjust to the individual sensor noise of the acquisition hardware drastically hampering generalization capabilities.[138] On the other hand, every real annotation has its own errors introduced either by the used ground truth

[123] Cf. Xiang et al. [454].
[124] The accompanying paper can be found as a preprint from Busam, Jung, and Navab [52].
[125] Cf. Kato and Billinghurst [204], Lepetit, Moreno-Noguer, and Fua [245], Hinterstoisser et al. [174], Mur-Artal, Montiel, and Tardós [297], Xiang et al. [454].
[126] Cf. Birdal, Dobryden, and Ilic [28].
[127] Cf. Mur-Artal and Tardós [298].
[128] Cf. Knapitsch et al. [223].
[129] Cf. Holynski and Kopf [182] as well as Busam et al. [51].
[130] Cf. Esposito et al. [101].
[131] Cf. Busam et al. [49].
[132] Cf. Drost et al. [92].
[133] Cf. Brachmann et al. [37], Kehl et al. [211], Wang et al. [441].
[134] Cf. Tan and Ilic [403] as well as Garon, Laurendeau, and Lalonde [136].
[135] Cf. Peng et al. [326], Zakharov, Shugurov, and Ilic [471], Hodaň et al. [179].
[136] Cf. Garon, Laurendeau, and Lalonde [136].
[137] Cf. Xiang et al. [454], Garon, Laurendeau, and Lalonde [136], Tekin, Sinha, and Fua [409].
[138] Cf. Kaskman et al. [203].

sensor system or by the human annotator. These errors propagate to every model trained on it. Modern 3D renderers, however, can produce photorealistic images in high quantity with pixel-perfect ground truth. Some recent scholars therefore propose to leverage such data[139] and fully **train on synthetic images**. Most widely used evaluation datasets[140] provide single image acquisitions and more recently, video sequences[141] with pose annotations are available even though video data is the natural data source in applications.

8.2.4.2. Novelty & Contribution

We leverage the temporal component in video data to accelerate our pose estimation performance and propose an RGB pose estimation pipeline by taking inspiration from the reinforcement learning approach proposed for 2D bounding box tracking[142] where the authors frame the problem with consecutive discrete actions for an agent. In contrast to these methods, we propose a pose estimation pipeline with a large convergence basin that can run a single network with multiple seeds as a pose detection pipeline omitting the use of another model. We frame 6D pose estimation as an action decision process realized by applying a network that determines a sequence of likely object moves as shown in Fig. 8.13.

At first, an initial pose is used to **render the 3D model**. Both the rendering and the current image are cropped around the virtual pose and fed to a lightweight CNN. The **network predicts a pose action** to move the rendering closer to the real object. All 13 possible actions are illustrated in Fig. 8.14. The stepsize is hereby fixed and predefined. It determines the accuracy of the process and the convergence speed. In case the process continues, the pose is modified according to the action and the new rendering is fed back into the pipeline with a new crop to **move the estimation incrementally closer to the observation**. This goes on until either the stop criterion fires or the maximum number of iterations is reached.
If our input is a video stream, we can use the pose retrieved at frame $t-1$ as an initial pose for frame t which can greatly reduce the computation time as the amount of iterations is determined by the pose actions needed between the initial pose and the result. An example is shown in Fig. 8.15 where the rendering is moved until stop is predicted. This pose initializes the next frame.

[139] Cf. Kehl et al. [210], Sundermeyer et al. [400], Zakharov, Shugurov, and Ilic [471].
[140] Cf. Hinterstoisser et al. [174] as well as Brachmann et al. [37].
[141] Cf. Xiang et al. [454] as well as Garon, Laurendeau, and Lalonde [136].
[142] Cf. Yun et al. [467].

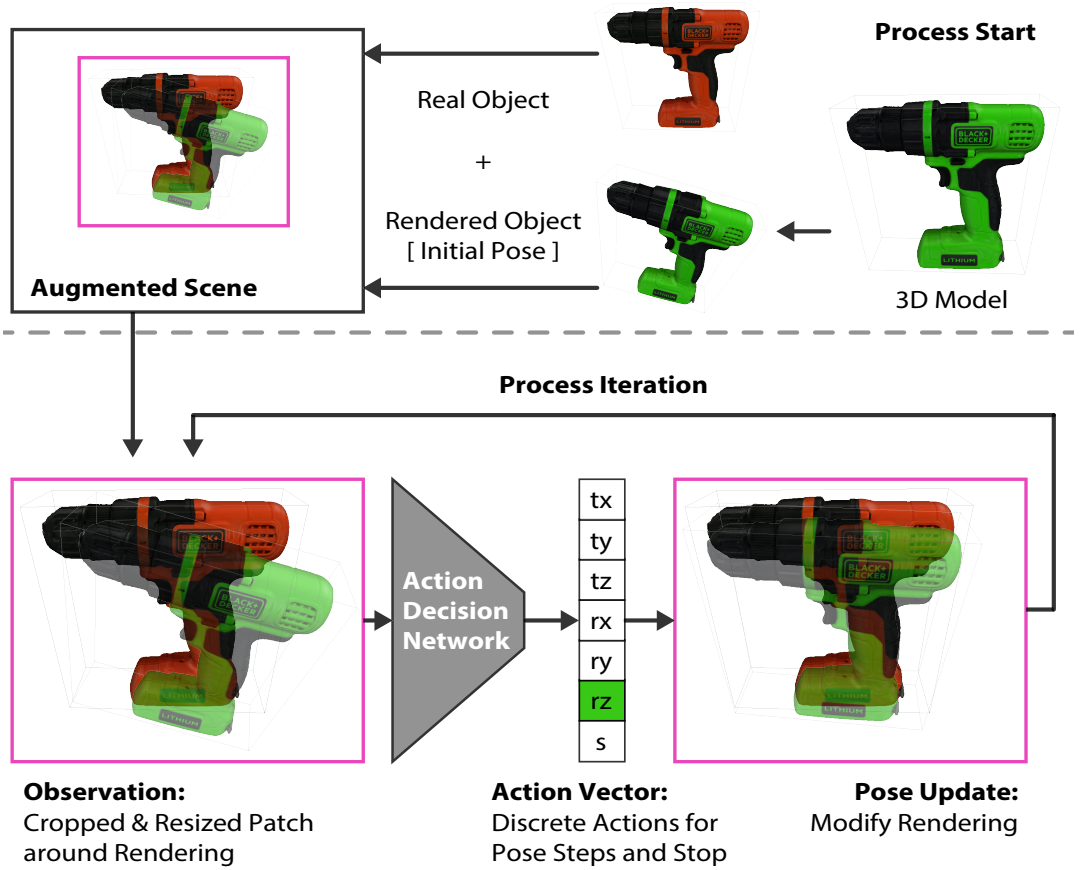


Fig. 8.13. **Action decision process for 6D pose estimation.** To estimate the pose of a real object, a virtual object is rendered with an initial pose (top from right). For illustration purposes we visualize the synthetic renderings of an example object in green and the real observations in original orange colour. Both image and rendering are cropped (RoI in pink, top left). A lightweight action decision network determines an incremental move to bring the rendering closer to the real observation. The updated pose is used to iteratively modify the rendering.

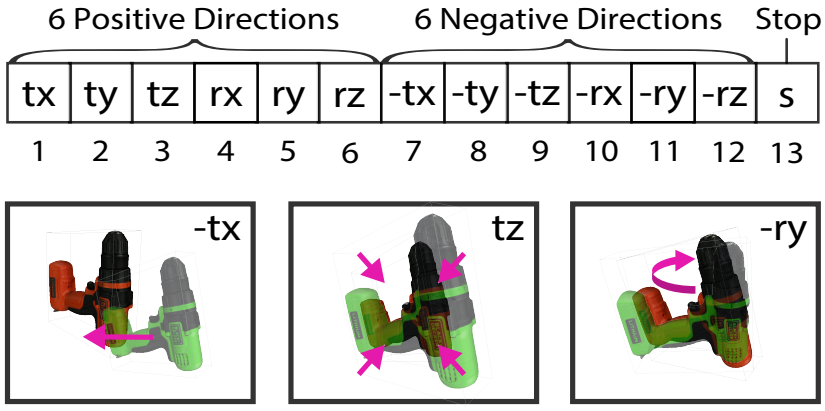


Fig. 8.14. **Pose actions for pose updates.** There are 13 possible actions (top): 6 pose actions to move in positive, 6 to move in negative direction, and one action for stop. In each process iteration loop, the next best action is predicted. On the bottom, we illustrate three example actions where the current rendering is shown in green, the observation in original orange colour and the effect of the predicted action as stated in the top right corner of the box is depicted with pink arrows.

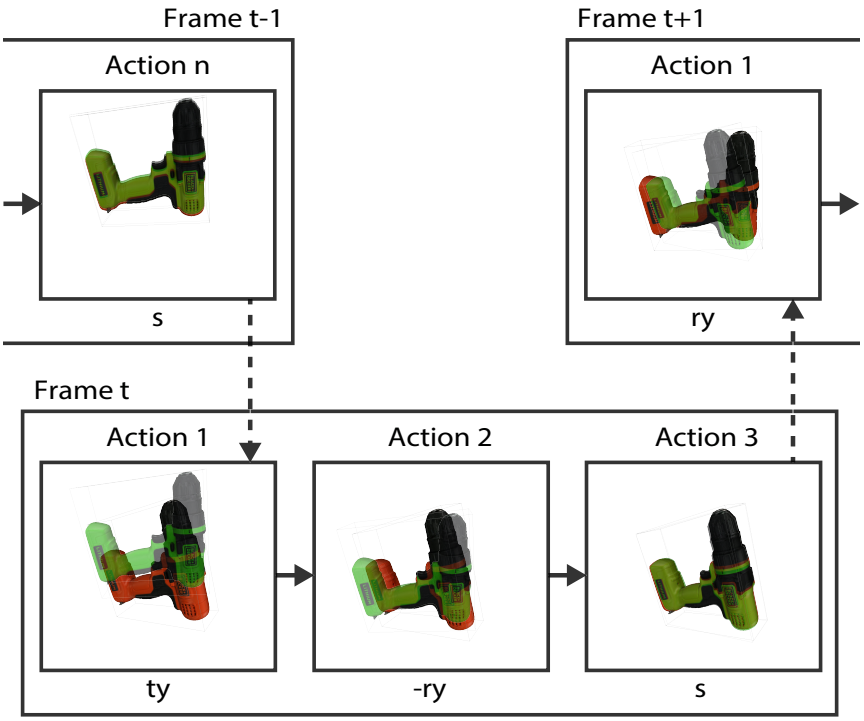


Fig. 8.15. **Action decision process for object tracking.** Consecutive frames in a video sequence have a similar pose if the framerate is high enough. The final pose from frame $t-1$ predicted after n actions (top left) initializes the pose rendering at frame t in a video sequence. Multiple actions (bottom) bring the rendering (green) closer to the observation (orange) until the stop action is detected (bottom right). The result is accepted as the estimation for frame t and used as an initial guess for frame $t+1$ (top right). The green colour is only chosen for illustration purposes.

Improving pose estimation with iterative inference has previously been explored by Li et al. [251] where a refinement network is iteratively applied to refine a pose predicted by an estimator such as PoseCNN.[143] However, the performance of their method actually decreases if more than two iterations are used while our pose estimation is gradually improved. In summary, our contribution is fourfold:

1. We reformulate 6D pose estimation as an **action decision process** and design a lightweight CNN architecture for this task that **generalizes to unseen objects**.
2. We **iteratively** apply our shallow network to optimize the pose and deploy a change-aware **dynamic complexity reduction** scheme to improve inference cost.
3. We provide an RGB-only method that is able to **improve** the state-of-the-art for **video pose estimation** while being able to track objects in presence of noise and clutter.
4. We provide a **data augmentation scheme** to render high-quality images of 3D models on real backgrounds under varying clutter and occlusion.

In the remainder of this section, we present our method and network architecture in detail before we report an extensive analysis and evaluation.

8.2.4.3. Learning Action Decisions

Our target is to optimize an action decision CNN to decide for iterative discrete actions to move a rendered 3D model to the observed position of the according object in an image sequence as shown in Fig. 8.13. An initial pose is used to crop the image with the projected bounding box of the object. We discretize the set of possible actions to move or not to move a 3D object depending on the six degrees of freedom for rigid displacement in space. The **13 possible actions** are divided into six pose actions for positive parameter adjustment, six for negative changes and an action to stop the process (i.e. not to move the object). For each of these actions, we set units depending on an image and a current crop: The movements for t_x, t_y are measured in pixels and determine movements of the bounding box. The parameters r_x, r_y, r_z are measured in degrees and t_z is determined as the diameter in pixels of the current bounding box. An action can change the position and size of the crop. The image crop is always rescaled to a quadratic $n \times n$ patch of the same size as the rendering.

We decide to implement the action decision CNN with a lightweight architecture that allows for training on a consumer laptop. The MoveIt architecture which realizes this is shown in detail in Fig. 8.16. An attention mechanism is implemented as guidance for the network to focus on relevant image regions and ignore occlusions. This attention map is learnt in an unsupervised way during training to mask the embedded feature tensor and to realize a weighted global average pooling. We train the model end-to-end with synthetic data where a random action vector is created, normalized and a softmax cross entropy loss between logits

[143] Cf. Xiang et al. [454].

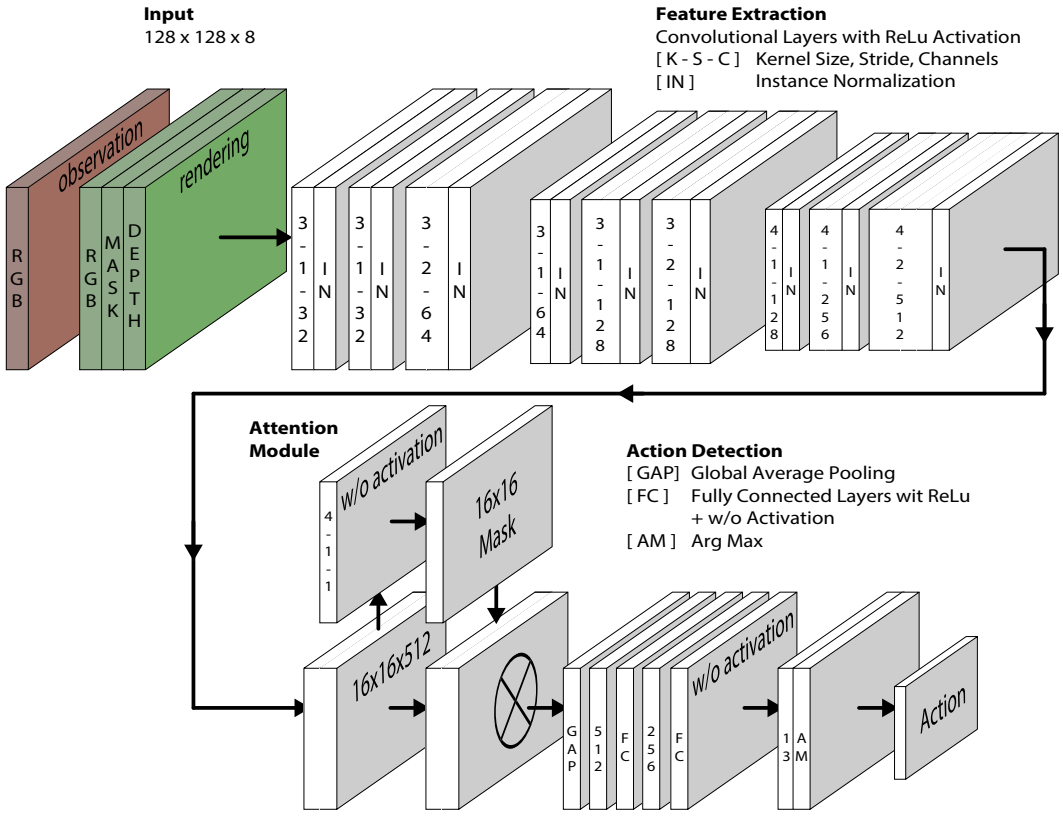


Fig. 8.16. **MoveIt architecture.** The input (top left) is the RGB video frame cropped (red) and concatenated with the rendered RGB, rendered depth and rendered segmentation mask (green). A series of convolutional layers with ReLu activations are used to extract an embedding (upper right block). An unsupervised attention mask (lower left) focuses the features in case of partial occlusions before a global average pooling layer. Two fully connected layers extract the set of action logits from which the most probable is selected with argmax (bottom right). Kernel size (K), stride (S), and channel amount (C) are indicated for the convolutional layers with K-S-C.

and labels is utilized to optimize the probability error in this mutually exclusive and discrete action classification task.

Usually the iteration process is stopped with the stop action in frame t and the last pose is used to initialize the process in frame $t+1$ as shown in Fig. 8.15. As we discretize the pose steps, the stop criterion, however, is not always met perfectly. Moreover, the decision boundary between the stop criterion and some close action may lead to oscillations between two or multiple predictions close to the correct result. To cope with this in practice, we can also stop the process early if we encounter oscillations and if an intermediate pose has been predicted already in the same loop or if a maximum number of iterations is reached.

8.2.4.4. Training on Synthetic Data

To train our model, we create a synthetic dataset generation pipeline where we **render** the **3D models** with changing backgrounds and varying poses in clutter and occlusion **on top of**

real images. Following Kehl et al. [210] we use images from the MS COCO dataset[144] as background. We randomly pick 40k images from it and use the high quality 3D models from YCB[145] and the models from Linemod[146] to render the objects during training in various poses on top of the images as shown in Fig. 8.17.

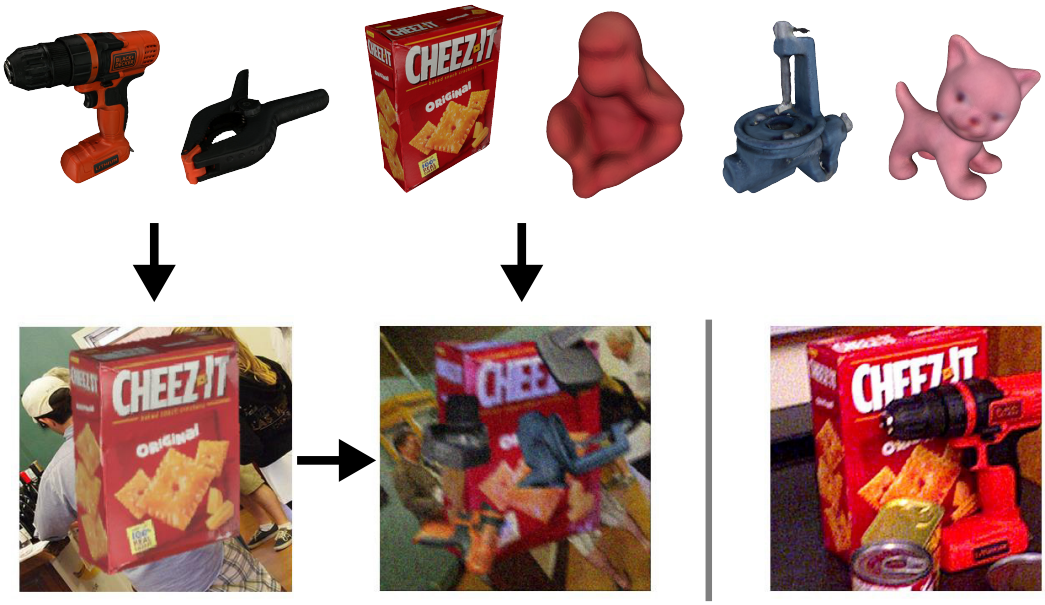


Fig. 8.17. **Synthetic dataset creation.** High quality 3D models from YCB (top left) and 3D models from Linemod (top right) are rendered in various poses on top of 2D images from MS COCO (bottom left). Augmentation in form of blur, light changes and occlusions is added (bottom centre). A comparison image from the real dataset is shown (bottom right).

Data Augmentation. We augment the renderings in different ways with occluders, crops, image blur as well as material and light changes before placing it on top of the COCO images. As our network operates on cropped images patches of size 128×128 pixels, we perform the augmentation on these patches, too. An augmentation example is shown in Fig. 8.17. We synthetically generate 50k images for each YCB object and 50k images for each Linemod model. To do so, we simulate two different kinds of **blur** to augment the data with TensorFlow. In 75% of the cases, we randomly add motion blur and in 25% of training scenarios a radial blur. Both are generated with a mean of $\mu = 0$ and $\sigma = 0.05$ standard deviation for all three colour channels. **Variety in the exposures** are augmented through changes of brightness, contrast and saturation values in the range of $[0.95, 1.25]$. For **object material and light** augmentation, we leverage the unity[147] engine and simulate 20% of unlit material and 80% of standard material (i.e. metallic in the range of $[0, 0.85]$ and glossiness/smoothness in $[0, 0.8]$). Light is augmented with five point lights at random positions with an intensity drawn from $[0.5, 1.5]$. We change the light colour randomly by picking one colour from $C = \{\text{blue, cyan, green, magenta, red, yellow, white}\}$ at every capture and set the same colour for all five lights. The colour brightness for the light is randomly enhanced offering subtle additional variation in contrast to the intensity changes. Then we do a **random crop** of the rendering patch with 128×128 pixels to a height and width within $[96, 128]$ and resize the

[144] Cf. Lin et al. [256].
[145] Cf. Xiang et al. [454].
[146] Cf. Hinterstoisser et al. [174].
[147] Cf. Haas [158].

patch to a value within $[32, 64]$. To simulate **occlusion**, we render 20k patches from YCB and Linemod models with random poses from which we pick four samples at each training step. Firstly, they all are processed by the aforementioned blur and colour augmentation scheme. In 50% of the cases, we do not occlude the patch. In the other cases we use these four samples for occlusion. With a 12.5% chance we respectively select either one, two or three occluders at random or use all four. Finally, we crop the entire masked region of the augmentation pipeline in 25% of the cases to simulate another occlusion scenario where we select the cropped region patch height and width randomly from $[72, 96]$. We apply this procedure to generate 50k images for each YCB object and 50k images for each Linemod model. We consider these images as our synthetic ground truth.

To simulate also the **initial pose seeds**, we produce a variety of 3D renderings without any augmentation a set of actions away from the related synthetic ground truth patch. We want our method to work particularly well close to the correct result where it is crucial to take the right decisions in order to converge. For this reason instead of rendering random seeds evenly distributed in pose space, we pay close attention near the ground truth by providing more training data in this region. We group the pose seeds in five clusters: 10k each for YCB and Linemod. The first cluster contains *small* misalignment in only one action direction, where each action has an equal chance of $1/13$ to be picked, also the stop-action. For the step size it holds $t_x, t_y \in [1, 5]$, $t_z, r_i \in [1, 4] \, \forall i$. The second group consists of *larger* misalignment in only one direction with equal chance. For this we chose $t_x, t_y \in [5, 30]$, $t_z \in [1, 15]$, $r_i \in [4, 20] \, \forall i$. The third group is *mixed* where we have one larger misalignment in one direction and the remaining actions are random small misalignment (e.g. $t_x = 10$ and all other directions are randomly chosen as in group one). The fourth and fifth groups are a *random small* and a *random large* mix of misalignments from groups one and two.

Training Modes & Attention. We train networks for each YCB model (**object-specific training**) and one network with mixed training including all YCB and Linemod models (**multi-object training**). Fig. 8.18 shows the unsupervised training of our attention map on the same image after different number of iterations for training with cracker box. It can be seen, that the attention mask first focuses on high gradient object regions (250 iterations) before the mask emphasizes on the overall object geometry excluding big occlusion patches (6k iterations). Finally it learns to exclude the finer occluder details such as the front part of the drill (15k iterations).

8.2.4.5. Experimental Evaluation

We implement the model using the 3D renderer from unity[148] with a customized version of the ML-agent toolkit[149] to seamlessly support our model, load training, provide visualizations for debugging purposes and run all our experiments. We combine it with TensorFlow (1.7.1 tensorflow-gpu) and use TensorFlow 1.10.0 for training to have the necessary functionality support. The batch size is set to 32 and we use the ADAM[150] optimizer with a learning rate of 10^{-4} and exponential decay of 5% every 1k iterations. We trained all our models until

[148] We use unity version 2018.3.6f1, Unity Technologies [429].
[149] Cf. Juliani et al. [201].
[150] Cf. Kingma and Ba [219].

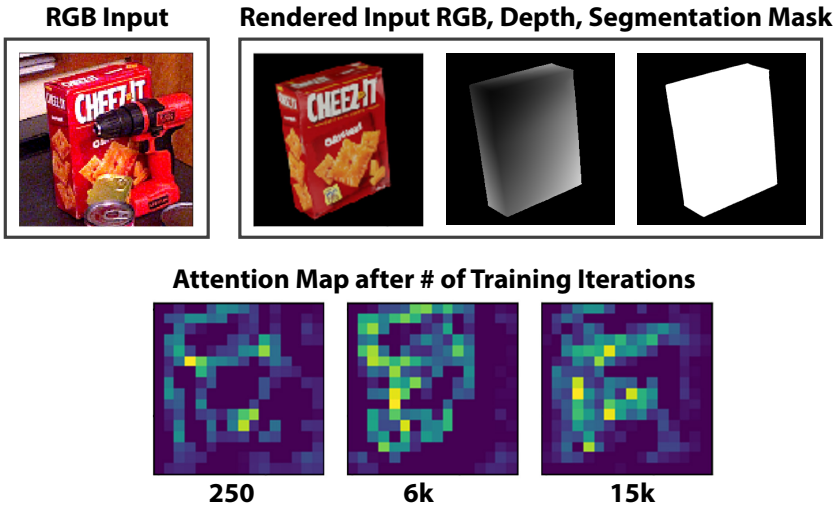


Fig. 8.18. **Unsupervised training of attention map.** The input RGB (top left) as well as the three input renderings (top right) are shown together with the results of the unsupervised training of the attention map after different numbers of training steps (bottom).

convergence (i.e. 25k iterations for object-specific training and 50k for multi-object training). All our **experiments as well as training** and dataset creation is done **on a consumer laptop** with an Intel Xeon E3-1505Mv6 CPU and an Nvdia Quadro P5000 mobile GPU.

Dataset Choice. High quality pose annotations are usually acquired with fiducial markers, manual annotation or a combination of both as discussed in section 8.2.2. This process is very time-consuming and thus video annotations for 6D pose estimation are not easily retrieved. In order to produce the marker-free video pose dataset **YCB**[151], the authors manually annotated only the poses of all the objects in the first frame of a sequence and refine them with an algorithm based on Signed Distance Functions. The ground truth labels for the rest of the frames within the sequences are retrieved by camera trajectory estimation with a depth-based tracker and the constraint for constant relative object poses within the scene. This eliminates possible fiducial marker cues that could eventually provide a signal to a learning-based method at the cost of not being able to freely move the objects. While this allows also for larger frame sets, the quality of the annotations can vary. The **Laval**[152] video dataset circumvents this issue through the use of a motion capture system and retro-reflective markers attached to the real objects in the scene. A post-processing step in their pipeline cures the depth images by removing strong artifacts that arise from marker reflections to provide cleaned depth images also for RGB-D methods. We test our models on these two datasets and evaluate both quantitatively and qualitatively. We note that the models in the YCB dataset are part of our training, while the objects from Laval are entirely *unseen*.

Quantitative & Qualitative Evaluation. For all quantitative experiments, we follow the protocol of Garon et al. [136] and reset the pose estimation with the annotated pose every 15 frames. The maximum number of action steps per frame is set to 30. At first, we test our networks trained on individual YCB models and compare with their ground truth poses. The

[151] Cf. Xiang et al. [454].
[152] Cf. Garon, Laurendeau, and Lalonde [136].

result is reported in comparison with the state-of-the-art[153] in Tab. 8.1 columns two to six. We utilize the 3D metrics for ADD and ADI (for symmetric objects) relative to the object diameter as proposed by Hinterstoisser et al. [174]. An extensive comparison with absolute thresholds is provided in the appendix A.2 and more qualitative examples can be found in a supplementary video online.[154]

We note an average improvement of 9.94% compared to Oberweger, Rad, and Lepetit [312] for our method and **investigate** the **failure cases**. While most of them seem visually plausible, we still observe a significant accuracy variance between the video sequences in YCB which we further analyze. It turns out that the **annotations for some of the objects** are slightly **shifted** as shown in Fig. 8.19. Our method – in contrast to others with which we compare in Tab. 8.1 – is fully trained on synthetic data. Thus, we cannot learn an annotation offset during training time due to the fact that our training setup provides pixel-perfect ground truth. Further investigations revealed that the ground truth annotation quality is a common issue amongst multiple video sequences in this dataset.
We believe that the main source for this is that an incorrect annotation in the first frame propagates constantly through the entire sequence, and the manual label was only given in frame one.[155] We **correct** this shift such that the **annotation** visually overlaps the RGB observation by one single, constant translation delta for each of the sequences and rerun the evaluation. The results are shown in the last column of Tab. 8.1, where also the accuracy of our method improves significantly to a margin of 28.64% over the state-of-the-art. The corrected annotations ease the comparison between synthetic and real data training on this dataset and help to improve future pipelines.

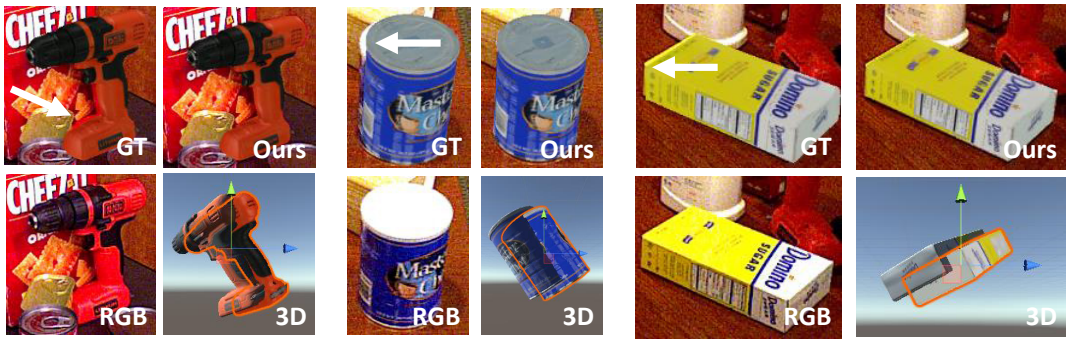


Fig. 8.19. **Annotation quality of YCB data.** The input image is shown together with our prediction and the ground truth annotation. Arrows and 3D visualization are added to detail the difference in these cases where our estimation is considered incorrect in comparison with the provided pose annotation, but visually aligns with the object.

The metric used for the evaluation (Tab. 8.1) is the standard ADD measure[156] relative to the object diameter where a pose estimate is considered successful if its ADD value is below 10% of the object diameter. The final ADD score is calculated by the percentage of frames with such a successful estimation. Tables A.1, A.2, A.3 in appendix A.2 additionally compare the area under the ADD threshold curve (AUC) for varying absolute thresholds from zero to 0.1 m.[157] The extensive study in comparison with the state-of-the-art shows that our method

[153] Cf. Xiang et al. [454], Fu and Zhou [125], Hu et al. [187], Oberweger, Rad, and Lepetit [312].
[154] The video can be found under `http://campar.in.tum.de/Chair/PublicationDetail?pub=busam2020_moveIt`.
[155] Cf. Xiang et al. [454].
[156] Cf. Hinterstoisser et al. [174].
[157] Cf. Xiang et al. [454].

compares favourable on the standard benchmark (Ours OS) and significantly better with the shift-correction.

Model	PC [454]	HMP [125]	SD [187]	HM [312]	Ours OS	Ours + Shift
002_master_chef_can	3.60	40.10	33.00	**75.80**	7.70	91.88
003_cracker_box	25.10	69.50	46.60	86.20	**88.36**	97.76
004_sugar_box	40.30	49.70	**75.60**	67.70	58.35	91.95
005_tomato_soup_can	25.50	36.10	**40.80**	38.10	38.23	57.99
006_mustard_bottle	61.90	57.90	70.60	**95.20**	87.74	98.49
007_tuna_fish_can	11.40	9.80	18.10	5.83	**47.90**	52.89
008_pudding_box	14.50	67.20	12.20	**82.20**	58.68	76.00
009_gelatin_box	12.10	59.10	59.40	**87.80**	37.08	89.20
010_potted_meat_can	18.90	42.00	33.30	**46.50**	45.99	60.61
011_banana	30.30	19.30	16.60	30.80	**74.02**	90.43
019_pitcher_base	15.60	58.50	90.00	57.90	**99.40**	100.00
021_bleach_cleanser	21.20	69.40	70.90	73.30	**95.04**	95.30
024_bowl	12.10	27.70	30.50	36.90	**99.44**	99.44
025_mug	5.20	12.90	40.70	17.50	**45.35**	76.59
035_power_drill	29.90	51.80	63.50	**78.80**	52.77	97.35
036_wood_block	10.70	35.70	27.70	33.90	**52.28**	63.48
037_scissors	2.20	2.10	17.10	43.10	**63.33**	81.11
040_large_marker	3.40	3.60	4.80	8.88	**39.53**	41.73
051_large_clamp	28.50	11.20	25.60	50.10	**64.01**	82.83
052_extra_large_clamp	19.60	30.90	8.80	32.50	**88.02**	91.37
061_foam_brick	54.50	55.40	34.70	66.30	**80.83**	80.83
Average	21.26	38.57	39.07	53.11	**63.05**	81.75

Tab. 8.1. **Evaluation on the YCB dataset with our object-specific models, AD{D|I}.** We compare the percentage of frames for which the 3D AD{D|I} error is < 10% of the object diameter as suggested by Hinterstoisser et al. [174]. Symmetric objects are shown in italic letters.

Generalization and Ablation. Given these problematic initial annotations, we refrain from further interpretation of the results and investigate another dataset.[158] To the best of our knowledge, we are the first RGB-only method to report object-specific results on the challenging sequences of Laval where we test the generalization capabilities of our multi-object model. Please note that the **objects of the Laval dataset have not been seen during training**. The results are summarized in Tab. 8.2 where we also ablate the rendered depth input channel, and Fig. 8.21 shows an example scenario. We follow the evaluation protocol of Garon et al. [136] and report separately the average error for translation and rotation.

Tab. 8.2 shows that our multi-object **model generalizes** well **to** the **unseen objects** of this dataset where the ground truth is acquired with a professional tracking system. Both models are able to track the unseen object in translation. While the full model provides close results both for translation and rotation, the ablated model focuses only on the translation component and predicts stop once the object centre is aligned with only weak corrections for the rotation. Without the depth rendering, the rotational error is significantly larger. Rendering

[158] Cf. Garon, Laurendeau, and Lalonde [136].

the synthetic depth helps with respect to the rotational accuracy. This can be explained by the fact that moving the object in a close proximity to the observation does not require detailed understanding of depth while rotating it correctly is more intricate. In practice we observed that the network first aligns the object in t_x and t_y before correcting rotations and t_z values. We leverage this observation as a tracker initialization after an analysis for robustness and convergence of the method.

8.2.4.6. Robustness & Convergence

The performance of conventional trackers largely depends on the difference between the correct pose and the initialization.[159] As their paradigm is temporally consistent motion in the videos, oftentimes close-to-correct poses are available from the result of the previous frame or they re-initialize with another algorithm.[160] Their **sensitivity to** the **initialization** can result in drift or tracking loss if the seed pose is too far off. Recent methods severely suffer, for instance, if the bounding box overlap is below 50%.[161] Moreover, most conventional 3D trackers are not able to detect whether their estimation is correct or not. In contrast to these methods, we propose a pose estimation pipeline with a large convergence basin that is able to detect its own drift by analysing the number of steps and our stopping criterion.

We test the **convergence radius** of our model by providing different initial poses with gradually increasing deviation from the correct result. After manually checking the ground truth poses of the YCB dataset, we decided to run a robustness test with power drill on all keyframes from video sequence 50 which provides reliable annotations. We prepare initial poses by deteriorating the ground truth annotations with increasing noise from the correct result to an initialization which is 270 actions apart. This is done by adding actions to the GT pose with the state $\left[t_x, t_y, t_z, r_x, r_y, r_z\right]$ in the form of:

$$\Delta \cdot \left[m(t_x), m(t_y), m(t_z), m(r_x), m(r_y), m(r_z)\right], \tag{8.6}$$

$$\text{where} \quad m(s) = m \cdot \operatorname{sgn}(X), \tag{8.7}$$

for all state variables s. We vary the value $m \in \{0, ..., 45\}$ and X is drawn from the uniform distribution $U(-1, 1)$ and determines the direction of corruption. The parameter $\Delta = 6$ determines the deterioration increment for our test.

We use the individually trained model and set the stepsize for all actions to three. Then we run the method and record the average ADD accuracy score as well as the average number of steps in case the model converges to the correct solution. We randomly reduce the amount of keyframes for $m \in \{25, ..., 30\}$ to 25% and for $m \in \{31, ..., 45\}$ to 10% to avoid unreasonably long computations. If convergence is not reached within 200 steps, we treat the run as a fail. The results are summarized in Fig. 8.20. Note that even for a large deviation of $m = 12$ which is significantly larger than the deviation found in the video sequence, our accuracy is ADD = 73.8%. Moreover, we can also see reasonable convergence in cases with 50% or fewer

[159] Cf. Akkaladevi et al. [4].
[160] Cf. Deng et al. [82].
[161] Cf. Garon and Lalonde [135] as well as Garon, Laurendeau, and Lalonde [136].

	Ours full				Ours w/o D			
Occlusion	0%	15%	30%	45%	0%	15%	30%	45%
Clock								
T[mm]	14.02	20.54	25.85	51.92	9.39	9.96	32.58	15.91
R[deg]	9.40	10.84	12.74	17.05	29.15	27.92	30.72	28.40
Cookie Jar								
T[mm]	3.82	5.99	9.52	15.18	1.79	2.75	11.62	5.95
R[deg]	6.48	17.82	18.22	15.89	28.77	18.18	24.30	19.02
Dog								
T[mm]	12.09	28.37	55.48	77.91	6.10	10.76	33.89	15.62
R[deg]	11.70	14.21	22.43	23.80	20.75	26.81	24.22	22.53
Dragon								
T[mm]	22.47	29.39	36.37	40.06	25.69	25.13	27.71	30.65
R[deg]	3.34	4.89	11.65	13.39	27.16	36.40	37.61	30.94
Shoe								
T[mm]	9.72	17.91	24.33	37.34	44.61	19.90	38.04	41.90
R[deg]	5.84	9.26	17.89	16.91	62.78	39.47	43.50	24.73
Turtle								
T[mm]	5.92	9.91	12.91	23.92	5.53	6.37	16.14	12.63
R[deg]	7.09	14.87	14.87	14.11	18.31	20.13	26.03	24.97
Walkman								
T[mm]	8.74	18.93	31.98	45.13	11.63	15.63	20.12	31.30
R[deg]	6.97	11.33	21.17	22.26	40.68	44.47	50.18	45.14
Watering Can								
T[mm]	14.67	21.66	18.68	33.26	11.61	20.54	20.96	26.10
R[deg]	11.89	19.80	23.43	33.54	38.89	40.85	36.30	35.23

Tab. 8.2. Evaluation results on Laval dataset for different levels of noise. We compare the full model to a model without rendered depth input.

bounding box overlap where other methods[162] struggle and drift.
We use this wide convergence basin to show that our framework can be modified without retraining to also provide an initial pose close to the correct one after further investigation of failure cases and runtime.

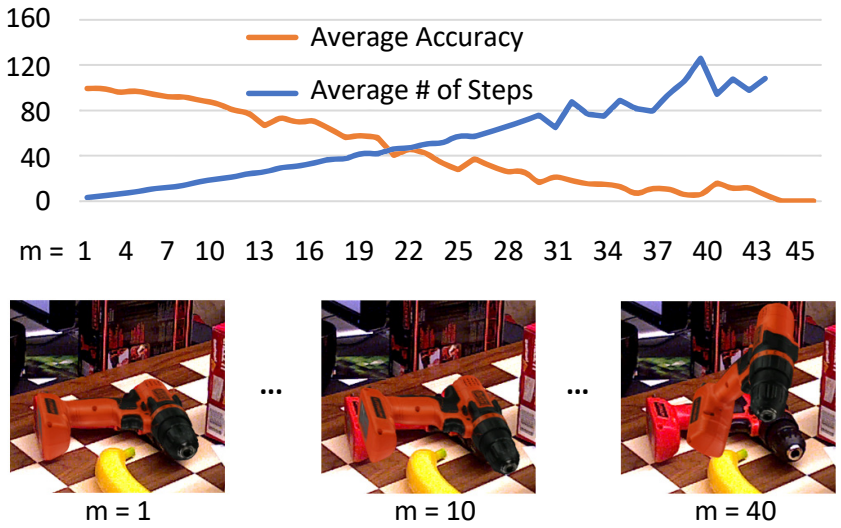


Fig. 8.20. **Sensitivity of pose decision process to initial pose.** The average ADD score (orange) is shown for increasing deviations from the ground truth while the average number of steps the method needed for convergence is illustrated in blue. For deviations with $m \geq 43$ the method did not converge within 200 steps. The lower part illustrates some examples with increasing deterioration as used as an initialization for the robustness test.

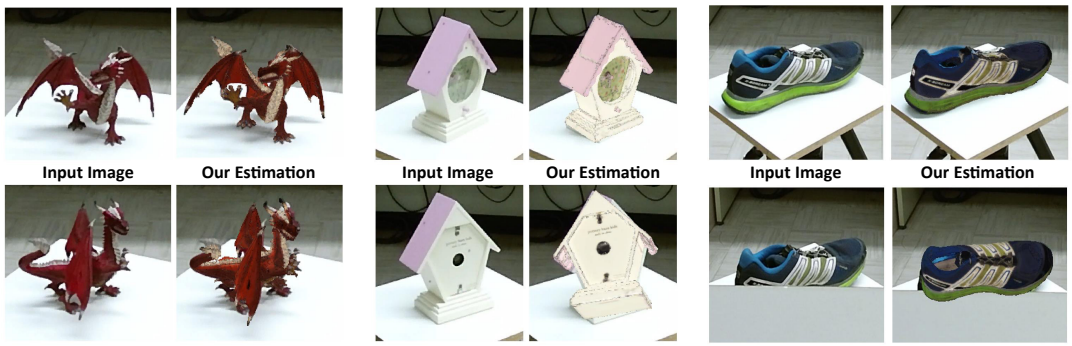


Fig. 8.21. **Pose estimation examples on Laval dataset.** We show some prediction examples from the Laval dataset. Upper row: Example predictions of unseen objects. Lower row: Self-occluded fine details (left), low texture (middle) and occlusions (right) can cause pose estimation failure for unseen objects.

Failure Cases. Even though the convergence of our method is reliable in most cases, the network capacity is limited. This results in pose estimation failures in case of heavy occlusions and fine detailed geometry. Moreover, we share the issue with other RGB-only methods that low-textured objects are difficult to estimate reliably which results in drift in some cases as depicted in Fig. 8.21 together with further examples.

Runtime. The structure of our approach allows for automatic **dynamic runtime improvements** in case of limited motion present in the scene. Since the number of iteration steps is

[162] Cf. Garon and Lalonde [135].

non-static and the 3D rendering is negligible for this comparison, the overall runtime depends on two parameters: the action decision cycle and the number of actions. In our current implementation, the runtime for one loop in the cycle breaks down in the image preprocessing done on CPU and the inference on the GPU. We performed a runtime test averaging 512 iterations. The results are shown in Tab. 8.3.

Average Runtime on	CPU	GPU	Total
Average Runtime in ms	14.6	5.2	19.8

Tab. 8.3. **Average runtime of action decision process cycle.**

Given the average of 4.2 actions on our YCB tests, we report an overall average runtime of 83.16 ms or 12 fps. Note that all tests are done on a laptop and the runtime could be increased if a more powerful desktop machine is used or the image processing was also ported to the GPU.

8.2.4.7. Initialization & Detection

Tracking is often done by detection[163] or with the help of a depth map.[164] However, Deng et al. [82] recently proposed an RGB-only tracking solution.
Other pose refinement models like the ones introduced by Manhardt et al. [273] for Kehl et al. [210] or Li et al. [251] for Xiang et al. [454] require an initial detector. We empirically observed that the model tends to first align the rendering for the translation and performs rotation actions afterwards. We make use of this observation and **run our network without retraining** with multiple seeds **as** a **pose detection pipeline** omitting the use of another model. For this, we randomly chose an object pose and seed the image at different locations by changing t_x and t_y for the pose. We then run one iteration of the network in every location and record just the values for t_x and t_y. We normalize the 2-vector given these inputs and generate a sparse vector field $\mathbf{V}$ on top of the image as shown in Fig. 8.22 where we place these vectors at the seed centres. This vector field is rather random for non-overlapping regions while its flux points toward the projection centre of the object if visible. Applying a divergence operation $\mathrm{W} = \nabla \cdot \mathbf{V}$ on the smoothed vectors allows to find the object centre as the maximum of W. Analyzing W helps also to determine a valuable bounding box size for a first crop. Running the method on a coarsely discretized rotation space in this crop allows to find an initial rotation as shown in Fig. 8.22 where the minimum number of iteration positively correlates with a possible starting rotation. As the initial seeds can be calculated independent from each other, this process can also be parallelized.

[163] Cf. Crivellaro et al. [74] as well as Xiang et al. [454].
[164] Cf. Garon, Laurendeau, and Lalonde [136].

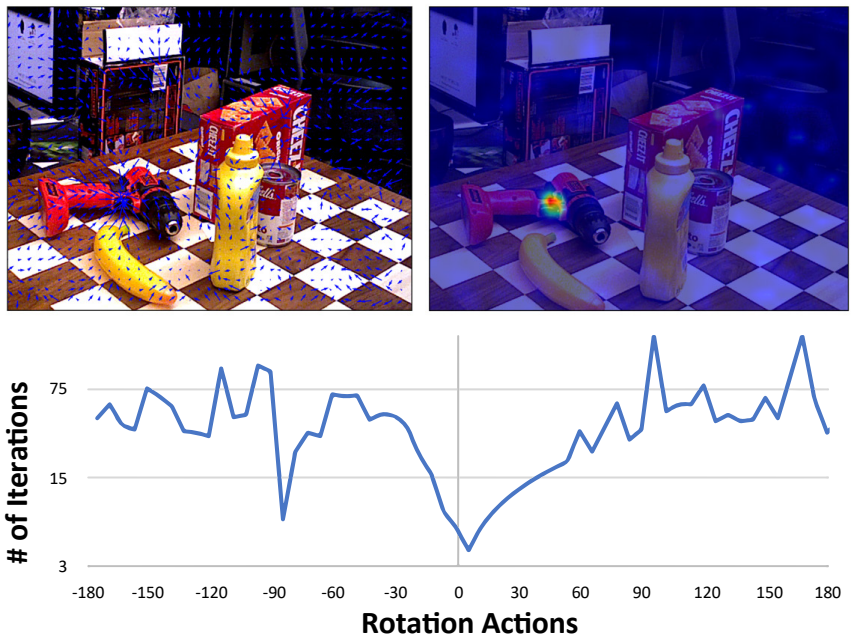


Fig. 8.22. **Initial point & rotation seeding.** The predictions for t_x and t_y generate a vector field over the image (top left) whose divergence (top right) determines the initial point. Seeding a random rotation at this point allows to calculate the initial pose. The necessary number of iterations before stop is predicted is plotted (bottom) against different seeds at a certain deviation from this rotation in just one action parameter (in this case r_z). A good initialization in the example is +5 actions away where the curve has its minimum.

8.2.4.8. Retrospective & Outlook

We reformulated 6D pose estimation as an action decision process and presented a pipeline to solve it as a generic task without the need for object-specific training. The method provides a dynamic runtime complexity depending on the inter-frame motion to increase runtime performance and it generalizes to unseen objects. However, while improving the state-of-the-art for RGB-based video pose estimation, it still struggles in challenging cases for unseen objects. Currently we search for the next best pose in every step. An interesting direction for future research could be to integrate built up knowledge over time leveraging e.g. reinforcement learning.

In this chapter, we studied the minimal case where the only source of information for our 6D pose estimation comes from pixel intensities of captured unseen environments leveraging greyscale binocular stereo in an inside-out setup or unseen objects with outside-in vision and a monocular RGB camera. We now want to make use of pose estimators as developed so far in order to practically fuse sensor information. For this, we also tackle pose noise and interpolation to enable a synchronized combination of multi-modal information in the next chapter.

Part IV

Sensor Fusion

9 Pose Modifications

> *Don't complain about things*
> *you're are not willing to change.*
>
> **– English Proverb**

A plethora of sensors are available in our everyday life. With the advances in modern smartphones and intelligent cars, videos, images, GPS and many other sensors are more mobile than ever before and their data provides inherent information on their location and relative poses. When we want to combine information from multiple spatial locations, we have to answer a set of questions to align the input from diverse sources correctly in order to improve the overall output or combine information to gain further insight.

Independent of the sensor modality or sensing frequency it is essential to know the relative pose between the devices to represent the data in a common reference frame. If the various sensors are mobile, this can be a dynamic parameter and accurate pose estimation methods help for better fusion. In an ideal setup, the acquisition is synchronized. From a hardware perspective this can be realized with sensors of the same acquisition frame rate and a trigger mechanism analogously to the stereo setup described in section 7.1.1. However, the hardware of acquisition devices is often closed and a software trigger remains the only option or it is necessary to run devices in streaming or free-run mode where no active triggering is possible. Moreover, the nature of the acquisition or the exposure process may not allow for real-time image broadcasting or it adds a slight delay. This results in images and poses often being provided at different times and with inconsistent frequencies or even with changing temporal offset. Online use of fusion systems severely suffer from these discrepancies that manifest in jitter, lag and incorrect augmentations which ultimately can harm a medical diagnosis.
To pragmatically address this, we investigate necessary pose corrections to provide **computational synchronization** through pose modification with interpolation and extrapolation methods and minimize the noise from optical tracking systems in this chapter. We then discuss how various sensor inputs can improve overall pose estimates and consecutively analyse spatial combinations of input signals in chapter 10.

The pose estimation pipeline is the backbone for spatially correct modality fusion. As such, it is important to know the pose of involved sensors at a given time correctly and accurately in order to place images in a joint visualization at the right place.
The time of a calculated pose usually does not coincide with the time of the sensor acquisition. In order to efficiently assign a pose to an acquired image, we correct its parameters in an online approximation step using time-based **geometric interpolation and extrapolation** methods which we present in section 9.1. We further aim to **remove noise** from the pose estimation step

using temporal smoothness constraints tailored for our (dual) quaternion pose parametrization in section 9.2 before we finally discuss ways to improve the signal through input and output **consistencies** in section 9.3.

9.1. Interpolation and Synchronization

The use of multiple devices and machines together with communication-induced lags and different measurment frequencies causes images and pose estimations to be acquired at unequal times. While the optical tracking system as defined in section 7.1 operates at a frequency above 20 fps with an exposure time of only 1.5 ms, other modalities such as the ones typically used in nuclear medical imaging can require much longer integration times in the order of seconds. On the other side of the spectrum, IMUs acquire information with more than 100 Hz[1] and event cameras provide an asynchronous stream of measurements up to the order of kHz.[2]

We identify fast tracking as an essential part in real-time 3D vision pipelines and improve the acquisition rate of optical tracking systems computationally. This allows to keep the advantages of accurate and reliable measurements while overcoming sensor limitations and physical constraints due to exposure time. Moreover, we address the problem of transmission lags and latencies by signal series extrapolation using the mathematical foundation of (dual) quaternions built in sections 6.2.3 and 6.2.5.
We leverage the fact that our OTS uses a dual quaternion based pose parametrization in its backbone algorithm as described in section 7.3 and introduce **quaternionic upsampling** in section 9.1.2 to increase the pose frequency on the fly.
We formulate different upsampling strategies on Riemannian manifolds to describe poses as points on a multidimensional hypersphere and a quadric in (dual) quaternion space. We interpolate piecewise continuous curves along geodesics and study computational complexity and accuracy of the results. Quaternionic upsampling allows for unified interpolation and extrapolation of pose series with just one parameter where linear variations directly translate into continuous pose changes with pose rates of over 4 kHz. Extrapolations with an accuracy of 128 μm and 0.5° can be realized online. While the method is designed for our optical tracking system and leverages the internal pose parametrization, it is generic and can speed up any 6 DoF rigid pose tracker or 3 DoF rotation estimation system.

9.1.1. Quaternion Interpolation Techniques

Losing pose information due to communication lag and data corruption is a relevant problem that limits the use of applications that depend on robotic manipulators and optical systems in practice.[3] Pose extrapolation can be an adequate tool to prevent tracking failures and mitigate pose loss through dead reckoning in small temporal windows. Through time stamp based pose calculations one can interpolate poses in retrospect to further help alignment of pose and

[1] Cf. Potter et al. [333].
[2] Cf. Scheerlinck, Barnes, and Mahony [367] as well as Gallego et al. [132].
[3] Cf. Johnson and Somu [197].

image streams and to synchronize movements of individual coordinate frames in real-time.[4] Computational increase of pose acquisition rates helps also multi-modal systems where data streams such as visual and inertial sensors provide information on different time scales.[5]

The fact that a **quaternion** rotation representation avoids gimbal lock[6] makes the parametrization interesting for interpolation tasks. Geodesic trajectories on the 3-dimensional hypersphere S^3 have been presented by Shoemake [374] as an efficient first order interpolation of rotations. The method is often referred to as spherical linear interpolation (SLERP) when sampled at constant speed. Interpolation applications on quaternion space $\mathbb{H}$ are also used in computer animations[7] and in virtual reality scenarios.[8]

Dual methods are not as frequently explored for pose interpolation despite their advantages. The graphics community uses **dual quaternions** more often. Kuang et al. [230] use $\mathbb{DH}_1$ for real-time motion animation of clothed body movements and Kavan et al. [207] apply the advantageous dual formulation for skinning. Dual methods are used in computer animation for smooth blending[9] and in Busam et al. [50], we describe how to extrapolate movements in real-time vision systems with dual quaternions. Moreover, complex hierarchical rigid body transforms can be efficiently represented with $\mathbb{DH}_1$.[10]
Interpolation between key frames is an essential ingredient in computer animation[11] where computationally efficient and physically plausible methods are important and the task has similar requirements to our pose parameter interpolation where an initial and an end pose determine intermediate poses. Instead of direct estimation of interior points on the Lie group between measurements, other approaches focus on optimization of support quaternions to fit an interpolated trajectory. Aside of our investigation, **extrapolation** with quaternions is not much explored. One of the few works by Chui et al. [70] explores a virtual reality application. They estimate a first initial pose for a smooth trajectory which is consecutively refined in the context of repetitive motion.

Higher order pose interpolation can also be efficiently implemented and used to describe **continuous time pose trajectories**.[12] Interpolation techniques for quaternions are started with the work of Shoemake [375] who proposes a bilinear parabolic blending scheme of four base quaternions which are positioned at the corners of a quadrangle. According to the construction scheme, it is sometimes referred to as spherical cubic spline quadrangle (SQUAD).[13] The resulting curve has C^1-continuity. Also Barr et al. [10] use multiple quaternions and interpolate a spline including velocity constraints and Kim et al. [218] form a spline on $\mathrm{SO}(3)$ with a cumulative basis. Their B-spline curve has local control and is C^2. A tension parameter is explored by Nielson [306] who performs key-frame animation with ν-quat splines. Spline Fusion[14] uses a continuous time representation to combine IMU and vision data from a rolling shutter camera as also done by Patron-Perez et al. [324]. They use a cumulative B-spline on

[4] Cf. Esposito et al. [101].
[5] Cf. Foxlin et al. [121].
[6] Cf. Lepetit and Fua [244].
[7] Cf. Kavan [205].
[8] Cf. Song et al. [388].
[9] Cf. Pennestrì and Valentini [327].
[10] Cf. Kenwright [214].
[11] Cf. Parent [320].
[12] Cf. Haarbach, Birdal, and Ilic [157].
[13] Cf. Dam, Koch, and Lillholm [78].
[14] Cf. Lovegrove, Patron-Perez, and Sibley [263].

SE(3). An arbitrary spline order can be calculated on Lie groups with the work of Sommer et al. [387] and a survey of higher order interpolation techniques is formulated by Haarbach et al. [157].

Visual-inertial SLAM methods benefit from a change of the representation from discrete poses to a continuous representation.[15] Since the IMU acquires signals at a much higher frequency, it is helpful to interpret the rolling shutter camera sensor as a higher frequency signal in their formulation. Despite the improvement, singularities in their interpolation scheme remain.[16] Direct interpolation on SE(3) can be detrimental from a practical perspective and it can be helpful to split non-robotic motion interpolation tasks into rotation and translation.[17]

9.1.2. Quaternionic Upsampling

We investigate a unified approach that is capable of both interpolation between past measurements and extrapolation of future poses in (dual) quaternion space with which both rotation and translation can be efficiently computed. This represents to the best of our knowledge the first method that uses a combined physical motivation and differential geometry approach for **joint interpolation and extrapolation** of 6 DoF rigid body movements.

We start off by defining a general upsampler in pose space which is used to define a set of methods for interpolation and extrapolation. The methods build on established pose parametrizations as discussed in section 6.2 and are consecutively evaluated.

Definition 9.1
*An **upsampler** $\gamma \in C^0$ from two transformations ϕ_1 and ϕ_2 of pose parameter space $\mathbb{Q}$ can be defined as*

$$\gamma : \mathbb{Q} \times \mathbb{Q} \times \mathbb{R}_0^+ \to \mathbb{Q} \tag{9.1}$$

$$(\phi_1, \phi_2, \tau) \mapsto \gamma(\phi_1, \phi_2, \tau) \tag{9.2}$$

with

$$\gamma(\phi_1, \phi_2, 0) = \phi_1, \tag{9.3}$$

$$\gamma(\phi_1, \phi_2, 1) = \phi_2. \tag{9.4}$$

Our observation range is $\tau \in [0,1]$ and the interval $[0,\infty]$ provides the sampling space. We interpolate with $\tau \in [0,1]$ and perform an extrapolation for $\tau > 1$. Since optical tracking systems operate at constant frame rates, we mostly extrapolate with $\tau \in (1,2)$ before the new pose measurements arrives around $\tau = 2$. A longer dead reckoning phase, however, can be necessary in case of network lag, occlusions or other disruptions. Interpolation constitutes a refinement of poses between two measurements and extrapolation gives an estimate beyond the

[15] Cf. Furgale, Barfoot, and Sibley [128] as well as Furgale et al. [129].
[16] Cf. Haarbach, Birdal, and Ilic [157].
[17] Cf. Ovrén and Forssén [318].

observable range. If the pose is parametrized by a separate rotation and translation component, the translation part is equally treated with a linear interpolation between the supporting estimates. More elaborate techniques beyond linear interpolation can preserve additional kinematic properties other than continuous motion. This can be continuity in the velocity or angular momentum ultimately leading to C^n-continuous pose interpolations. For efficiency reasons, we want to concentrate on C^0-continuity and focus on the more complex rotation part first before we jointly encapsulate both translation and rotation in a joint upsampling and pose refinement formulation with dual quaternions.

We develop a set of upsampling strategies in the next sections. The **angular velocity** v of a potentially upsampled pose sequence $(\phi_n)_n$ that uses parameter space $\mathbb{Q}$ can be visualized using centred averages

$$v : \mathbb{Q} \to \mathbb{R} \tag{9.5}$$

$$\phi_i \mapsto v(\phi_i) := \frac{\|\phi_i - \phi_{i-1}\| + \|\phi_i - \phi_{i+1}\|}{2}. \tag{9.6}$$

Note that a sequence of n samples does not provide angular velocity values for $v(\phi_0)$ and $v(\phi_n)$ with this definition.

To visually compare the results of 6 DoF pose upsamplers, we concentrate on the more interesting rotational part. Leveraging quaternion representations for visualization here is intricate as quaternion space $\mathbb{H}$ has 4 dimensions. We instead separately illustrate a projection of the rotation axis onto the sphere S^2 together with an angular velocity plot. For a pose sequence of four example poses we show in Fig. 9.1 the results of different upsamplers we discuss consecutively together with their individual rotation angle change. Further examples and the effect of different upsampling strategies on rigid body motion for an object are exemplified in a supplementary video which is available online.[18]

9.1.3. Euler Angles & Rotation Matrices

The generic formulation allows also to interpolate non-quaternion rotation representations such as Euler angles and rotation matrices. Suppose we use the parameter vector for Euler angles $\mathbf{a}_i = \left(\alpha_i^x, \alpha_i^y, \alpha_i^z\right)$ where two rotations are given by $\mathbf{a}_1$ and $\mathbf{a}_2$, we can frame **Eu**ler angle **L**inear **Up**sampling as

$$\mathrm{EuLUp}(\mathbf{a}_1, \mathbf{a}_2, \tau) := (1-\tau)\,\mathbf{a}_1 + \tau \mathbf{a}_2. \tag{9.7}$$

An interpretation of the resulting interpolations can be counter-intuitive as the trajectory of the rotation axis may not follow the shortest path from its source to its target position. In fact, it may take a detour. This inhibits the physical interpretation of the result as shown in Fig. 9.1.

[18] The video can be found under `http://campar.in.tum.de/Chair/PublicationDetail?pub=busam2016_3dv`.

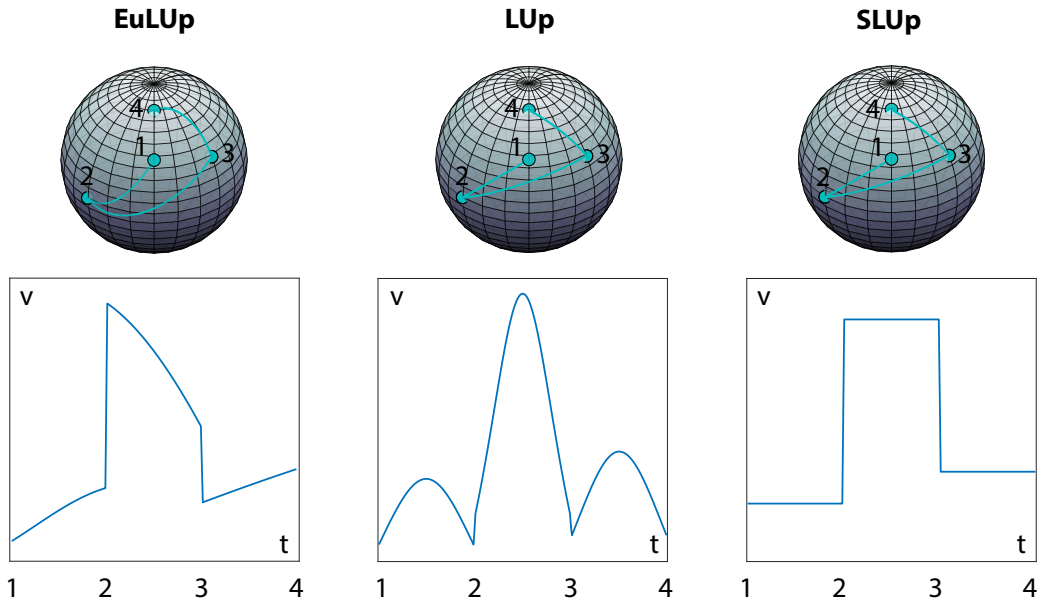


Fig. 9.1. **Pose upsampling methods and resulting velocity.** The three pose upsampling methods EuLUp, LUp, SLUp are compared for the four pose observations $1, 2, 3, 4$. The spheres illustrate the rotation axis with the green points indicating the orientation of the axis at start and end of the interpolation. The accompanying graphs visualize the velocity v of the interpolated motion where the time steps $t \in \{1, 2, 3, 4\}$ coincide with the pose observations. The interpolation with EuLUp (left) causes nonlinear velocity changes together with an axis motion that is not following the shortest path between interpolation points. The interpolated rotation axes in LUp (centre) follow the shortest path. However, the velocity increases and decreases between pose measurements with the maximum velocity in the middle. SLUp (right) results in constant rotation velocity between poses and shortest path motion.

A potential direction could be the use of rotation matrices. If we interpolate on the parameter space of matrices, we can formulate element-wise linear upsampling for rotation matrices as

$$\mathrm{MLUp}(\mathbf{M}_1, \mathbf{M}_2, \tau) := (1-\tau)\,\mathbf{M}_1 + \tau \mathbf{M}_2, \tag{9.8}$$

with $\mathbf{M}_i \in \mathbb{R}^{3\times 3}$, $i \in \{1, 2\}$ and the constraints $\mathbf{M}_i^{\mathrm{T}} = \mathbf{M}_i^{-1}$ as well as $\det(\mathbf{M}_i) = 1$. There is no guarantee that the resulting matrix $\mathbf{M}_\tau = \mathrm{MLUp}(\mathbf{M}_1, \mathbf{M}_2, \tau)$ fulfills the orthonormal constraint and is therefore not necessarily an element of $\mathrm{SO}(3)$. Geometry changes and object scaling immediately cause problems in this case and we do not want to investigate re-orthogonalization methods here. Thus, we consider the space of quaternions $\mathbb{H}_1$ as a next step.

9.1.4. Quaternions

Normalized **L**inear **Up**sampling (LUp) between the quaternion rotations $\mathbf{q}_1$ and $\mathbf{q}_2$ in $\mathbb{H}$ can be written as

$$\mathrm{LUp}(\mathbf{q}_1, \mathbf{q}_2, \tau) := \frac{(1-\tau)\,\mathbf{q}_1 + \tau \mathbf{q}_2}{\left\|(1-\tau)\,\mathbf{q}_1 + \tau \mathbf{q}_2\right\|}. \tag{9.9}$$

The interpolated rotations are given by the points on the straight line connecting the quaternions $\mathbf{q}_1$ and $\mathbf{q}_2$ in $\mathbb{H}$ which are normalized. However, the described trajectory follows the

shortest path in $\mathbb{H}$ connecting these points before normalization rather than staying in $\mathbb{H}_1$. In Fig. 9.2 we illustrate the process where the interpolated points are projected onto the sphere $\mathbb{H}_1$. The resulting trajectory on $\mathbb{H}_1$ coincides with the geodesic describing the shortest path from $\mathbf{q}_1$ to $\mathbf{q}_2$. The projection step, however, causes a variation in sampling rate on this trajectory resulting in a continuous increase of angular velocity until the middle point and an equal decrease afterwards. The interpolation method involves little computation and it can still serve as an efficient solution in time-critical applications where a large amount of interpolations need to be calculated or hard runtime constraints limit the allowed computation complexity. For extrapolation, this may be problematic as illustrated also in Fig. 9.2.

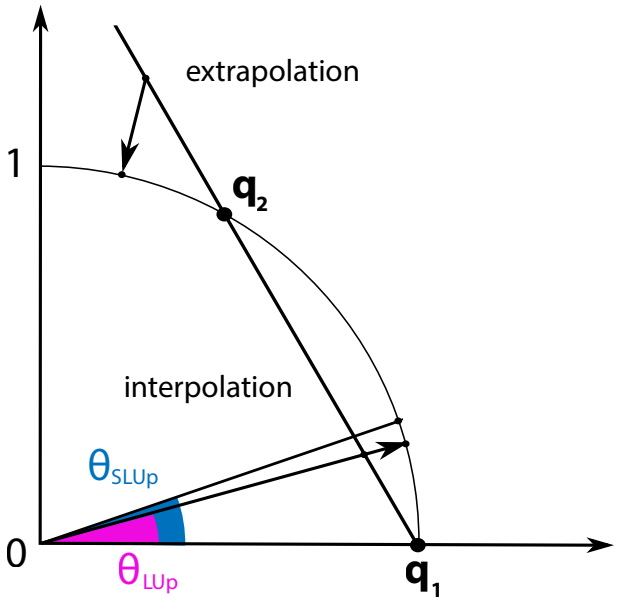


Fig. 9.2. **Linear (LUp) and Spherical Linear (SLUp) upsampling in $\mathbb{H}_1$.** We illustrate a partial cut through the quaternion hypersphere $\mathbb{H}_1$ on which the two quaternions $\mathbf{q}_1$ and $\mathbf{q}_2$ lie. Interpolation steps for direct upsampling (SLUp) along the geodesic thin line in $\mathbb{H}_1$ produce equidistant points along the arc. The faster linear upsampling (LUp) is reached by interpolation (and extrapolation) on the straight line with consecutive projection onto the arc through normalization. While both interpolation methods retrieve points along the geodesic, the interpolated rotations can vary. For the same interpolant $\tau \in \mathbb{R}_0^+$ two results can have different geodesic distance from $\mathbf{q}_1$ and $\mathbf{q}_2$ as illustrated here in blue and pink colour for the angles θ_{SLUp} and θ_{LUp}. This discrepancy increases with the distance between the line and the arc making extrapolations with LUp far beyond $\mathbf{q}_2$ impractical.

We further want to describe the interpolation along the geodesic line and use the definitions for the exponential and logarithm map in quaternion space as introduced in chapter 6.2.6 to define an exponentiation for unit quaternions:

Definition 9.2
*For $\boldsymbol{q} \in \mathbb{H}_1$, we define the **exponentiation** of $\boldsymbol{q}$ with the real exponent $\tau \in \mathbb{R}_0^+$ as*

$$\boldsymbol{q}^\tau := \exp(\tau \log(\boldsymbol{q})). \tag{9.10}$$

With this definition, we can look at a unit quaternion $\mathbf{r} \in \mathbb{H}_1$ of the form

$$\mathbf{r} := [\cos(\theta), \sin(\theta)\mathbf{v}] \tag{9.11}$$

with $\mathbf{v} \in \mathbb{R}^3$, $\|\mathbf{v}\| = 1$ and $\theta \in \mathbb{R}$. It holds with $\tau \in \mathbb{R}_0^+$:

$$\mathbf{r}^\tau = \exp(\tau\,[0, \theta\mathbf{v}]) \tag{9.12}$$

$$= [\cos(\tau\theta), \sin(\tau\theta)\mathbf{v}] \tag{9.13}$$

and we can write **S**pherical **L**inear **Up**sampling for the quaternions $\mathbf{q}_1, \mathbf{q}_2 \in \mathbb{H}_1$ as

$$\mathrm{SLUp}(\mathbf{q}_1, \mathbf{q}_2, \tau) := \mathbf{q}_1 \cdot (\bar{\mathbf{q}}_1 \cdot \mathbf{q}_2)^\tau. \tag{9.14}$$

The upsampling method SLUp interpolates along the great arcs on the quaternion hypersphere in $\mathbb{H}_1$ using the geodesic path between $\mathbf{q}_1$ and $\mathbf{q}_2$ and extrapolates further along the arc. The basis for SLUp is the interpolation step which was first introduced by Shoemake [374] as Spherical Linear Interpolation (SLERP).
In Fig. 9.1, we see the change of the angular velocity for the projection dependent LUp interpolation in an example case in comparison with the SLUp method which provides a constant speed while both their interpolation paths along the geodesic remain the same. We combine translational and rotational components now and transfer the ideas to dual quaternion space $\mathbb{DH}$.

9.1.5. Dual Quaternions

The representation of rigid body motion with dual quaternions has some efficiency and compactness advantages over using homogeneous matrices.[19] We leverage this to define an efficient upsampler.
Analogously to the quaternion case and with the concepts and definitions of the differential operators from chapter 6.2.5, we formulate an exponentiation on $\mathbb{DH}_1$.

Definition 9.3
*For $\mathbf{Q} \in \mathbb{DH}_1$, we define the **exponentiation** of $\mathbf{Q}$ with the real exponent $\tau \in \mathbb{R}_0^+$ as*

$$\mathbf{Q}^\tau := \exp(\tau \log(\mathbf{Q})). \tag{9.15}$$

A unit dual quaternion $\mathbf{Q} \in \mathbb{DH}_1$ can be written in its standard form given in equation (6.64)[20] as

$$\mathbf{Q} = [\cos(\Theta), \sin(\Theta)\mathbf{V}] \tag{9.16}$$

[19] Cf. Funda, Taylor, and Paul [127].
[20] Cf. Daniilidis [80].

with the dual entities $\Theta \in \mathbb{D}\mathbb{R}$ and $\mathbf{V} \in \mathbb{D}\mathbb{H}_1$ from quatations (6.60) and (6.61) defined as

$$\Theta = \theta + \varepsilon\theta_\varepsilon \tag{9.17}$$

$$\mathbf{V} = \mathbf{v} + \varepsilon\mathbf{v}_\varepsilon. \tag{9.18}$$

We can then write

$$\mathbf{Q}^\tau \overset{(6.64)}{=} \exp(\tau \log([\cos(\Theta), \sin(\Theta)\mathbf{V}])) \tag{9.19}$$

$$\overset{(6.86)}{=} \exp(\tau\mathbf{V}\Theta) \tag{9.20}$$

$$\overset{(6.84)}{=} \cos(\tau\Theta) + \sin(\tau\Theta)\mathbf{V} \tag{9.21}$$

with the dual trigonometric operators from equations (6.62) and (6.63).
This can then be interpreted with the screw linear displacement view on dual quaternions (cf. Fig. 6.8) where a linear variation of the interpolant $\tau \in \mathbb{R}_0^+$ causes a linear change of the translation component in the direction of $\mathbf{v}$ together with a rotation about the screw axis with constant speed.

Extending SLUp with this insight into dual quaternion space, allows to formulate screw linear upsampling **ScLUp**. Let us consider the two dual quaternions $\mathbf{Q}_1$ and $\mathbf{Q}_2$ of unit length in displacor form as proposed in equation (6.31) with

$$\mathbf{Q}_1 = \mathbf{r}_1 + \frac{\varepsilon}{2}\mathbf{t}_1\mathbf{r}_1 \tag{9.22}$$

$$\mathbf{Q}_2 = \mathbf{r}_2 + \frac{\varepsilon}{2}\mathbf{t}_2\mathbf{r}_2 \tag{9.23}$$

together with the interpolant $\tau \in \mathbb{R}_0^+$. Then we can define

$$\mathrm{ScLUp}(\mathbf{Q}_1, \mathbf{Q}_2, \tau) := \mathbf{Q}_1 \cdot \left(\mathbf{Q}_1^{-1} \cdot \mathbf{Q}_2\right)^\tau, \tag{9.24}$$

where $\mathbf{Q}_1^{-1} \in \mathbb{D}\mathbb{H}_1$ represents the inverse displacement. Since $\bar{\mathbf{t}} = -\mathbf{t}$, we can also write

$$\mathbf{Q}_1^{-1} = \bar{\mathbf{r}}_1 + \frac{\varepsilon}{2}\bar{\mathbf{r}}_1\bar{\mathbf{t}}_1 = \bar{\mathbf{Q}}_1 \tag{9.25}$$

and thus it holds

$$\left(\bar{\mathbf{Q}}_1 \cdot \mathbf{Q}_2\right)^\tau = \left(\left(\bar{\mathbf{r}}_1 + \frac{\varepsilon}{2}\bar{\mathbf{r}}_1\bar{\mathbf{t}}_1\right) \cdot \left(\mathbf{r}_2 + \frac{\varepsilon}{2}\mathbf{t}_2\mathbf{r}_2\right)\right)^\tau \tag{9.26}$$

$$= \left(\bar{\mathbf{r}}_1\mathbf{r}_2 + \frac{\varepsilon}{2}\left(\bar{\mathbf{r}}_1\mathbf{t}_2\mathbf{r}_2 - \bar{\mathbf{r}}_1\mathbf{t}_1\mathbf{r}_2\right)\right)^\tau \tag{9.27}$$

$$= \left(\bar{\mathbf{r}}_1\mathbf{r}_2 + \frac{\varepsilon}{2}\bar{\mathbf{r}}_1\left(\mathbf{t}_2 - \mathbf{t}_1\right)\mathbf{r}_2\right)^\tau \tag{9.28}$$

which represents the rigid displacement between $\mathbf{Q}_1$ and $\mathbf{Q}_2$. Thus, ScLUp gives the shortest path with a screw linear displacement between the two dual quaternions. Equidistant samples of parameters $\tau \in \mathbb{R}_0^+$ then result in a translation and rotation with constant speed interpolating between $\mathbf{Q}_1$ and $\mathbf{Q}_2$ for $\tau \in [0, 1]$ and extrapolation with a screw linear motion for $\tau > 1$.

Instead of treating translation in $\mathbb{R}^3$ and rotation with a quaternion from $\mathbb{H}_1$ separately, this allows for unified upsampling in $\mathbb{D}\mathbb{H}_1$ resulting in an efficiency advantage over the former approaches as we analyze hereafter. However, for critical high speed applications and constraint

hardware with limited computation capabilities, an approximate solution that safes computation time may be required. Taking the attractive computational savings from LUp into account, we also approximate an interpolation using a normalized **D**ual quaternion **L**inear **Up**sampling (**DLUp**). The approach is used for interpolation in 3D animation[21] where dual quaternion linear blending (DQLB) improves the frame rate and helps to refine character poses.[22] In our case, the approximation reads as

$$\mathrm{DLUp}(\mathbf{Q}_1, \mathbf{Q}_2, \tau) := \frac{(1-\tau)\mathbf{Q}_1 + \tau\mathbf{Q}_2}{\|(1-\tau)\mathbf{Q}_1 + \tau\mathbf{Q}_2\|}. \tag{9.29}$$

If the displacement between pose measurements is not too far and the update rate for pose estimates is fast, **DLUp** closely approximates ScLUp while being more efficient. However, angular differences between the two methods of up to 8.15° for interpolation scenarios are possible.[23]

9.1.6. Pose Stream Synchronization

With the tools we have developed so far, it is possible to compute poses between measurements and beyond the latest estimate. This is a core necessity for systems built from various sensor that are not perfectly synchronized in time or run at different sampling rates. However, to leverage the upsampling methods, it is crucial that the different systems are temporally calibrated such that different data can be fused with the correct time stamps. If various pose estimation systems are involved, a **temporal calibration** is possible by automatically finding a constant temporal offset that solves the optimization problem

$$\min_{\Delta t \in \mathbb{R}} \left\| \mathbf{P}(t)_{\mathrm{A}}^{\mathrm{W}} - \mathbf{P}(t+\Delta t)_{\mathrm{B}}^{\mathrm{W}} \right\| \tag{9.30}$$

for the poses $\mathbf{P}$ in some world reference system W when they come from the two estimations A and B. This is illustrated in Fig. 9.3 where the shift of the belayed measurements from system B is shown with a constant temporal offset. While the sampling rate of one system may vary, pose upsampling in t or a continuous time assumption allows to calculate a constant delay.

Such a **temporal offset** can arise from delays caused by the runtime of different pose estimation algorithms that do not correct the time stamp of the estimated displacement or do not provide temporally corrected poses. It can also be influenced by network delays when the optical tracking systems send the pose over an interface such as the one described in chapter 7.6 and thus can depend on the network traffic. Non-synchronized system clocks additionally affect the time offset when several computers are involved. Such delays are in practice not constant and vary during a measurement procedure in particular if we use pose estimations from a closed system. We therefore generally take a two step approach where we first synchronize the clocks of the involved machines and consecutively minimize a constant offset continuously if necessary. If we have full control of the pose estimation pipeline we can minimize the correction steps by using the time stamp of the image acquisition from our OTS as the time stamp of the calculated pose rather than the time when the pose has been calculated.

[21] Cf. Feng and Wan [108].
[22] Cf. Kavan et al. [206].
[23] Cf. Kavan and Žára [208].

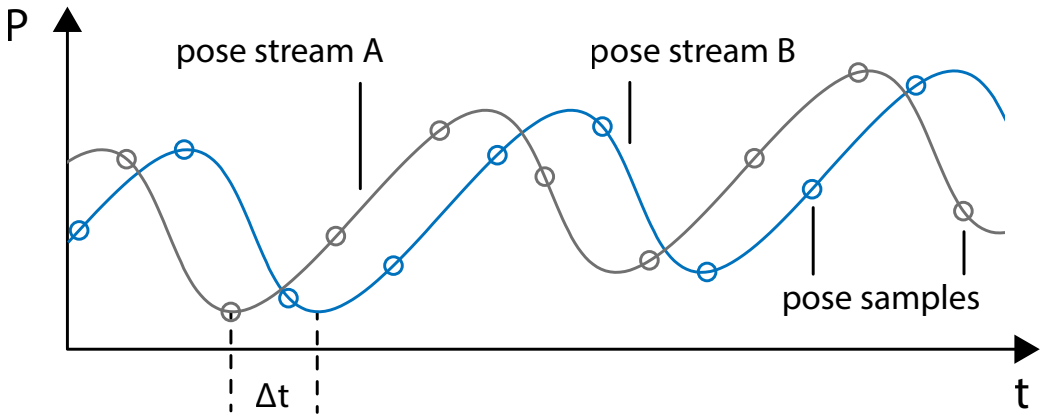


Fig. 9.3. **Time shift of pose streams.** An object motion is captured by the two pose estimation systems A and B in the same world reference coordinates. The object motion at different time steps for the two systems is shown in grey and blue colour and the pose samples taken from the tracking systems by pose estimation are illustrated as circles. The measurements from system A are slightly ahead and the time shift is shown as Δt.

A **synchronization of** involved **clocks** is also not necessary a sole calculation of a constant offset as the clocks may slightly change for instance due to embedded synchronizations with standard time server over the internet. We synchronize the clocks of various computers at the operating system (OS) level leveraging the commonly used NTP protocol.[24] This is a pragmatic solution when multiple computers are located in the same local network and can be connected via Ethernet. The procedure allows the clocks of multiple machines to converge to a minimum time difference below 1 ms which can be maintained. After convergence, we have a shared clock and can directly compare measurements by their individual time stamps.

9.1.7. Extrapolation Accuracy

Now it is possible to temporally calibrate systems such as robot and OTS and we are able to maintain the synchronization. With this, we study the different upsampling strategies comparing the methods for EuLUp, LUp, and SLUp in an **accuracy** assessment and testing the **computational efficiency** also for DLUp and ScLUp. To interpolate the translational component for the quaternion and rotation-only methods, we perform a separate linear upsampling on the translation component in $\mathbb{R}^3$. The translation component is also linearly upsampled in the dual methods, but they provide a more efficient way for upsampling. Thus, we study them in a consecutive experiment targeting the effect of their advantageous computation complexity in practice.

All methods are tested with poses provided from three different optical tracking systems which are pre-calibrated. We use a commercially available OTS (Polaris, Northern Digital Inc., Waterloo, Canada) which provides poses at 60 Hz together with two prototypes of the system described in chapter 7.[25] One of them is using a stereo setup with two 2 MPix cameras (SMARTEK

[24] Cf. Mills [286].
[25] Cf. Busam et al. [49].

Vision, Croatia) that acquire images and pose measurements at 15 Hz. The other one uses two VGA cameras (SMARTEK Vision, Croatia) and runs at 30 Hz. The 2D image processing pipeline and the 6D pose tracking algorithm is implemented in the FRAMOS Application Framework (FRAMOS Imaging Systems, Germany). All trackers run on the same computation platform with an Intel Core i7-4770K at 3.5 GHz. For reliable ground truth pose estimates, we utilize a second machine controlling an industrial robotic manipulator (LWR4+, KUKA Gmbh, Augsburg, Germany) and its forward kinematics. This provides a baseline with a precision of 0.05 mm and a sampling rate of 1 kHz.[26] Similarly to the approach in chapter 8.1.2, we install a combined marker with markers for all three tracking systems at the end effector of the robot.

During our tests, the **manipulator** is **moved by a human operator** while the robotic arm runs in gravity compensation mode with zero stiffness. The robot and the tracking systems are connected via the ROS[27] framework while the tracking systems transfer the poses via TCP/IP with the OpenIGTLink API.[28] The Fast Research Interface (FRI) connection of the robot is open at a 10 ms rate, in order to sample its position at 100 Hz. All systems are calibrated in time using a ground-truth synchronization clock provided via NTP with a time offset smaller than 1 ms. The output is recorded with the rosbag utility of ROS and then converted to CSV format for post-processing and statistical evaluation. The Cartesian position of the end effector is derived from the sampled joint position through the KDL library solver.[29]
Hand-eye calibration[30] is used for co-calibration between robot and the tracking systems using the algorithm of Tsai et al. [421] in eye-on-base variant. For each hand-eye calibration, we use 20 pose samples and the calibration pipeline implemented in the ViSP library.[31]
Additional to the current pose estimates, the different upsampling strategies are implemented to send poses in real-time for future displacements at time stamps with (+5, +20, +100, +500 ms). All upsamplers are provided only with two poses for the current and the previous measurement from the individual tracker and constantly provide pose extrapolations. The underlying poses used for the extrapolation accuracy experiment only change when a new pose is provided by the tracking system.

After recording the pose streams of the tracking systems and the extrapolations, an **evaluation of** the data is performed. The results are summarized in Fig. 9.4 for the **translation component and** Fig. 9.5 for the **rotations**. An accuracy evaluation for the different native pose streams is shown together with the tested extrapolations. A first analysis reveals that the pose predictions for the future estimates with +5, +20, +100 ms stay within the range of the robotic ground truth. Further looking into the 100 ms estimates shows that the tracker with 15 Hz performs best with an accuracy of $128 \pm 5.7 \mu m$ and $0.5 \pm 0.3°$ using SLUp. This improves the accuracy compared to the 60 Hz OTS by a factor of two although it calculates the poses only one fourth of the time. Compared to the commercial optical tracking system, we believe that its improved accuracy is a result of the sub-pixel precise algorithmic pipeline[32] which cannot fully unfold its potential with the low resolution VGA cameras due to the comparably low pixel count. This still holds true even though the average calibration error in pixel units

[26] Cf. KUKA GmbH [232].
[27] Cf. Quigley et al. [338].
[28] Cf. Tokuda et al. [413] and compare with chapter 7.6.1.
[29] Cf. Smits, Bruyninckx, and Aertbeliën [384].
[30] Cf. chapter 7.7.2.
[31] Cf. Marchand, Spindler, and Chaumette [275].
[32] Cf. Busam et al. [49].

was smaller for the VGA camera system: The 15 Hz tracker was calibrated with an average error of 0.49 pixels while the 30 Hz tracker had only an average error of 0.23 pixels. A similar relative pattern for the tracking systems can be seen for extrapolations with +500 ms where the major error amplitude is due to the movement within this time span. While the median error of the 30 Hz tracker is smaller than the others in this case, its extrapolation precision is slightly worse than its 15 Hz pendant. The 15 Hz tracker provides the best approximation here with SLUp which additionally provides a physically interpretable constant angular velocity by design.

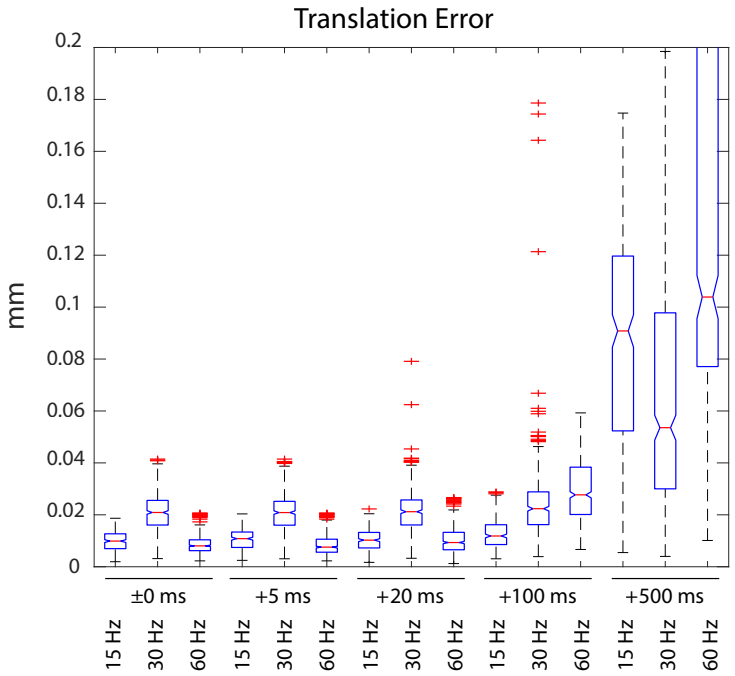


Fig. 9.4. **Translation error of pose estimates for different future extrapolations.** Three tracking systems that run at 15 Hz, 30 Hz, and 60 Hz provide poses (±0 ms) that are used to extrapolate future translations after +5, +20, +100, +500 ms. The box plot shows their median translation error in millimeters (red) together with their interquartile range (blue) for the native measurements (left) and the different linear extrapolation with an increasing extrapolation step.

9.1.8. Efficiency Evaluation

We further empirically test the computational efficiency of all proposed upsampling methods by implementing the upsamplers in C++ using Eigen[33] with its quaternion class which we extend to be able to use all necessary dual quaternion methods. For both interpolation and extrapolation the number of FLOPS is the same. Thus, we fix two poses and test all methods for interpolation between them. We report the **average pose frequency** of 10'000 runs in Table 9.1 which is calculated with the same machine we used for the accuracy tests. The result of our optical tracking algorithm from chapter 7.3 can be provided in dual quaternion form, but we test the rotation-only methods by splitting the information into translation and rotation component first. The split methods EuLUp, LUp, and SLUp that perform separately translation

[33] Cf. Guennebaud, Jacob, et al. [154].

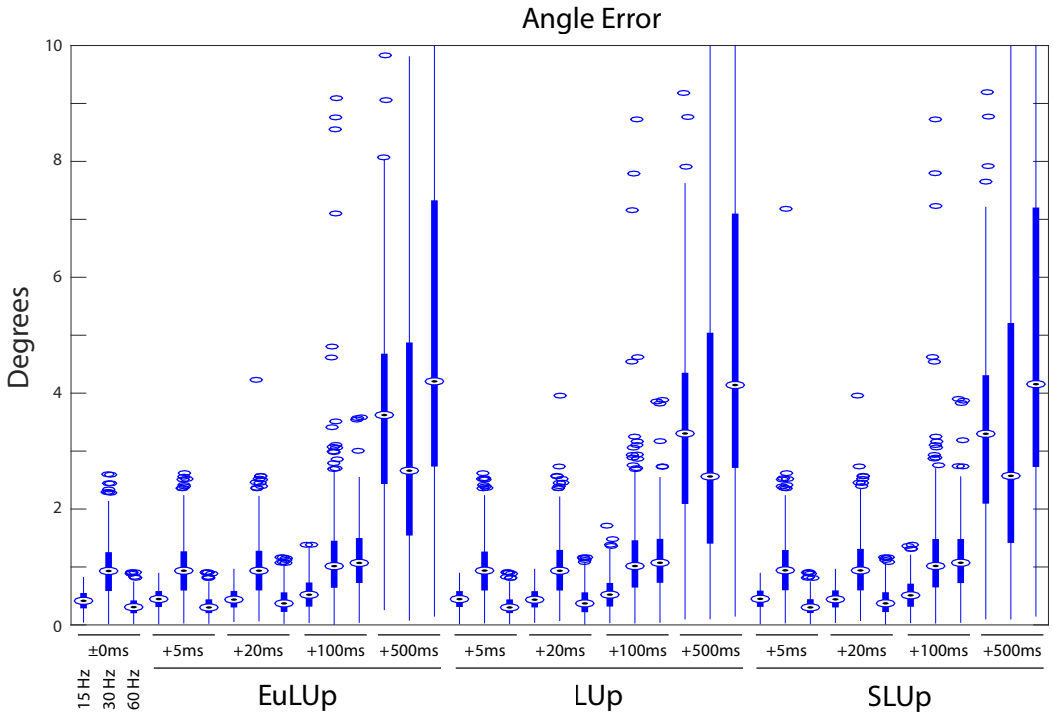


Fig. 9.5. **Rotational error of pose estimates for different future extrapolations.** Three tracking systems that run at 15 Hz, 30 Hz, and 60 Hz provide poses (±0 ms) that are used to extrapolate future rotations after +5, +20, +100, +500 ms. The box plot shows their median rotation error in degrees (dotted blue circle) together with their interquartile range (bold blue) for the native measurements (left) and different extrapolations strategies using the upsampling methods EuLUp, LUp, and SLUp with an increasing extrapolation step each.

and rotation upsampling are slower than the joint upsamplers. The dual quaternion methods DLUp and ScLUp jointly estimate translation and rotation along the Riemannian manifold $\mathbb{DH}_1$ which significantly improves computational efficiency. While we saw that the physical motivation behind SLUp provides reliable extrapolation results, it performs slowest amongst the tested methods. However, its dual counterpart ScLUp can provide accurate interpolations and extrapolations also at high frequencies paving the way to seamless integration with IMUs.

EuLUp	LUp	SLUp	DLUp	ScLUp
469 Hz	306 Hz	236 Hz	4.55 kHz	2.53 kHz

Tab. 9.1. **Efficiency evaluation for different upsampling strategies.** We compare the proposed upsamplers regarding their runtime. The results report the average pose frequency of the methods for 10'000 runs. The computed time for the rotation-only methods EuLUp, LUp, and SLUp involves also the split into rotation and translation and a separate linear interpolation for the translation.

To conclude this part, we observe that pose accuracy can be more important than pose measurement rate for optical trackers in scenarios with the speed of natural hand motion where the necessary missing poses can be upsampled reliably and efficiently. Accurate trackers can further provide useful pose estimates also for future displacements. With the experiments, one may consider the decision for a fast tracking system under these circumstances differently. Hence an accurate though slower system can potentially provide both speed and accuracy for

free if the considered movement is not on a significant other scale than the tracking frequency. **Quaternionic upsampling** can reliably predict human hand motion with a precision of 0.1 mm and 0.5° even up to 100 ms in the future. The discussed upsampling approach for 6D pose interpolation and extrapolation only requires one linear sampling parameter. While the most efficient solution leverages dual quaternion displacement parametrization, the method is not bound to it and can be used for any tracking system to improve its pose frequency or help to fuse it in time with other sensors.

A possible extension of the method could address the smoothness of the estimated curve. While the upsampler currently provides constant velocity between pose measurements, the angular velocity jumps at the interpolation points. Considering multiple past pose estimations in a series could help to provide a smoother and physically plausible solution with the drawback to include older pose information. One direction could be an investigation on how to conserve mathematically smoothness constraints or physical entities such as angular momentum, and the pose history can also be considered to reduce the pose estimation noise and improve the robustness of the optical tracking pipeline. In the subsequent sections, we look in this direction and consider pose sequences with noise and formulate a regularization framework with local control.

9.2. Pose Denoising

Noise is unavoidable in real applications and optical tracking systems suffer from different amounts of inaccuracies. A temporal stream of poses can help to minimize the pose error and improve vision applications and 3D camera systems. In the following sections, we investigate a novel pose **filter based on robust local regression** that **utilizes the Riemannian structure of the (dual) quaternion pose space**. We use differential geometry operators to formulate a principal component regression on the locally linearized pose space of rigid body displacements. With the concepts of chapter 6.2.6, we numerically process a temporally connected set of pose estimates with a Lie algebra and the exponential and logarithm maps such that the pose trajectory is smoothed with local control. In contrast to other filter methods, we directly treat displacements on the 6-dimensional quadric defined by $\mathbb{DH}_1$ in the real projective space $\mathbb{RP}^7$. Exploiting the pose space structure, we formulate a set of different filters including an iterative outlier-aware reweighting scheme. The theoretical contribution is supported by an experimental evaluation on both synthetic and real pose data which proves the practical relevance of the system for camera pose filtering and tracking trajectory denoising.

Accurate pose measurements are the fundamental backbone of many 3D computer vision pipelines and correct estimation of rotation and translation for rigid body motion is crucial for tasks such as camera localization, 3D reconstruction or robotic applications. Every pose estimation algorithm provides displacements with some form of error depending on various factors and outlier-free, precise pose estimates constitute an important aspect for the choice of a system independent of the sensor or modality used for data acquisition. A tracking stream naturally provides an **ordered set of temporally aligned poses** that can be processed sequentially. Optical tracking systems such as the ones discussed in chapters 7.3, 8.1.2, and 8.2.4

all provide discrete pose trajectories independent of whether they utilize markers or measure without object modifications or whether they view from outside-in or track the camera in an egocentric inside-out setting and solve a SLAM problem. We formulate a motion smoothing framework that synthesizes a novel camera trajectory based on the calculated per frame displacements and improves the pose estimations to better match the underlying motion.

We motivate our filter pipeline with a moving window of local linear regressions exemplified with a 2D signal. Suppose we consider a 2D signal of measurements as in Fig. 9.6. A robust line fit can help here to correct a noisy signal point by projection onto a common regression line that can be estimated for example with an outlier-aware robust linear least squares fit. We can apply the same regression idea on a temporal sequence of the signal as illustrated in Fig. 9.7 where we choose a temporally adjacent subset of current noisy measurements as local support for a robust linear regression and correct one measurement in the set before moving the local window in time. If the process is iterated by moving the local window over the measurement sequence, we receive a denoised and corrected trajectory where the local control is guaranteed by the window size.[34] In the following, we establish a pipeline to linearize the pose space locally and perform a similar smoothing where we denoise the pose measurements on trajectories in $\mathbb{DH}_1$. Before we investigate this new **manifold denoiser** and model a set of variants, we take a look at some related literature.

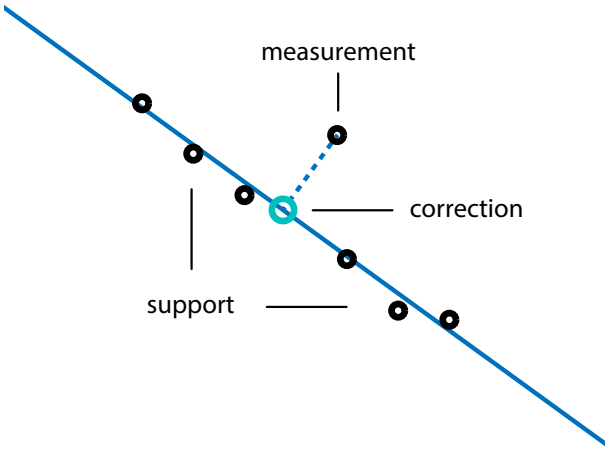


Fig. 9.6. **Measurement correction with linear regression for 2D signal.** One measurement of a 2D signal shown here with black circles is corrected via projection (dashed blue line) onto a robustly detected linear regression line (blue) that is calculated with a robust linear fit of all support measurements including the one that is to be corrected. The corrected measurement is drawn in turquoise.

9.2.1. Related Pose Regression Models

We use **dual quaternions** to represent spatial displacements and denoise the poses along trajectories in $\mathbb{DH}_1$. An essential tool for various 3D computer vision tasks is accurate and reliable pose data that can be retrieved with regression models and interpolation.[35] Real-time blending with unit dual quaternions can be achieved in computer graphics using approximations based

[34] Cf. Fox [120].
[35] Cf. Parent [320].

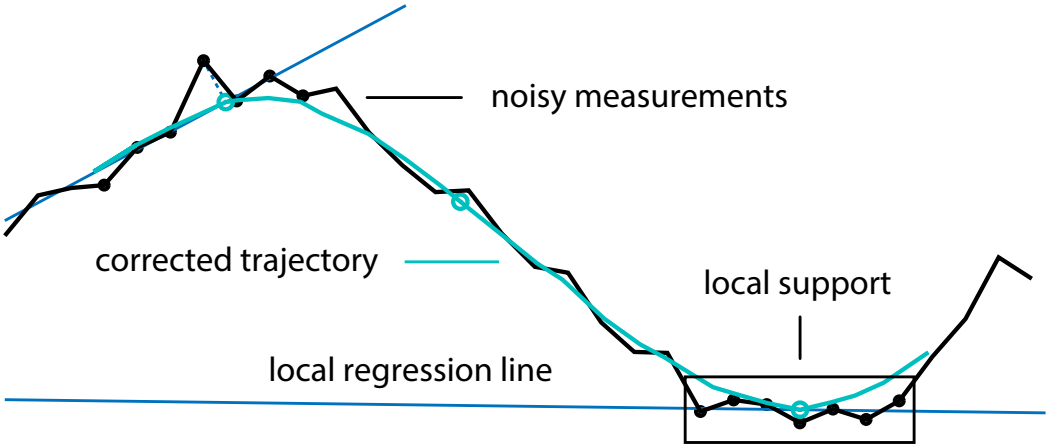


Fig. 9.7. **Denoising with local linear regression for temporal 2D signal.** A temporal sequence of a 2D signal is shown in (black)from left to right. A local support window of seven noisy measurements (black box lower right and circles upper left) around the current signal point is chosen to for robust linear regression (blue). The current measurement is then corrected by projection (dashed blue line) onto the current line and the support window is shifted to include the next measurement before the process is repeated. All corrected points (exemplified with three turquoise circles) provide the corrected measurement trajectory (turquoise trajectory).

on the L2-norm in the embedding space $\mathbb{R}^8$, but the manifold of dual quaternions is not an Euclidean space. In a similar fashion, Torsello et al. [417] use an optimization framework leveraging the Riemannian metric with diffusion principles for multiview image registration. Likewise, we use the Riemannian structure of $\mathbb{DH}_1$ together with differential geometry operators to apply Euclidean smoothing approaches on the locally linearized tangent space.

Previous **denoising methods** study the problem intensively in joint rotation and translation space $\mathbb{H}_1 \times \mathbb{R}^3$.[36] Gaussian smoothing is proposed for rotation parametrization with non-dual quaternions[37] and Srivatsan et al. [390] design a linear Kalman filter to act on dual quaternion space by modeling the noise for the displacement estimates. An Extended Kalman Filter (EKF) for denoising on SE(3) is proposed by Filipe et al. [114]. Other scholars apply pose denoisers for camera video stabilization[38] and utilize pose smoothing in the robotics domain.[39] Pose regression is used with a Kalman filter on a quaternion sequence in a virtual reality application by LaViola [237] who tracks human hands and heads, and a fusion approach with inertial and magnetic sensors is proposed with a filtering technique in the work of Yun et al. [468].

9.2.2. Local Regression Geodesics

It is crucial to consider the pose parametrization landscape in order to achieve a pipeline that is able to perform a meaningful local linear regression as pose space is non-Euclidean. 6D poses can be parametrized in different ways as discussed in chapter 6.2. Homogeneous matrices or quaternion and vector parametrizations usually treat rotation and translation separately. We

[36] Cf. Farenzena, Bartoli, and Mezouar [104] as well as Jia and Evans [195].
[37] Cf. Ng et al. [305].
[38] Cf. Jia and Evans [195].
[39] Cf. Farenzena, Bartoli, and Mezouar [104].

utilize again a joint dual quaternion representation and leverage the shape of the pose space to constrain our filter locally. Aside of the computational efficiency of the dual quaternion representation as shown in part 9.1, we also gain numerical stability for the regressions by choosing re-normalization over re-orthogonalization of matrices.[40] We can additionally make use of the differential geometric aspects discussed in chapter 6.2.6 and leverage the exponential and logarithm operators to linearize the space and to design a smoothing pipeline for pose sequences that is robust to measurement noise.

The noise models of an optical tracking system or a given pose estimation algorithm can be intricate. It depends on various factors such as calibration steps, jitter and the blur from a rolling shutter sensor, but can also be influenced by other sources such as the velocity of the camera motion or the scene illumination. The resulting noise with all these interdependent error sources is highly non-linear and very difficult to model adequatly. It is possible to chose an input dependent filtering approach to reduce statistical noise with a Kalman filter. However, setting all parameters is not always trivial in applications. For this reason, we design a non-parametric regularization method without specific models for the individual error sources and leverage a **manifold-aware regularization from dual quaternion pose space** where screw linear motions are the underlying displacements. We constitute a linear regression where we first select a local set of poses adjacent in time. A simple linear regression as motivated in Fig. 9.6 with the pose parameters is not possible as the pose space itself is not Euclidean. As a result, we perform the regression with geodesics instead of lines in the embedding space. Since we only consider a local neighbourhood of poses, we can linearize the pose space around the centre point in a temporal pose window by establishing the tangent space to the pose manifold there as illustrated in Fig. 9.8. All relevant points from the local window can then be mapped with the logarithm map into the Euclidean tangent space where a principal component regression can be performed seamlessly. The centre point can then be corrected via a projection onto the line and remapping onto the pose manifold with the exponential map brings a corrected pose. An example is depicted in Fig. 9.9 and the iteration of the process can be used to denoise the entire pose sequence. Note that the choice for the centre point is arbitrary, but produces a filter which is equally considering past and future pose measurements. If the task is also to minimize the pose stream delay, only taking previous poses works in the same vein.

We further model a principal component analysis (**PCA**) for the linear regression on tangent spaces and formulate the filter for rotations using quaternions and for joint pose parametrizations with dual quaternions. An experimental investigation indicates that **weighted PCA** on the Riemannian pose manifold provides adequate denoising for accurate pose estimation systems while iteratively reweighted least squares (**IRLS**) smoothing on $\mathbb{DH}_1$ improves fidelity in presence of pose outliers. The intrinsic screw linear pose regularization through the dual quaternion quadric smoothing further improves the signal to noise ratio for higher levels of noise.

[40] Cf. Belta and Kumar [19].

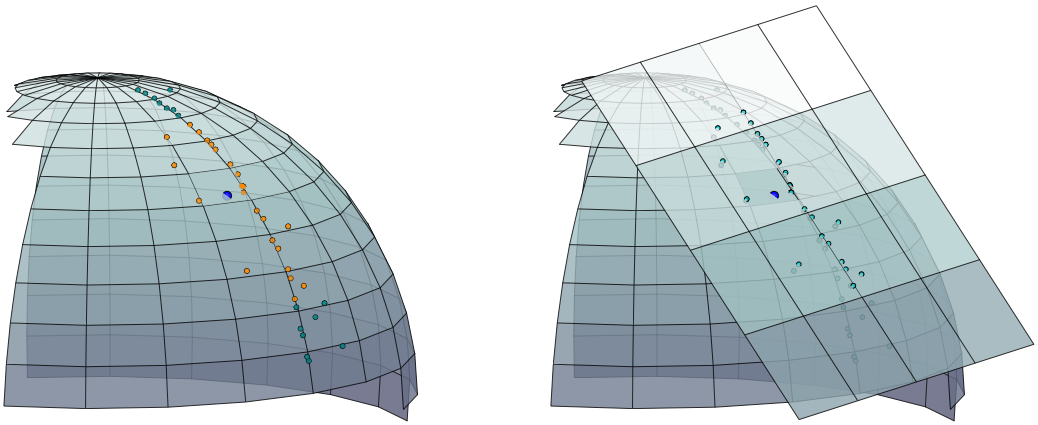


Fig. 9.8. **Local linearization of moving measurement window.** The left image shows the window selection of a subset (orange) of pose estimations (turquoise) around a centre point (blue) on the pose manifold. The right image illustrates the local linearization around the centre point by constructing the tangent space to the Riemannian manifold. Poses are mapped onto the tangent space with the logarithm map.

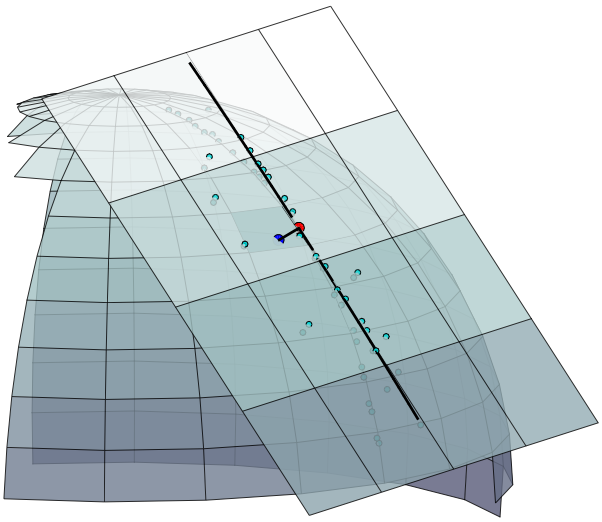


Fig. 9.9. **Robust geodesic regression on tangent space.** The pose measurements on the tangent space are used for an outlier-aware robust linear regression. The regression line is drawn in black and represents a geodesic on the manifold. The centre point holding the tangent space is then projected onto the line (red) and mapped back on the Riemannian pose manifold with the exponential map. It is the final denoised pose. The process is iterated with a moving window.

9.2.3. Robust Quaternion Pose Filters

The differential operator for the logarithm map from chapter 6.2.6 helps to linearize the pose manifold locally and apply principal component regression in the Euclidean tangent space for both the rotation group $\mathrm{SO}(3)$ and the rigid body motions $\mathrm{SE}(3)$. The centre point of the local window can be corrected in the tangent space and be seamlessly mapped back with the exponential map utilizing parallel transport and the Lie algebra to the Lie group. The detour through the tangent space allows to represent geodesics from the manifold with lines in the linearized space. In order to apply robust motion stabilization, we perform an iterative application of the process on a moving window of poses.

The following section details the design of our robust denoiser for (dual) quaternions. We filter the temporal discrete pose signal in order to stabilize the estimated motion via **geodesic regression** leveraging lines in tangent space. We constitute (weighted) PCA and an iterative reweighting least squares optimization scheme (IRLS).
The sequential pose estimations with temporal dependency define a trajectory in pose space. The path in $\mathbb{DH}_1$ is in general noisy, non-linear and includes outlier measurements. Without further assumptions on the motion or physical constraints given by the application, the velocity can vary and the sampling rate may not be regular. These factors generally impede the design of noise models and complicate parameter tuning of filters. Our algorithm for path smoothing instead is parameter-free and regresses pose sequences robustly which make it a flexible tool of practical relevance.

In our **local tangent space window**, we consider the trajectory as linear. Our task is to estimate the linear dependency between the data points $\mathbf{X}$ and the responses $\mathbf{y}$ for the regression such that

$$\mathbf{y} = \mathbf{X}\beta + \epsilon \tag{9.31}$$

is fulfilled with the regression coefficient β and the error term ϵ where

$$\mathbb{E}[\epsilon \mid \mathbf{X}] = 0, \tag{9.32}$$

$$\mathrm{Cov}[\epsilon \mid \mathbf{X}] = \mathbf{C} \tag{9.33}$$

with the covariance matrix $\mathbf{C}$. One way to write an optimization function for this problem is using **generalized least squares** minimization as

$$\beta^* = \arg\min_{\beta} (\mathbf{y} - \mathbf{X}\beta)^{\mathrm{T}} \mathbf{C}^{-1} (\mathbf{y} - \mathbf{X}\beta). \tag{9.34}$$

The solution to this quadratic problem can be expressed analytically as

$$\beta^* = (\mathbf{X}^{\mathrm{T}} \mathbf{C}^{-1} \mathbf{X})^{-1} \mathbf{X}^{\mathrm{T}} \mathbf{C}^{-1} \mathbf{y}. \tag{9.35}$$

If the error terms are all uncorrelated, the covariance matrix is diagonal and we solve a weighted least squares problem. The exact covariances are rarely known in practice and we need to estimate the covariance matrix $\mathbf{C}$ usually.

An uneven temporal sampling of the data can lead to a different consideration for the criteria of a suitable linear regression which, for instance, maximizes the variance with a **principle component analysis** (**PCA**). The principal component of the local window $\mathbf{X}$ would then be a suitable solution and we can consider the singular value decomposition (SVD)

$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}} \tag{9.36}$$

of $\mathbf{X}$ with the unitary matrix $\mathbf{U}$, the diagonal matrix $\mathbf{S}$ with the singular values, and the right singular vectors $\mathbf{V}$. The column entries of $\mathbf{V}$ define an orthonormal basis of eigenvectors for $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ and the i-th principle component is given by the i-th entry under the transformation $\mathbf{XV}$. A dimensionality reduction can be achieved by keeping only the first k principle components using the first k columns of $\mathbf{V}$. The principle components are then given by

$$\mathbf{X}\mathbf{V}_k \quad \text{with} \quad \mathbf{V}_k = (\mathbf{v}_1 \cdots \mathbf{v}_k). \tag{9.37}$$

We smooth the pose sequence by taking the central point $\mathbf{c}$ from the pose window and project the other points onto the principal component line given by the first principle component as illustrated in Fig. 9.9 before we move the pose window further over the measurements. While this approach assumes the poses to locally follow a Gaussian distribution, the global error can be different and is not explicitly modeled here. PCA can be done on Euclidean spaces, but cannot be directly applied to Riemannian manifolds such as $\mathbb{DH}_1$ for dual quaternions which are not Euclidean. Even the regression of a great arc on the hypersphere $\mathbb{H}_1$ has ambiguities.[41] In the local moving window, the central point $\mathbf{c}$ is known and can be used to hold the tangent space $T_{\mathbf{c}}\mathbb{DH}_1$ which is populated with the local neighbourhood of dual quaternion measurements after the mapping by the logarithm operator. The Riemannian manifold structure of the pose space allows for this as it locally behaves like an Euclidean space, where we can apply the aforementioned PC-regression or a geodesic least squares fitting. The pose path is then smoothed by projection of $\mathbf{c}$ onto the regression line in $T_{\mathbf{c}}\mathbb{DH}_1$ before mapping the corrected point back to the Riemannian manifold of dual quaternions with the exponential operator. In our further investigations, we refer to this regression as plain PCA filter.

The poses which are in closer proximity to the centre $\mathbf{c}$ of the window are more relevant for the current linear approximation since the assumption quality for the local linearity of the motion decreases with the distance to $\mathbf{c}$. Weighting the pose measurements according to their distance with a Gaussian as

$$w_i^0 = \exp\Big(-\frac{1}{2}(\mathbf{x}_i - \mathbf{c})^{\mathrm{T}}\mathbf{D}(\mathbf{x}_i - \mathbf{c})\Big) \tag{9.38}$$

using the Mahalanobis distance with the positive semi-definite matrix $\mathbf{D}$ then models a decaying influence of points further from the centre. We reference to this together with the previous ideas as **weighted principal component analysis** (**wPCA**).

The developed concepts so far can still fail in the presence of **pose outliers**. To tackle also this case, we leverage an iterative process that updates the weights of the measurements for the next iteration based on the reciprocal residuals given in the current iteration step. This process

[41] Cf. Buss and Fillmore [54].

is known as **iteratively reweighted least squares** (**IRLS**) fitting. We update our weights with

$$\mathbf{w}_{i+1} = 1/\max\Big(\delta, \frac{1}{K}\sum_{k=1}^{K}|\mathbf{r}_i^k|\Big). \tag{9.39}$$

where the small number $\delta > 0$ prevents division singularities and the residual $\mathbf{r}_i^k$ is the fitting error of pose parameter k in step i. The number $K \in \{3, 4, 8\}$ defines the embedding space dimensionality of our pose parametrization. It is $K = 3$ to denoise translation vectors, $K = 4$ is used for rotations described by quaternions and we utilize $K = 8$ for displacements with dual quaternions. The algorithm for the **local regression with iterative weight updates** is summarized in Algorithm 9.1. In the special case of $N = 1$ with weights according to equation (9.38) and no updates, it defines wPCA. Further initializing all weights with the same value gives the PCA method. We compare these three approaches for principal component regression in Fig. 9.10 schematically.

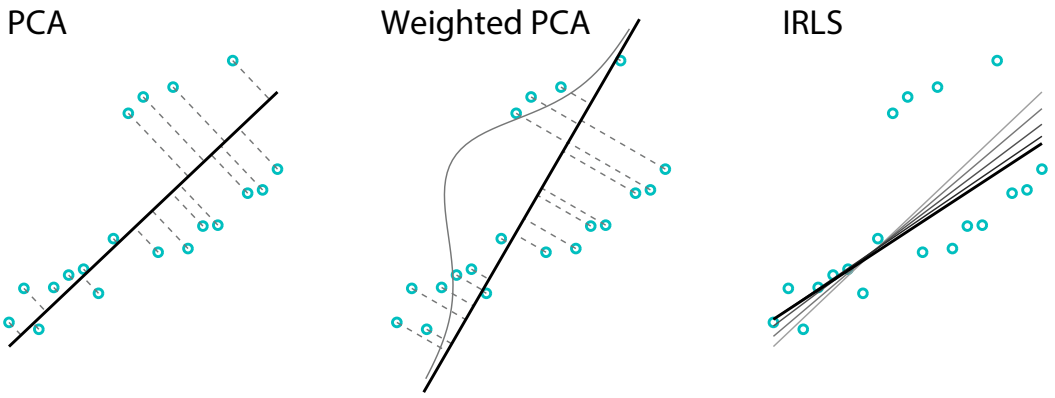


Fig. 9.10. **Linear regression methods for tangent space line fitting.** The three proposed methods for principal component regression in the tangent space are illustrated. The selected window of pose measurements is depicted in turquoise. The resulting linear fit is shown in black. On the left, we see a classical principal component analysis on the selected pose measurements where the weight for each point is equal. The centre part shows a weighted PCA version where the weight decays with a Gaussian function according to the distance from the centre point. On the right, we show the intermediate results of an iterative reweighted least squares (IRLS) fit with $N = 5$ iterations for the sample set of poses with increasing opacity for the intermediate to final calculated lines.

Finally, Algorithm 9.2 details the overview of the **pose denoising** process which is independent of the pose parametrization used. All previous concepts can be leveraged for the three cases of translation, rotation and joint pose denoising by replacing the differential operators for non-dual quaternion space $\mathbb{H}_1$ and by using the identity maps for the Euclidean case of translations with vectors in $\mathbb{R}^3$. We leverage this aspect in our consecutive experiments by comparing joint representations with separate treatment of translation and rotation.

9.2.4. Synthetic Data Validation

We compare the proposed variants of our robust quaternion pose filters with each other and against a linear Kalman filter on two different datasets. The first experiment investigates

Algorithm 9.1. Iterative Reweighted Least Squares for Weighted PCA Correction

Input parameters:

- Local pose window: $\mathbf{X} = (\mathbf{X}_i)$ with $\mathbf{X}_i \in \mathbb{R}^K$
- Prior weights: $\mathbf{w}^0 = (1, \ldots, 1)^{\mathrm{T}}$
- Number of iterations loops: $N \in \mathbb{N}$
- Small robustifier: $\delta = 10^{-4}$

Computation steps:

Initialize weights: $\mathbf{w} = \mathbf{w}^0$

for $j = 1$ **to** N **do**

  // Estimate current linear regression line

  $\mathbf{l} \leftarrow \text{weighted\_pca}(\mathbf{X}, \mathbf{w})$

  // Orthogonal projection of poses $\mathbf{X}$ onto line $\mathbf{l}$

  $\mathbf{X}^{\downarrow} = \text{orthog\_proj}(\mathbf{X}, \mathbf{l})$

  Calculate residuals: $\mathbf{r} = \mathbf{X} - \mathbf{X}^{\downarrow}$

  // Update weights with equation (9.39)

  $\mathbf{w} = 1/\max\left(\delta, \frac{1}{K}\|\mathbf{r}\|_1\right)$

  // Dampen weight estimates

  $\mathbf{w} = \mathbf{w} \odot \mathbf{w}^0 / \left\|\mathbf{w} \odot \mathbf{w}^0\right\|$

Select projected centre point: $\mathbf{X}_c = \mathbf{X}_c^{\downarrow}$

Output:

- Corrected centre point: $\mathbf{X}_c$

Algorithm 9.2. Local PC-Regression Geodesics based Pose Denoiser

Input parameters:

- Kernel size for local window: $2n + 1$ with $n \in \mathbb{N}$
- Pose sequence: $\mathbf{X} = (\mathbf{X}_i)$ with $i \in \{1, \ldots, m\}$ as part of a larger sequence such that we can always select a local window

Computation steps:

for $j = 1$ **to** m **do**

  // Select local pose window

  $\hat{\mathbf{X}} = (\mathbf{X}_i)$ with $i \in \{j - n, \ldots, j + n\}$

  // Map window into tangent space at $\mathbf{X}_j$

  $\hat{\mathbf{X}}^{\perp} = \log_{\mathbf{X}_j}\left(\hat{\mathbf{X}}\right)$

  // Perform IRLS for wPCA correction on tangent space

  $\mathbf{X}_c^{\perp} = \text{irls\_wpca}\left(\hat{\mathbf{X}}^{\perp}\right)$ (Algorithm 9.1)

  // Map corrected pose back onto Riemannian manifold

  $\mathbf{X}_c = \exp_{\mathbf{X}_j}\left(\mathbf{X}_c^{\perp}\right)$

  Update corrected pose sequence: $\bar{\mathbf{X}}_j = \mathbf{X}_c$

Output:

- Corrected pose sequence: $\bar{\mathbf{X}} = \left(\bar{\mathbf{X}}_i\right)$ with $i \in \{1, \ldots, m\}$

the ability of the denoiser to recover synthetically generated poses in the presence of outlier measurements and the second test applies the denoising methods on real data where we leverage the hand motion dataset from Busam et al. [50] which we described in chapter 9.1.

To evaluate the manifold denoiser we **simulate a motion trajectory** to which we synthetically add noise in order to **evaluate accuracy and robustness** by comparing the regressed displacements with the initial ground truth rotation and translation.
We randomly select five rotations $\mathbf{R}_i$, $i \in \{1,\dots,5\}$ with five rotation axes $\mathbf{v}_i \in \mathbb{R}^3$ and according rotation angles $\theta_i \in [0, 2\pi]$. A selection of five random translations $\mathbf{t}_i \in [0,1]^3$ is also chosen. We interpolate all three components, axes, angles, and translation using cubic splines and parametrize the resulting pose sequence components in quaternion $\mathbb{H}_1$, dual quaternion $\mathbb{DH}_1$ as well as in Euclidean space $\mathbb{R}^3$ for the translation. Pose filtering is performed on the joint space $\mathbb{DH}_1$ and applied twice by independent denoising on the rotation and translation component in $\mathbb{H}_1 \times \mathbb{R}^3$. For the analysis of the regression, artificial noise is added to the ground truth pose trajectory directly to angles, axes and translation vectors by sampling from the uniform distribution in $[-\sigma, \sigma]$ with $\sigma = 0.02$. Outliers are randomly added to 5% of the simulated pose measurements with a significantly higher noise using $\sigma = 0.2$.
The resulting noisy pose sequences with outliers are tested with the methods using PCA, wPCA, IRLS on quaternion and Euclidean space $\mathbb{H}_1 \times \mathbb{R}^3$ and with dual PCA, dual wPCA, dual IRLS on $\mathbb{DH}_1$. The results are additionally compared with the denoising of a linear Kalman filter. In all tests, we select a window size of 19 poses and leverage the Kalman filter from Welch et al. [448] as implemented in MATLAB[42] with covariances for the process and measurement noise of rotations in $[0.5, 2]$ and translations in $[0.2, 1]$.

An example pose series with the denoised results of dual IRLS is shown in Fig. 9.11 and an overview comparing the resulting accuracy of the proposed denoisers as well as the linear Kalman filter can be found in Fig. 9.12. The Kalman filter values are chosen in a way that the pose signal can be recovered without over-smoothing. In comparison to the other filters, it only considers half of the window size as future measurements are not part of the filter input. It notably suffers more from the severe outliers than our other proposals resulting in a performance decrease compared to them. The local PCA approaches reduce the error significantly. Compared to this, the Gaussian weights of wPCA slightly improve the mean error with a result of $0.37 \pm 0.55°$ rotation error and 0.010 ± 0.020 for translation. The full potential of the joint **dual quaternion** treatment manifests in the results from the **IRLS** method where the average error **outperforms all other methods** while the improvement for its non-dual counterpart is minimal with only $2.1 \cdot 10^{-3}$. The space of dual quaternions $\mathbb{DH}_1$ helps to reduce the influence of outliers significantly which results in a median improvement of $4.3 \cdot 10^{-3}$ and $0.26°$. From our perspective, this is achieved through the restriction of the local neighbourhood in joint space where the linear tangent space regression puts more constraints on the pose sequence than a separate neighbourhood treatment in $\mathbb{H}_1 \times \mathbb{R}^3$ where the effect of an outlier can be more influential. We noticed that the adjustment of the parameters for the Kalman filter requires time and technical expertise for tuning while the only parameter that needs to be chosen for the proposed methods is the size of the local window which directly reflects both motion speed and pose measurement sampling rate.
The interested reader may be pointed to a video of the synthetic temporal signal where the

[42] Cf. MathWorks [277].

denoising method is best illustrated with motion data which shows the temporal pose stream with added noise together with the effect of denoising.[43]

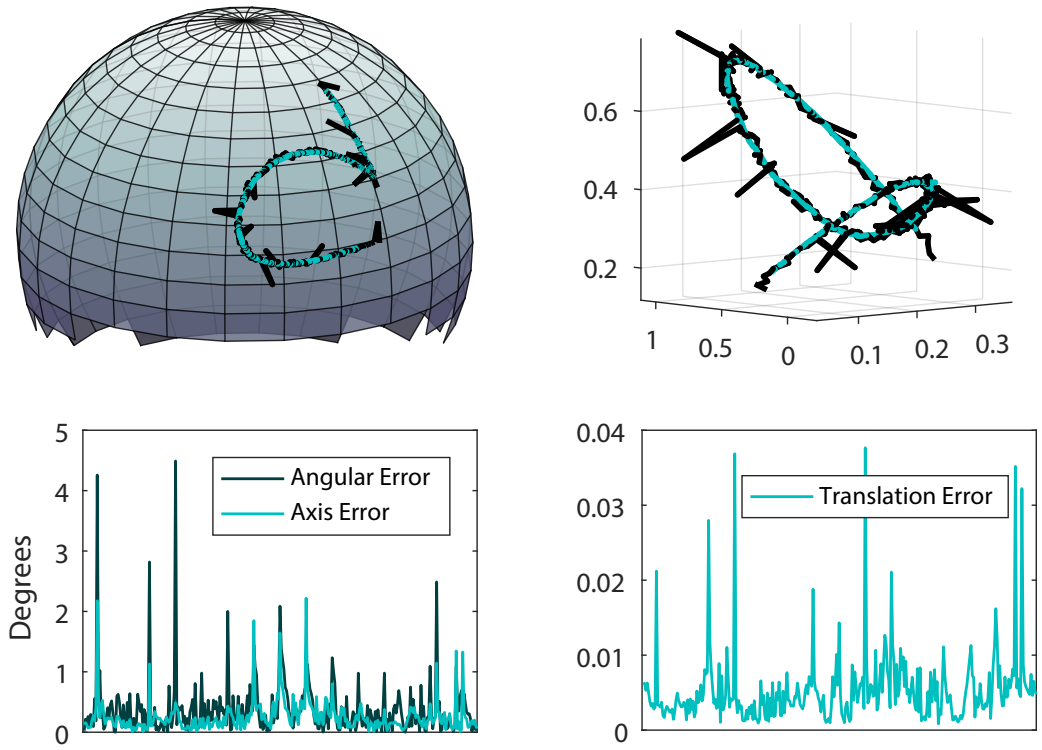


Fig. 9.11. **Dual quaternion pose filter on synthetic data.** A synthetically generated pose sequence (black) is compared to the result of denoising with our dual IRLS method (turquoise). The rotation is analysed on the left where the axis projection onto S^2 is shown (top left) and both angular (dark green) and axis error (turquoise) are shown (bottom left). From the translation trajectory (top right), the noise is is clearly visible together with the spiky outliers while the denoised signal is significantly smoother. The translation error is also visualized (bottom right).

9.2.5. Denoising Poses from Optical Tracker

We further assess the quality of our denoisers on real data to **refine and improve** the **tracking stream** of optical pose measurements. Tracking accuracy plays a crucial role in collaborative robotics, thus we investigate the dataset from chapter 9.1 with ground truth from a medical robotic arm.[44] The dataset comprises a set of human manipulations of a robotic end effector that follows the hand motion with zero stiffness in gravity compensation mode. During the procedure, the end effector is tracked with our optical tracking algorithm as described in chapter 7.3 where we record ground truth poses of the co-calibrated end effector through its forward kinematics. For evaluation, we process the 30 Hz data sequence with the different denoisers and compare against this ground truth. The results are summarized in Fig. 9.13 with the same naming convention as before.

[43] A video can be found online under `http://campar.in.tum.de/Chair/PublicationDetail?pub=busam2017_mvr3d`.
[44] Cf. Busam et al. [50].

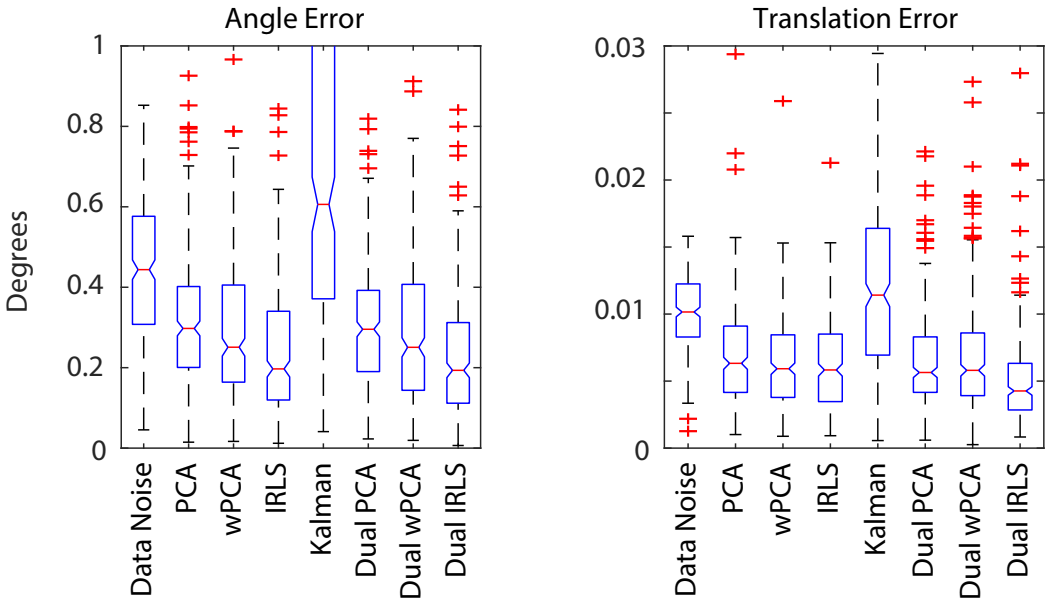


Fig. 9.12. **Accuracy evaluation of different pose denoising methods on synthetic data.** We show box plots for the results of the proposed pose filters in comparison with a linear Kalman filter on synthetically generated noisy pose sequences with outliers. The data noise is added as a reference for validation. The left figure shows the angular error which is given by the residual for the rotation axis in degrees while the right figure illustrates the translation deviation. All proposed pose filters reduce the data noise and the dual IRLS method shows the smallest error. The Kalman filter result is not reducing the data noise in presence of outliers.

The best performance is achieved with wPCA while the difference between dual and non-dual method is marginal. The non-dual results of wPCA with $0.7 \pm 1.4°$ (median 0.5°) and $11.2 \pm 3.9\mu m$ (median $11.4\mu m$) gives the minimal error while our IRLS method performance is mediocre with an accuracy between PCA and wPCA in both the quaternion ($13.0\mu m$, 0.6° median) and dual quaternion case ($14.1\mu m$, 0.6° median). The results achieved with a linear Kalman filter are acceptable and do not significantly change when heuristic parameter fine-tuning is applied. Using dual space denoising let to a significant improvement in our synthetic experiments where outliers were present in the pose measurements. A similar advantage of the joint space filter cannot be observed here as the optical tracking provides already highly accurate poses with limited outliers where the constraint for the signal on the joint space is not required. The higher pose quality also influences the IRLS result as the weights for poses of similar importance are equally reduced. As a result, the non-dual regression methods where rotation and translation are treated separately are preferable over joint methods when reliable pose data is given.

9.2.6. Qualitative Evaluation on RGB-D Data

For a **qualitative comparison of different window sizes**, we use a video acquisition from an RGB-D camera that is moved around an object while KinectFusion[45] is applied to retrieve a

[45] Cf. Newcombe et al. [304].

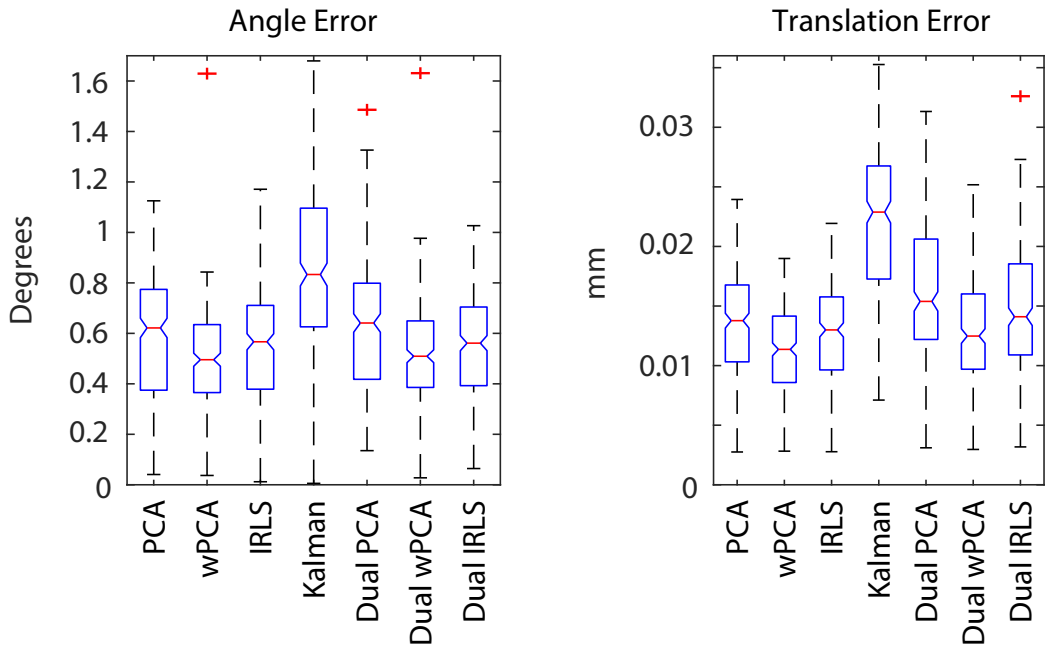


Fig. 9.13. **Accuracy evaluation of different pose denoising methods on real data.** We show box plots for the results of the proposed pose filters in comparison with a linear Kalman filter on the collaborative robot motion data from chapter 9.1. The left figure shows the angular error which is given by the residual for the rotation axis in degrees while the right figure illustrates the translation deviation in millimeters. The best results are achieved with wPCA.

map and the according camera poses. The extraction process for the poses is shown in Fig. 9.14 with some example images and depth maps.

The retrieved displacements are then processed with our pose denoiser and a variable window size w is applied. The results can be seen in Fig. 9.15 where a window size of $w = 15$ frames only affects minor pose jitter, while $w = 30$ already changes the pose sequence more visibly in the areas with more rapid movement and $w = 60$ produces a smooth motion that varies significantly from the initial camera poses. While such a large window size may not be helpful for pose denoising, it can be an improvement for video stabilization and hyperlapse motion shots. The choice of the window parameter thus has to be chosen depending on the video frame rate or the pose estimation frequency if the poses are not directly provided from the video stream. While higher sampling rates allow to resolve better in time another factor is also the motion speed of either camera or object where a slow motion results in the same effect as a high pose frequency. Finally, prior knowledge of the reliability of the provided pose stream can be used to adjust the window parameter such that maximal pose denoising and minimal over-smoothing can be reached.

In summary, the presented pose denoisers use local regression on the pose space to minimize influence of noise and outliers in the pose stream. Using concepts from differential geometry such as the exponential and logarithm maps help to study the problem in the tangent space to the pose manifold where a robust linear regression can be used to determine the principal component and gradually correct the pose sequence either on joint dual quaternion space $\mathbb{DH}_1$ or separately for rotation and translation in $\mathbb{H}_1 \times \mathbb{R}^3$. Our experimental evaluation has shown that the constraints of the **dual space formulation can improve the robustness of**

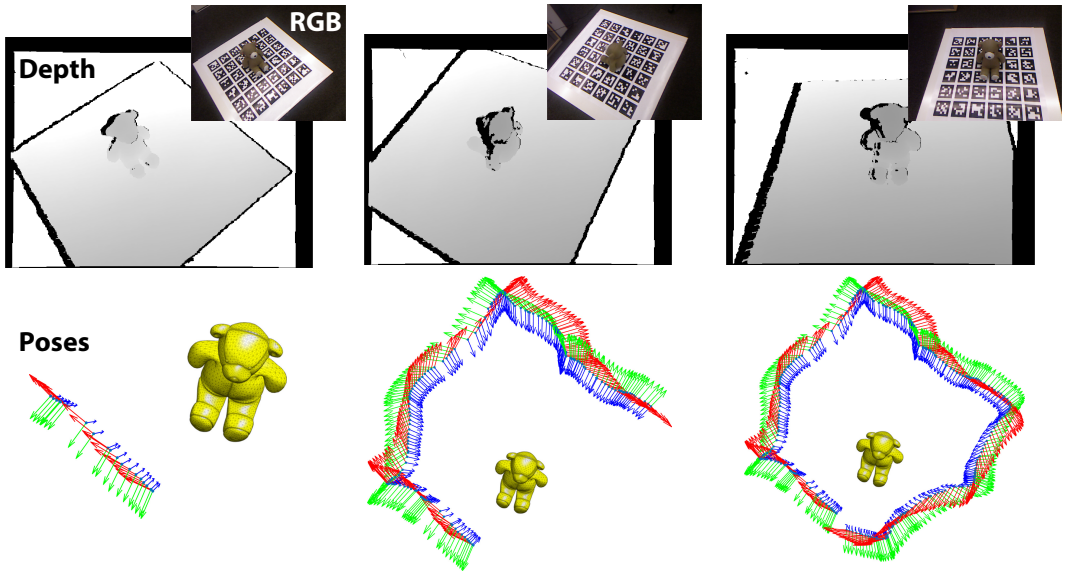


Fig. 9.14. **Camera pose extraction with KinectFusion.** We leverage a video sequence acquired with a sensor that captures both RGB and depth images as shown in the upper row. The depth maps are processed with the KinectFusion algorithm and the RGB images are only shown as reference. The extracted poses are saved as visualized with the coordinate axes (lower row) for the different frames in the video. We visualize also the teddy object as reference. A full acquisition around the object is performed where the right most column shows the last frame and its result.

the denoising. The pose processing further allows for local control where the only parameter that is set is the window size to define the local influence of a pose change. The method can be used for **outlier aware denoising** and **pose trajectory smoothing** without the need of an explicit noise model. Using a local linearization has, however, the downside to require the pose sampling to be adequately dense around the tangent space to ensure a reliable regression. If the system is applied online in a real-time setup where half a window of lag is not acceptable, one can either cut the window into half and use only past points for the regression or leverage extrapolation techniques similar to the concepts from chapter 9.1 to minimize the lag. Formulating the regression as a continuous pose model that uses pose timestamps as additional inputs could therefore also be an interesting future direction that may increase robustness of visual odometry pipelines. A possible extension of the denoiser could include pose uncertainties to weight the local displacements. These could come from feature reprojection residuals in tracking applications such as the feature-based SLAM used in chapter 8.1 or from the mean backprojection error of a 3D localization approach.

An important ultimate outcome of pose denoising is the increase of the pose estimation quality which plays an integral role when different modalities are fused in space. Before we leverage accurate pose estimates for data fusion, we briefly address also pose improvement strategies that follow other ideas in particular in the context of multiple modalities and consistency formulations.

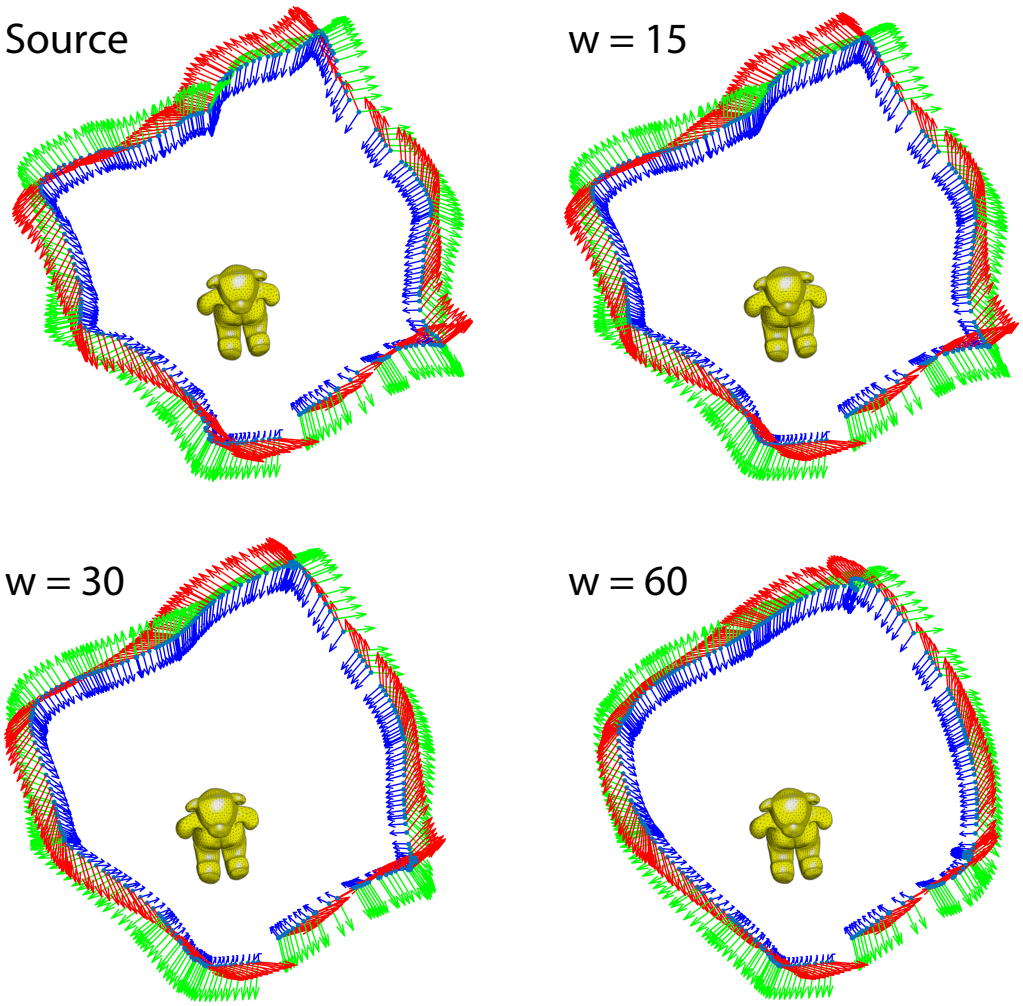


Fig. 9.15. **Effect of different pose window sizes on filtered poses.** The plot shows the result of wPCA on the teddy sequence with the extracted poses as shown in the source case (top left). The window size w affects the poses. An increase from $w = 15$ (top right), $w = 30$ (bottom left) and $w = 60$ (bottom right) gradually applies a smoothing to the resulting pose trajectory.

9.3. Pose Improvements

Spatial sensor fusion requires accurate pose knowledge. The better the position and orientation of various sensors is known, the more reliable becomes the data when their signals are spatially joined. In the past, many scholars proposed ideas to improve upon the initial pose estimation quality with different thoughts. To be able to give a holistic view of the field as a preparation of poses in the presence of multiple input signals, we briefly touch two concepts to enhance pose quality through the formulation of consistency loops that can also help as training signals in data-driven pipelines. The first one focuses on the **intrinsic consistencies** within various predictions from the same input and the other takes into consideration **extrinsic consistencies** where multiple sensor inputs are checked for agreement and discrepancies.

9.3.1. Task Consistency

Task consistency is the underlying concept for **intrinsic** methods where different tasks help each other during learning or inference. This is not limited to pose estimation only but stands as a general concept where, for instance, the knowledge of surface normals can help to estimate depth as the information of the surface structure constraints depth variation and depth information implicitly contains surface normal information through differentiation. Two tasks may also have different outcomes. An object detector may tell there is a sphere while a depth mapper detects similar values which suggest a planar surface. This is inconsistent and at least one of the estimates must be false. In these cases, an improvement of one estimate results in an improvement of the other which can simplify and speed up the learning of these predictions while it can improve the overall accuracy that can be reached. Many scholars have explored cross-task consistencies in the past where 3D correspondences are used by Zhou et al. [486] and multi-view consistency is used to help both pose and shape estimation in the work of Tulsiani et al. [423]. Agreement between depth and flow estimates are used by Zou et al. [488] and the team of Dwibedi et al. [94] leverages temporal consistency cycles. The time component can also be used to design self-supervision losses where pose and depth estimates need to agree in a temporal window.[46] Oftentimes the dependency between tasks is obvious as in the examples above. However, a mutual or single-sided benefit can also arise from non-trivial dependencies. An extensive empirical task-dependency test in this regard is considered by Zamir et al. [473]. Taking the consistency cycles one step further, we can argue that consistency among many tasks improves the individual ones. This can be enforced forming a large number of consistency cycles which significantly improve the results.[47]
While the output of a **prediction** algorithm **forms the consistencies** in all these case, the input can also be considered.

9.3.2. Modality Consistency

Combining multiple input signals can benefit an estimation outcome and can help to overcome drawbacks of individual sensors. In the context of line-of-sight considerations, we had a look in chapter 7.8.2 into the visual-inertial odometry literature where the signal of an IMU is combined with vision data to improve pose estimation and visual dead reckoning scenarios. Amongst classical parametrization, the dual quaternion formulation we consider improves visual-inertial tracking in AR setups.[48] Other input modalities also lead to **extrinsic consistency cycles** as their measurements must agree in the same situation. In the context of depth estimation for instance, a sparse signal arising from a LiDAR sensor can be combined with RGB information to complete a dense depth map.[49] Both signals mutually improve each others single task of estimating depth. The same holds true if a commodity RGB-D sensor is used.[50] Synthesized images of another modality can thereby improve the results on real data[51] and even a feature hallucination of a non-existing modality adds side information relevant for the

[46] Cf. Godard et al. [145].
[47] Cf. Zamir et al. [472].
[48] Cf. Varghese, Chandra, and Kumar [433].
[49] Cf. Uhrig et al. [427].
[50] Cf. Zhang and Funkhouser [477].
[51] Cf. Lopez-Rodriguez, Busam, and Mikolajczyk [261].

accuracy of a task.[52] For pose estimation pipelines, we have already seen in chapter 8.2.3 that an additional depth modality can help to significantly boost the accuracy of RGB-only methods which has been used frequently in the literature.

The improvement of pose estimation pipelines with co-modalities certainly remains an open topic that can be further explored and is certainly worth to mention here. However, it is not the main focus of this dissertation. Equipped with the ability to temporally synchronize poses from various sources and reliable displacement measurements, we specifically target now the application of accurate pose signals for the task of sensor fusion to enrich the image information by geometric fusion of multiple modalities.

[52] Cf. Hoffman, Gupta, and Darrell [181].

10 Spatial Modality Fusion

> *Like art, revolutions come*
> *from combining what exists*
> *into what has never existed before.*
>
> **– Gloria Steinem**
> (Moving Beyond Words)[1]

Imaging modalities are omnipresent in our world and coexist with many other pose-relevant sensors. Monocular and stereo setups of monochrome and RGB cameras are included in mobile smartphones together with GPS sensors and IMUs that operate accelerometers and gyroscopes. More recent mobile devices integrate active depth sensors[2] with structured light cameras or time-of-flight technology. Surveillance cameras measure with thermal imaging[3] or use multispectral methods[4] and autonomous car prototypes are equipped with LiDAR remote sensors.[5] Medical diagnoses are often based on radiology where tomographic images such as X-ray computed tomography (CT), magnetic resonance imaging (MRI) or ultrasound (US) scans use penetrating waves or the magnetic moment. Amongst others, nuclear medicine also leverages positron emission tomography (PET) and single photon emission tomography (SPECT) by using gamma rays.
All these imaging methods provide a rich source of information which is oftentimes treated separately, but can mutually benefit each other.

We aim to combine sensor data from different sources to enhance the mutual information value. There are several reasons why somebody may be interested in sensor fusion.
On one hand, we can use multiple modalities as input to improve the accuracy or robustness of a specific task. 6D pose estimation, for instance, benefits from additional IMU data that helps and robustifies visual SLAM and SfM pipelines as discussed in chapter 7.8.2. Besides the support with additional information if one of the inputs is not providing data, e.g. if the line of sight is occluded and the visual signal is blocked, the **combination of multiple input signals** provides a way to leverage consistencies to improve reliability through redundancy.
On the other hand, visual signals can also be used to augment and recombine simultaneously acquired information from various sources in a spatially correct manner. This is of particular interest if different imaging sources provide data which is not mutually present, for instance from different parts of the electromagnetic spectrum. If the relative pose is known, also more

[1] Moving beyond words: Age, rage, sex, power, money, muscles: Breaking boundaries of gender. New York: Simon and Schuster, 1994. Part 4: The Masculinization of Wealth, p.196.
[2] Cf. Park et al. [321].
[3] Cf. O'Conaire et al. [308].
[4] Cf. Denman et al. [84].
[5] Cf. Cao et al. [62].

orthogonal sensing technologies can be combined. An RGB image of an opaque object, for example, can be augmented with information from its inside using an additional tomographic scan that leverages a penetrating wave to visualize information inside the object. An RGB image of an opaque object, for example, can be augmented with information from its inside using an additional tomographic scan that leverages a penetrating wave to visualize information inside the object. The core element to realize such **multi-modal spatial sensor fusion** tasks are high-performance visual pose computation algorithms and 3D vision systems. To prove the use of our pose estimation ideas and algorithms in practice, we investigate applications for spatial modality fusion in three different domains which require individual features from accurate optical pose measurements and tracking systems.

We start with **industrial manufacturing** in section 10.1 where an optical tracking system observes the motion of a robotic head that uses a laser to melt metal for additive manufacturing. It brings together the 3D geometric shape of the manufactured object using a structured light stereo system with a thermal profile acquired from another perspective for quality and process monitoring. Accurate poses are a requirement in this setup for correct spatial alignment and multi-redundant high quality 3D reconstruction.
A second investigation - detailed in section 10.2 - focuses on **mobile augmented reality** in a medical environment. An optical tracker provides poses of a smartphone and an ultrasound transducer. Calibrations are used to bring both the ultrasound images and the RGB video sequence from the smartphone camera into the same reference system where the live ultrasound acquisition is overlayed on the images for guided ultrasound positioning. Due to the low computation capabilities and data transmission bottlenecks for the edge device, we rely on accurate poses at low frequencies and pose upsampling for temporal synchronization.
The final use case deals with a setup in **cooperative medical robotics** for breast cancer staging where a radioactive tracer is injected into the breast tissue. It travels through the directional lymphatic system and gathers in the sentinel lymph node from which a biopsy has to be taken with a needle under ultrasound guidance. A camera-in-hand optical tracking system thereby helps to combine the anatomical information from a handheld ultrasound scanner with the radioactive information acquired by a gamma camera mounted on a robotic manipulator that collaborates with the physician. The setup is described in section 10.3 and requires reliable real-time poses with minimal lag for a safe and seamless interaction.

The essence of analyzing these orthogonal use cases is to explore the generalization capabilities of the developed ideas in practice. Spatial combination of multiple inputs thereby serves as a common denominator and allows to fuse relevant information from multiple sources.

10.1. Use Case 1: Industrial Manufacturing

Leveraging the accuracy of the high performance optical tracking system from chapter 7, we apply it for **quality and heat control in industrial manufacturing**. The project SYMBIONICA[6] investigates the additive manufacturing of smart prosthetics with functional and geometrical

[6]The project "SYMBIONICA – Next Generation Bionics and Smart Prostetics" was funded between 2015 and 2018 with Horizon 2020 as an EU-Project under H2020-EU.2.1.5.1. - Technologies for Factories of the Future, Project Reference: 678144.

customization. Within the project, a machine is built for additive manufacturing with multiple materials using deposition and ablation processes. A metallic powder is heated by a focused high energy laser to 3D print a component layer-by-layer through melting. To monitor the quality of the resulting component, a check is performed where a stereo vision system and a pattern projector extract the fine geometric details of the component. The geometry is combined with its thermal profile acquired through the use of a co-calibrated thermal sensor.

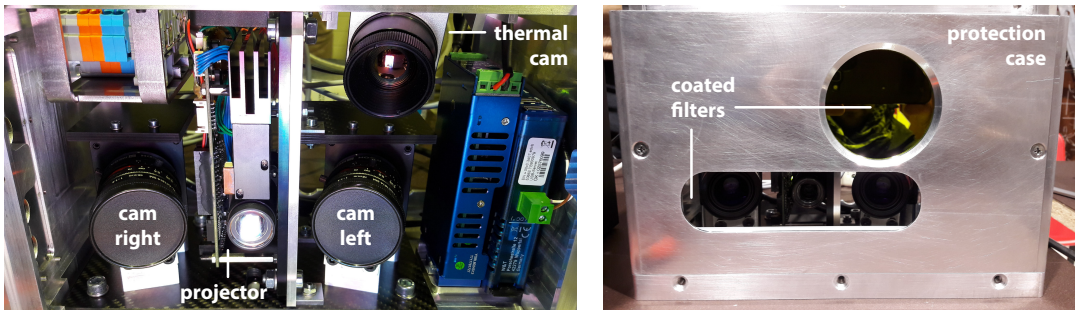


Fig. 10.1. **Vision box for thermal and geometric part inspection.** The vision box is shown with an open blend on the left. A thermal infrared (IR) camera is monitoring the heat dissipation of the manufactured part. It is co-calibrated with a stereo camera setup (cam right, cam left) and a projector between the stereo sensors. The case (right) protects the electronics from hot flying particles throughout the deposition process. The filter glass is used for protection of the sensors with a band-pass filter in their sensitive spectrum. It can be noted that the coated filter for the thermal camera is not transparent for the visible spectrum as can be seen from the image, but passes longer wavelengths.

The **vision inspection box** for this process is shown in Fig. 10.1. All camera intrinsics as well as the extrinsic parameters of the stereo-system are calibrated[7] and the stereo images are rectified.[8] For a stable calibration, the stereo cameras are sandwiched between two carbon plates that show minimal thermal expansion. The vision box is then co-calibrated with the robot forward kinematics through hand-eye calibration[9] and co-calibrated with the **external optical tracking system** to be able to measure everything in a common reference frame.
After deposition of several layers, an inspection is triggered where the vision box is moved to several positions around the workpiece while it is tracked with sub-millimeter precision by our optical tacking system with attached retro-reflective markers to retrieve a position and rotation accuracy beyond the forward kinematics of the robot. This allows to combine individually extracted point clouds acquired from the structured light system (cf. section 6.1.1) and the infrared camera in one reference frame.
The **structured light** system consists of two stereo vision cameras together with a projector placed in the middle that projects a sequential pattern structure into the scene to extract a local point cloud. The consecutive projections of the calibrated projector allow for sub-pixel precise correspondence matches between the rectified images and highly accurate triangulations of the workpiece surface. A third **thermographic camera** is co-calibrated to the remaining two sensors and acquires in each scan an additional image which allows to extract a combined point cloud in reference coordinates that includes information on the heat dissipation of the part. The system can then compare the manufactured part with the planned CAD design and can readjust or ablate the workpiece in case of deviations. The additional heat measures identify heat sinks or bridges and also monitor the time necessary for a cool down of the part until the process can continue.

[7] Cf. section 4.2.
[8] Cf. section 6.3.3.
[9] Cf. section 7.7.2.

The windows with a coated glass protects the vision box from high energy reflections from the melt pool outside the measurable spectrum of the cameras during the manufacturing and protects the sensor from flying particles during the process. Electromagnetic waves in the visible spectrum can pass the lower coating where the projector-stereo system is located. The visible spectrum is blocked by the filter glass before the thermal camera to minimize noise. Its coating, however, lets through radiation of longer wavelengths in the infrared spectrum to enable thermal measurements.

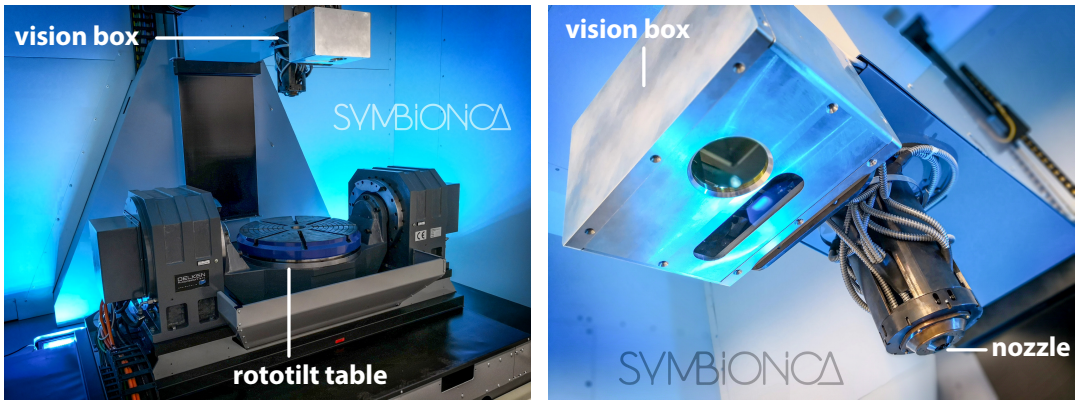


Fig. 10.2. **SYMBIONICA addititive and subtractive manufacturing machine.** The SYMBIONICA machine (left) produces next generation fully personalized bionics and smart prosthetics through additive and subtractive manufacturing. The workpiece is deposited on the rototilt table which has two degrees of rotational freedom. The robotic head (details on right) is able to move along three translational axes (up-down, right-left, front-back) towards the table. Metallic powder is blown through the nozzle and melted with a focused laser beam that shines through the middle part during the manufacturing. The vision box is used for geometric and thermal inspection during the process and is tracked with an external tracking system.

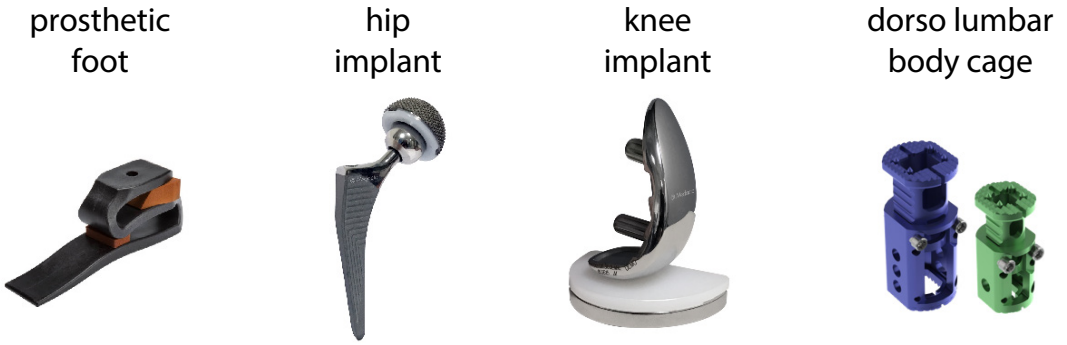


Fig. 10.3. **Use cases for personalized medical parts.** The SYMBIONICA machine addresses manufacturing of prosthetics and implants. Morphologically adjusted parts can be printed with the machine with multiple materials. These include a prosthetic foot (left), an implant of a hip (second from left), a knee implant (second from right), and a dorso lumbar body cage (right).

The entire machine with a close up of the nozzle together with the vision box is illustrated in Fig. 10.2 where it is placed in a $4 \times 4 \times 3$ m^3 cabin which is filled with a reaction suppressing gas during operation. The part manufacturing happens on a rototilt table of 600 mm diameter below a robotic head that can translate in three spatial directions and holds the vision system for quality control and inspection. The entire system has five degrees of freedom: three for the deposition head movements and two from the rototilt table. The head can travel 800 mm in both horizontal X- and Y-direction and 1200 mm vertically.

After automatic inspection and comparison with the planning models, the geometry of the result is controlled with a closed loop inspection and an individualized prosthesis is tailored to the body properties of a patient. The part can be adjusted to fit the dynamic and static needs of a patient with morphological and geometrical part customization. A co-engineering platform allows for interaction during the prosthesis planning where use case projects involve a prosthetic foot (Ottobock, Duderstadt, Germany), a multi-material hip and knee implant (Medacta International SA, Castel San Pietro, Switzerland), and a dorso lumbar body cage (Sintea Plustek, Assago MI, Italy) as illustrated in Fig. 10.3.

A first prototype of a similar vision box with an optical tracking system is adjusted for another machine of significantly larger working volume up to 4000 × 1500 × 750 mm^3 in the BOREALIS[10] project in order to demonstrate additive and subtractive manufacturing processes for complex metal parts. The demonstration cases for this project range from an automotive gearbox (DIAD Group ES, Torino TO, Italy) for the motorsport sector, a hand surgery prosthesis (Sintea Plustek, Assago MI, Italy) for med tech to an accessory drive train main housing (Avio Aero, Colleferro, Rome, Italy) in aerospace engineering. The machine is shown in Fig. 10.4.

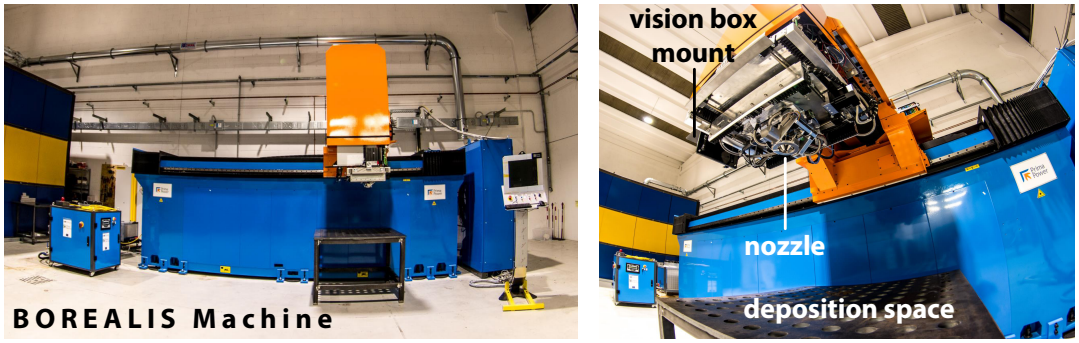


Fig. 10.4. **BOREALIS machine for 3D metal part manufacturing.** The image shows the BOREALIS machine during setup. The left image illustrates the spatial extension of the machine where the human control panel can be used by standing in front of it. The comparably small vision box mount is indicated on the right image. The nozzle is here shown above a deposition space before a rototilt table is mounted.

Both systems are successfully deployed and the manufacturing of the parts shows a possible use case for spatial sensor fusion where **thermal imaging** is **combined with 3D surface extraction from structured light**. The **accurate pose estimation** with an optical tracking system thereby **enables** a **quality control** from combined multiple perspective measurements in a robotic setup.

10.2. Use Case 2: Mobile Augmented Reality

Edge devices have limited computation capabilities and augmented reality scenarios demand synchronize content from various sources. In this second application we focus on the **combination of RGB video** from a mobile device **with ultrasound imagery in** a **real-time** augmen-

[10]The project "BOREALIS – Enlightening Next Generation Material" was funded between 2015 and 2017 with Horizon 2020 as an EU-Project under H2020-EU.2.1.5.1. - Technologies for Factories of the Future, Project Reference: 636992.

tation setup which leverages an optical tracking system that continuously estimates the poses of both the mobile device and the ultrasound transducer.

Augmenting ultrasound images in their spatial location can help for teaching anatomy to novice users and enables holistic surgery planning in the presence of multiple modalities. If the system is capable for real-time feedback, it can be interesting for ultrasound guided interventions and guided needle and tool injections. This form of augmentation has a long history where Bajura et al. [9] already propose an augmented reality system in 1992 which required a considerable amount of hardware and an head mounted display (HMD). Ultrasound guidance can help for biopsies where the team around State et al. [391] proposed an early system leveraging an HMD. They used visual-magnetic tracking for the display and a mechanical tracker for the ultrasound transducer. More modern AR systems have a measurable influence on the accuracy of biopsies.[11] While the pose estimation is commonly done with the help of outside-in tracking systems, also inside-out attachments to the ultrasound probe exist.[12]
We want to utilize a **single commodity smartphone for** the **computation** and exemplify the procedure with the de-facto standard for medical outside-in tracking leveraging the pose upsampling and synchronization methods developed in section 9.1. Other scholars did show elaborate systems based on mobile augmented reality using iOS and Android platforms before. In the work of Kiss et al. [221], for instance, the team shows a tablet version for augmentations of cardiac anatomy that can be used for teaching the anatomy of the human heart. In their setup, the orientation of the ultrasound transducer is tracked with the help of an IMU and Palmer et al. [319] extend the concept with visual markers that are printed on paper. We rely instead on a more accurate optical tracker with retro-reflective rigid body markers and pre-calibration to bring all involved components in a common reference frame in order to perform pose-guided anatomical scans where a target pose is provided as a landmark to start anatomical inspection and help novice users.

An annotated view of the augmented scenario as rendered on the mobile phone screen is shown with all involved components in Fig. 10.5. We use an optical tracking system for continuous pose estimation of two markers that are rigidly attached to the mobile phone and the ultrasound transducer. The mobile phone camera is calibrated[13] and the attached marker is co-calibrated to the RGB reference view with a hand-eye calibration[14]. We use a modified version of ViSP[15] to run in Java on our mobile phone. For calibration between the ultrasound image and the marker of the ultrasound probe, we use the PLUS[16] framework as described in section 8.1.4. With the OTS measurements, we can then describe all poses in the coordinate system of the RGB camera on the mobile phone. The relevant reference frames are illustrated in Fig. 10.6.

A **pose-aware rendering of live ultrasound data** augmented on top of the RGB image of the mobile phone is then reached in several steps. We communicate all poses and the ultrasound images over Wi-Fi via OpenIGTLink[17] where we leverage a Java implementation of the pro-

[11] Cf. Rosenthal et al. [353].
[12] Cf. Stolka et al. [396].
[13] Cf. section 4.2.
[14] Cf. section 7.7.2
[15] Cf. Marchand, Spindler, and Chaumette [275].
[16] Cf. Lasso et al. [236].
[17] Cf. section 7.6.

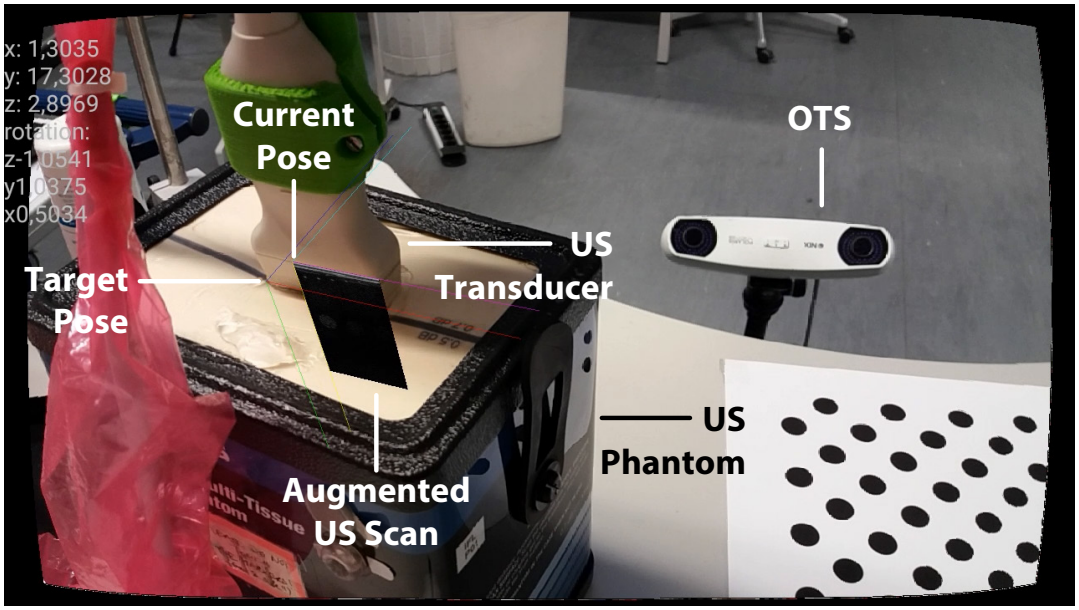


Fig. 10.5. **Augmented reality setup on RGB image from mobile phone.** The image shows the medical setup where an optical tracking system (OTS) computes the poses of both the mobile device and the ultrasound (US) transducer. The current ultrasound image is augmented at the spatial correct location in its current pose while a predefined target pose defines a structure of the phantom that is to be explored. The coordinate frames thereby help to adjust the handheld device. The displacement between the current and the target pose is overlayed on top left in millimeters and degrees for rotations around the axes. The pattern board on the lower right is used for hand-eye calibration of the mobile phone. Black borders around the image arise from the camera calibration which corrects the image for distortions.

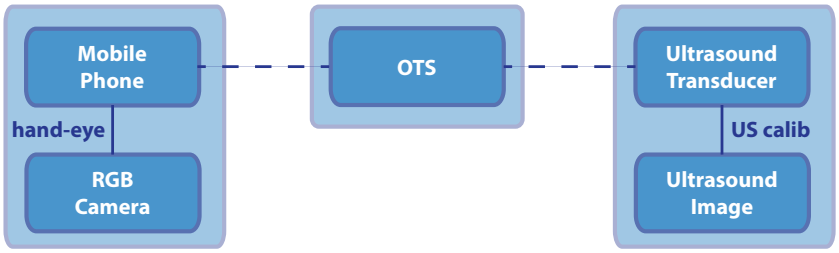


Fig. 10.6. **Reference frames and calibration processes of mobile AR setup.** The image shows the involved reference frames in our medical AR setup. The optical tracking system (OTS) depicted in the centre constantly provides poses for the markers attached to the mobile phone (left) and the ultrasound transducer (right). Before we start the real-time application, we calibrate the RGB camera of the mobile phone and use the OTS poses to perform a hand-eye calibration (eye-in-hand variant) by observing a static pattern which is visible in Fig. 10.5. The marker on the ultrasound transducer is calibrated via a stylus (see section 8.1.4) to the ultrasound image. This allows to describe all poses in the RGB camera system.

tocol to run on Android.[18] For the augmentation we utilize OpenGL ES 2.0[19] for rendering and use a Polaris Vicra (Northern Digital Inc., Waterloo, Canada) for tracking. Due to the low computational power, we leveraged a simple time synchronization of the poses with the upsampler from section 9.1 where an incoming ultrasound image time stamp determines the time for which a pose is calculated based on the last estimates. The mobile phone used for our tests is a Samsung Galaxy J5 (2016) running Android 6.0 Marshmallow with API level 23. The device integrates a 5.2" display with a resolution of 1280×720. It uses a Qualcomm MSM8916 Snapdragon 410 (28 nm) chipset and a 1.2 GHz Quad-core Cortex-A53 CPU as well as an Adreno 306 GPU and 2 GB internal RAM. The main camera provides wide angle RGB images with 13 MP and up to full HD video of 1920×1080 at 30 fps through a 28 mm lens with F1.9. We use screen resolution images for all our experiments.

In order to validate the setup, we perform a small **user study** with six medical experts which are asked to find a nylon wire cross in a phantom under ultrasound and AR-guidance in two different ways where we provide a target pose in which the wire crossing is visible as reference. In a real setup such an approximate pose could come from a preoperative planning that indicates the position of an anatomical structure to be scanned in detail. There are two wires attached in the phantom that span a plane. The goal is to align the ultrasound image with the phantom in such a way that this plane becomes incident with the US scan and a cross is visible in the ultrasound image with the cross section being visible in the image centre.
In the first test, we augment the ultrasound image on the RGB view and provide guidance to the target pose only with the numerical information shown in the upper left part of Fig. 10.5. The second attempt is done with the additional guidance of showing the current and target reference coordinate axes. A view of the experimental setup is shown in Fig. 10.7.

We measure the time from a start signal until the medical expert signalizes to have finished the task which turns out to be a correct alignment in all but one cases. After this test we give the experts time to familiarize themselves with the augmentation until they feel comfortable and rerun both experiments. The results are summarized with their average localization time in Fig. 10.8. While the self-teaching does not significantly improve the speed for the guidance with coordinate frame augmentation, it improves the results upon the numeric guidance from 14.95 sec to 11.30 sec. However, the average times for the visual pose guidance with 8.26 sec before and 7.77 sec after training indicate that the AR coordinate support is more helpful than showing the numerical deviations. At the same time, the close durations before and after training prove that the coordinate frame alignment is more intuitive.

Due to the low computational power of the edge device, our first prototype was realized with an average frame rate of approximately 1 fps mainly influenced by the rendering time which increased the difficulty of the alignment process for our expert group. A correct **temporal pose synchronization with quaternionic upsampling** at low frame rates is crucial to minimize the jitter of the rendered ultrasound image and the augmented reference frames. The successful finish of the test shows that this can be realized with the proposed techniques. For a future version of the demonstrator, further rendering optimization could be investigated or the mobile phone could be exchanged with a head mounted display.

[18] Cf. Lakshminarayanan [234].
[19] Cf. Munshi, Ginsburg, and Shreiner [296].

Fig. 10.7. **Experimental setup for wire phantom alignment.** The mobile phone is fixed in front of the phantom while the user is asked to align the ultrasound image with a plane given by two wires inside the water bath. The numerical values (top left of the screen) show the deviation from a target helper pose which is augmented here (red-green-blue) in comparison with the current pose (magenta-yellow-cyan). The current ultrasound image is overlayed as anatomical guidance.

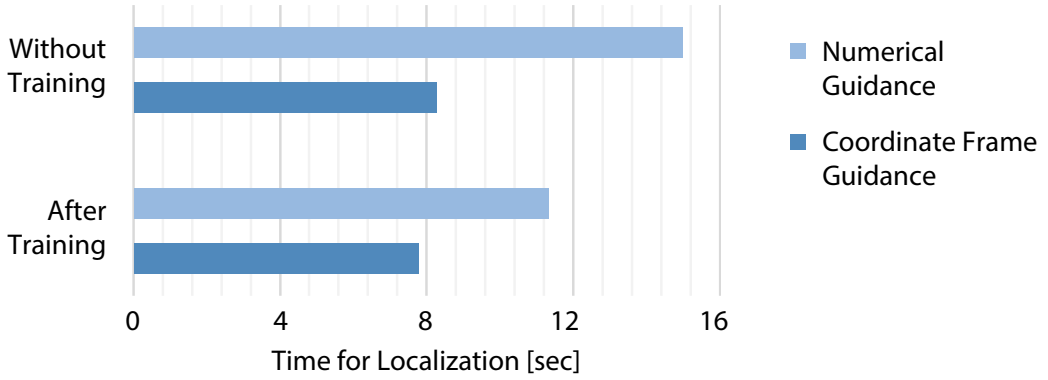


Fig. 10.8. **Positive feedback from the medical experts.** The plot shows the average time required for the alignment of an ultrasound image with a wire phantom. Two different augmentations are tested: Numerical guidance shows the parameter deviation as numbers while coordinate frame guidance shows a current and target reference frame close to the goal pose (see Fig. 10.7). The first set of experiments (no training) is done by the medial experts without spending time with the system beforehand while the second set of experiments (after training) is done after getting familiar with it.

10.3. Use Case 3: Cooperative Medical Robotics

The medical sector provides a plethora of imaging devices targeting different aspects of diagnostic scenarios. We leverage the reliability and efficiency of the **camera-in-hand tracking** system presented in section 7.9 **for** a **real-time combination of ultrasound and nuclear imaging** in a collaborative medical setup.
Our proposed framework constitutes an intraoperative system that combines multiple imaging modalities in real-time to guide the surgeon during a breast cancer staging procedure. We program a medical robotic manipulator to position a gamma camera in such a way that its information can be spatially fused with the anatomical data the physician is investigating with a handheld ultrasound scanner such that a needle punch biopsy can be performed in a simulated interventional setup under multi-modal imaging guidance with combined live nuclear and anatomical information.

In the following, we provide the idea, implementation, and the prototype tests of a medical collaborative system that uses a camera-in-hand optical system mounted jointly with a radioactive sensor on a lightweight robotic arm to realize a real-time fusion of 2D ultrasound with 2D gamma imaging. The machine allows for automatic co-modality support of the surgeon during an anatomical scan which is applied for sentinel lymph node needle biopsy to demonstrate its practical relevance. We briefly introduce the medical background for breast cancer staging with sentinel lymph node biopsies in section 10.3.1 and describe our system and its components in part 10.3.2 before we detail the studies and evaluation in section 10.3.3.
The results have been presented in an oral and poster presentation at the International Conference on Medical Image Computing and Computer-Assisted Intervention MICCAI 2015[20] and have been extended for a journal version published in the International Journal of Computer Assisted Radiology and Surgery IJCARS 2016[21] which received the IJCARS MICCAI 2015 Special Issue Best Paper Award.[22] The joint work combines robotics and 3D computer vision. This thesis contributes the computer vision part of the collaborative system while the first author was responsible for the robotics side of the demonstrator.

10.3.1. Medical Motivation

Breast cancer manifests through a series of symptoms such as a lump in the breast tissue or a change of the breast shape caused by an abnormal local cell growth. While it affects both male and female patients, the risk factor for women is significantly higher.[23] It is the most common cancer in female patients[24] and the mortality rate varies strongly with the growth properties of the cancer and its spread. The investigated 5-year survival rate for cancer with only local growth lies at 99%.[25] This rate drastically decreases when the tumor spreads and forms metastases in different sites within the body often through the directional lymphatic

[20] Cf. Esposito et al. [101].
[21] Cf. Esposito et al. [102].
[22] The figures are reprinted here with the permission of Springer Nature.
[23] Cf. Fentiman, Fourquet, and Hortobagyi [110].
[24] Cf. Siegel, Miller, and Jemal [378].
[25] Cf. Siegel et al. [377].

system resulting in a 5-year survival rate of only 26% after metastasizing. Thus, **staging** the severity of the cancer is a crucial factor for the outcome and measures of a medical treatment. This process involves an assessment of the magnitude of the primary cancer together with an evaluation of the extend of its spread.

The medical **treatment** usually involves an intervention in which the tumor is removed (lumpectomy) or the breast is amputated (mastectomy).[26] Since the cancer can spread through the lymphatic system that connects to a set of lymph nodes in the axillary region, an axillary lymph node dissection can be additionally performed. A significant amount of 10 to 40 lymph nodes are removed in this surgical procedure and consecutively examined for their pathological characteristics.[27] Since some decades, it is known that the closest lymph nodes indicate the likelihood for metastases reliably.[28] Removal of these so-called **sentinel lymph nodes** (SLNs) can stop the cancer spread[29] and an analysis of their tissue with histopathological measures is useful to stage the cancer.[30] Thus, a punch biopsy that provides sentinel lymph node tissue can be used as a minimally invasive procedure for reliable breast cancer staging in oncology. While being less invasive than an open surgery, the current imaging technologies prevent needle punch biopsies from being an equally reliable procedure.[31] A medical expert can use an ultrasound scan to retrieve the anatomical information and identify lymph nodes in the respective body part.[32]

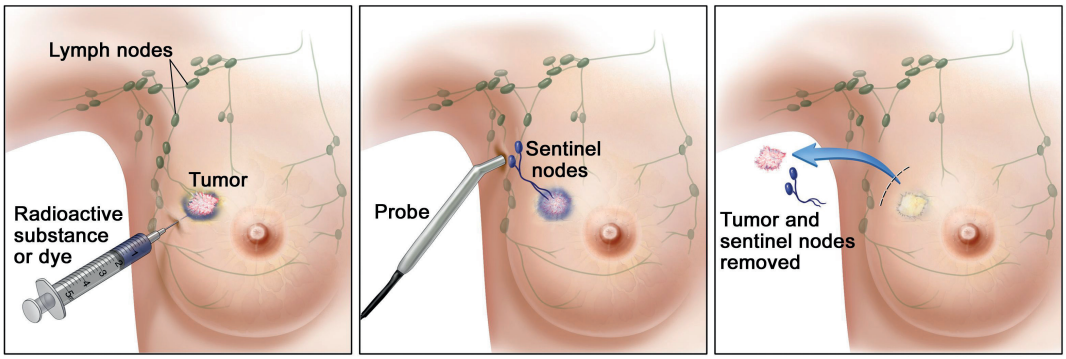


Fig. 10.9. **Sentinel lymph node biopsy and tumor removal for breast cancer treatment.** A tracer is injected into the breast tissue close to the tumor (left). Common substances involve nuclear tracers and visible dye. The tracer fluid travels through the directional lymphatic system (centre) and makes it possible to detect the sentinel nodes with a probe or in an open surgery. The identified sentinel nodes can then be removed together with the tumor (right).

The detection of small lymph nodes, however, is intricate and a reliable retrieval of axillary nodes with a size below 5 mm cannot be guaranteed[33] while some of them may still be necessary indicators for the staging.[34] The anatomical structure of multiple dozens of lymph nodes close to each other in the axilla region requires additional information for the correct identification

[26] Cf. Whelan et al. [451].
[27] Cf. Lin et al. [255].
[28] Cf. Valagussa, Bonadonna, and Veronesi [432].
[29] Cf. Krag et al. [225].
[30] Cf. Krag et al. [226].
[31] Cf. Joseph, Oepen, and Friebe [199].
[32] Cf. Hoskins, Martin, and Thrush [184].
[33] Cf. Tate et al. [405].
[34] Cf. Obwegeser et al. [313].

of the first lymph nodes connected to the tumor tissue.[35] To identify sentinel nodes, one can use multi-redundant tracers to support the search. A common procedure is a **tracer injection** close to the primary tumor in the breast tissue as illustrated in Fig. 10.9. The tracer fluid then spreads through the directional lymphatic system and accumulates in the sentinel nodes which can then be identified and removed together with the tumor. Such tracers can consist of visible blue dye, fluorescent or radioactive material. Visible dye colours the sentinel nodes with a fluid like patent blue V (Bleu Patenté V, Guerbet, Brussels, Belgium) and is a direct indicator in open surgeries. Fluorescent material can also be used up to a tissue depth of 2 cm to indicate the lymph node position more accurately[36] and a radio-tracer such as ^{99m}Tc-nanocolloid emits high energy gamma rays which can be used as a redundant factor for radio-guidance in open surgery.[37] While the former two require a close visible inspection, the advantage of the latter is that it still indicates the direction of the node even without an open surgery. In order to measure the radioactive signal, freehand SPECT can be used to detect gamma particles. While early methods use a 1D sensor with acoustic feedback, 2D gamma cameras are also used in oncology.[38] These measure the low amount of incoming gamma particles as events on a 2D grid. Similar to the exposure time of a vision sensor, we can integrate this signal over time into a visible 2D heat map indicating the directions for the incoming gamma rays. The unstructured spatial fusion of nuclear imaging with co-modalities such as ultrasound, however, is challenging. A physician observing the anatomical information with an ultrasound transducer in one and a gamma camera in the other hand then receives the information of a 2D slice through the tissue on one screen with a spatial resolution allowing to identify small structures (US) while he additionally receives a low resolution gamma image (e.g. 16×16 pixels) as a projection of the radiation into the 2D camera sensor (gamma) on another screen. Even if both images are shown on the same monitor a pure **cognitive fusion** of the modalities **remains very difficult** and the **hands of the surgeon are both occupied** with the imaging devices ultimately not allowing for a biopsy under guidance from both modalities. Additional lymph nodes that are located close to each other reduce the likelihood of a correct identification of the single sentinel lymph node or multiple sentinel nodes.

Interventional methods to combine ultrasound imaging with the information from a gamma camera exist[39] and 3D ultrasound has been combined with freehand SPECT by the team of Okur et al. [315]. Their system, however, requires two subsequent scans and is prone to tissue deformation during a 3D ultrasound compounding or the respective gamma extraction which is performed offline. While providing a solution for spatial fusion of both modalities, it still requires the biopsy to be performed solely using ultrasound. Furthermore, the external tracking system used in their case comes with all the line of sight drawbacks discussed in chapter 7.8. A spatially correct fusion requires a third hand to hold the gamma camera. One solution could be that a human assistant holds the gamma device. This increases the complexity of the cognitive fusion process drastically and includes another stress component for the surgeon. We propose to utilize a **robotic assistant** to provide the functional information of the tracer by adjusting the gamma camera displacement at a constant relative pose to the manual ultrasound scan. In this way, the surgeon can fully concentrate on the ultrasound images and the anatomy of the patient without changing the procedure of ultrasound guidance for the needle punch biopsy.

[35] Cf. Bugby, Lees, and Perkins [45].
[36] Cf. Vorst et al. [438].
[37] Cf. Vidal-Sicart and Valdés Olmos [435].
[38] Cf. Bricou et al. [39].
[39] Cf. Wendler et al. [449].

Additional to that, we are then able to augment the nuclear information on the ultrasound image in real-time providing an additional radioactive heat map on top of the images for a live functional and anatomical scan.
We describe the individual system components as well as the control and visualization pipelines in the next section.

10.3.2. Collaborative US-Gamma Imaging

We equip a robotic manipulator with a gamma camera to measure radioactivity of a radioactive tracer injected in a breast tissue phantom. A camera-in-hand stereoscopic tracking provides the relative pose between a handheld ultrasound transducer of a physician and the gamma camera that is rigidly attached to the end effector. The robot then positions the gamma camera perpendicular to the ultrasound imaging plane such that we can combine the two modalities in an **augmented multi-modal visualization**. The full system is shown in Fig. 10.10 during an example run of a sentinel lymph node biopsy with a punch biopsy needle on a phantom. The collaborative robotic pipeline and the individual components of the system are described hereafter.

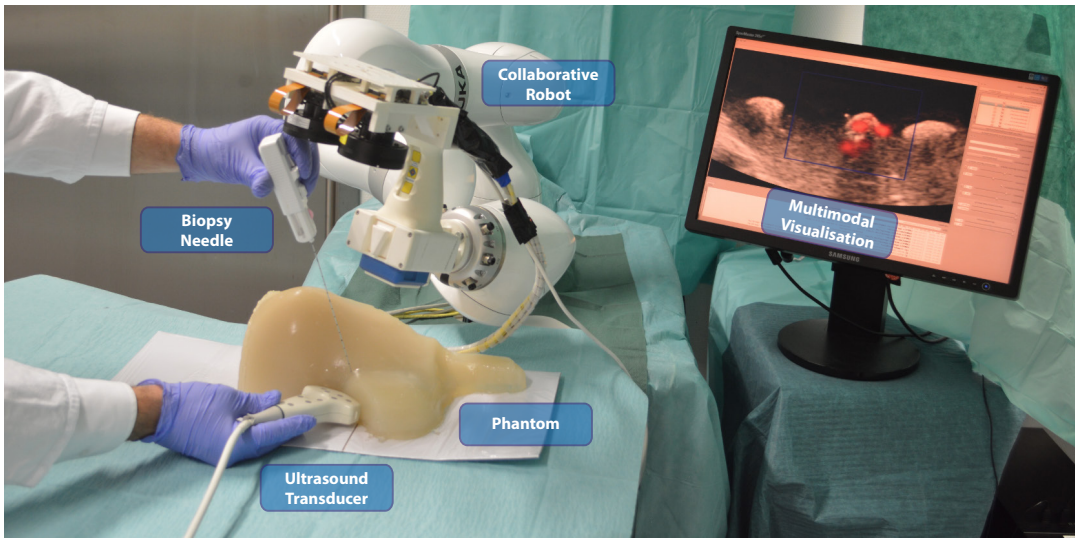


Fig. 10.10. **Collaborative robotic punch needle biopsy under multi-modal guidance.** A medical expert examines a phantom (centre) in the form of an upper part female torso which includes synthetic lymph nodes, some of which are filled with a radioactive tracer. A punch needle biopsy is performed here under both ultrasound and gamma guidance. The needle (left) is inserted into the synthetic tissue while an ultrasound transducer (bottom) provides anatomical information. The probe with self-adhesive circular markers is tracked by a camera-in-hand vision system which is jointly mounted with a gamma camera on the robotic flange (top) that follows the examiner. The modalities are spatially fused and a joint multi-modal image is visualized live (right). The lymph nodes are visible as white blobs and radioactive measurements are augmented in red with an opacity proportional to their intensity inside the field of view of the gamma camera which is indicated in blue.

In order to realize a sentinel lymph node biopsy under multi-modal guidance, we construct a camera mount for a robotic arm that holds a gamma camera together with the stereo tracking system from section 7.9. Running the tracking algorithm from section 7.3, we are capable to reliably and robustly detect and track the markers attached to a handheld ultrasound transducer

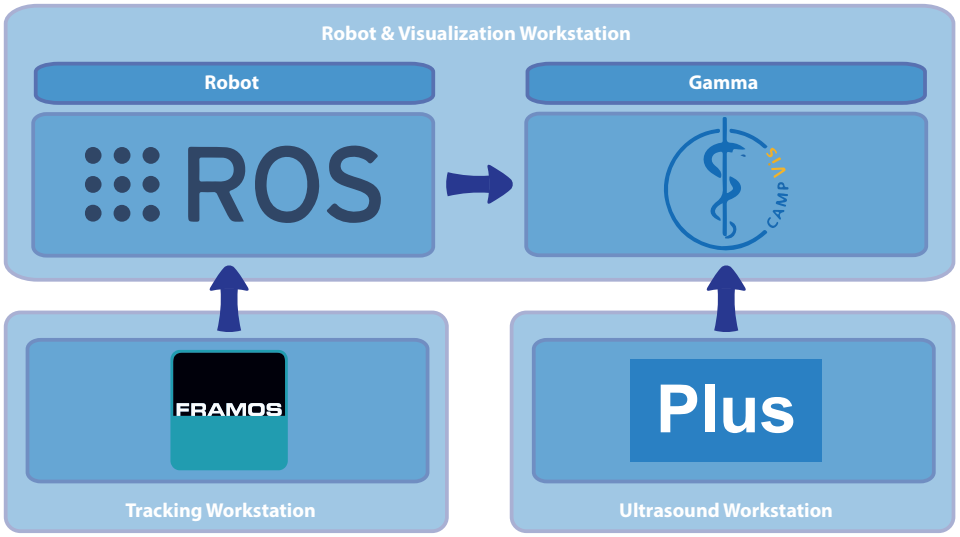


Fig. 10.11. **High-level system components for collaborative modality fusion system.** Three workstations run different components of the collaborative fusion system. Visualization and robot control is done on the robot and visualization workstation (top) which integrates the robot control via ROS (left) and the joint multi-modal visualization of both gamma and ultrasound within CAMPVis (right) which is provided with the relative pose between gamma camera and ultrasound. Image processing, optical pose estimation and pose tracking is done on the tracking workstation (lower left) which runs an instance of the FRAMOS software framework. It sends poses of the tracked ultrasound transducer to ROS. The ultrasound acquisition and imaging is done on an ultrasound machine (lower right) running Plus. It provides the ultrasound image and its meta data to CAMPVis.

while a surgeon is performing an ultrasound scan with it. Calibration procedures for rigidly attached components allow to steer the robot through iterative manipulation of its end effector in such a way that the gamma camera is positioned orthogonal to the ultrasound image plane while the physician is manually adjusting the ultrasound transducer. The medical expert then has a free hand to extract tissue with a punch biopsy needle from a correctly identified sentinel lymph node in a lymph node cluster that is highlighted through a radio-tracer under joint multi-modal guidance.

The involved high-level **system components** are illustrated in Fig. 10.11. An UltraSonix RP system (Ultrasonix, MA, USA) is used in research mode with a C5/60 curvilinear probe and the Ultrasonix Ulterius SDK on a dedicated **ultrasound workstation** for acquisition and processing of the US sensor data. We use the ultrasound toolkit Plus[40] as in section 8.1.4 and communicate images and ultrasound meta data such as pixel spacing via an OpenIGTLink server that provides the data over a Gigabit Ethernet connection (cf. section 7.6).
The OpenIGTLink interface is also used on the **tracking machine** to provide the US poses. The workstation runs the algorithm of Busam et al. [49] as detailed in section 7.3 inside the FRAMOS Imaging Systems (FIS) framework (FRAMOS GmbH, Taufkirchen, Germany). The stereo tracker estimates the pose of the marked ultrasound transducer relative to the master camera and performs quaternionic upsampling[41] to super-sample the hardware acquisition frequency and ease temporal synchronization. The machine is attached to its cameras via Ethernet with a PCI Express card and uses a second port for connection with the robot control system that also renders the augmented ultrasound images.

[40] Cf. Lasso et al. [236].
[41] Cf. section 9.1.

The **robot and visualization workstation** handles the robot control with ROS[42] and is connected to the robot via Ethernet. Respecting all calibrated transformations, the poses are internally provided to the visualization part. For this, an instantiation of CAMPVis[43] is set up to augmented the multi-modal image output. It connects via Ethernet to the ultrasound image server which feeds the images. All machines are synchronized via NTP[44] to be able to use time stamp information from a synchronized clock and to stabilize the system against oscillation effects caused by delays.[45]

The tracking system **hardware** constitutes the same components as the described in section 7.9 and we leverage a KUKA LWR iiwa R800 robotic arm (KUKA Roboter GmbH, Augsburg, Germany) with the KUKA Sunrise control software. The radioactivity is detected with a 2D CrystalCam gamma camera (Crystal Photonics, Berlin, Germany) attached to a customized 3D printed mount that rigidly attaches also the camera-in-hand tracker. For the medical experimentation, we utilize a standard punch biopsy needle (HistoCore 250 mm, BIP GmbH, Germany).

10.3.2.1. System Coordinates & Calibration

For all involved components we require to know their spatial relationship to be able to fuse the data. The relevant coordinate reference frames are summarized in Fig. 10.12 where the world anchor and static reference frame is the robot base. We calibrate all involved components to each other such that they can be described in robot base coordinates.

A first step involves a **camera calibration** of both stereo cameras and their relative pose following the calibration steps described in section 4.2. Knowing their relative poses allows for image rectification[46] and to run the stereo tracking algorithm from section 7.3. We attach self-adhesive retro-reflective circular markers to the ultrasound transducer and **train a marker setup** for the attached marker as described in section 7.5.1 by manually moving the transducer in the line of sight of the vision system. Varying the pose of the robotic arm lets us change the position and orientation of the robot end effector. As both the gamma camera and the camera-in-hand tracking system are rigidly attached to the robot flange, their relative pose is static. While observing a fixed object on the table as shown with the ultrasound transducer in Fig. 10.13, we can run a **hand-eye calibration** routine (**eye-in-hand** variant) when moving the end effector around it.[47] This provides the relative pose between the robot flange which we know by the forward kinematics of the robot and the tracking system. The remaining poses towards the gamma camera and the gamma image plane can be retrieved from the **CAD model** of the 3D mount and its internal description.
An external Polaris Vicra tracker (Northern Digital Inc., Waterloo, Ontario, Canada) is then co-calibrated with a rigid body marker attached to the end effector utilizing a hand-eye calibration in **eye-on-base** variant. Using a pointer with a marker visible to it, we can perform a **pivot calibration**[48] and determine the **ultrasound calibration** for the relative poses between the

[42] Cf. Quigley et al. [338].
[43] Cf. Schulte zu Berge et al. [371].
[44] Cf. section 9.1.
[45] Cf. Corke and Good [73].
[46] Cf. section 6.3.3.
[47] Cf. chapter 7.7.2.
[48] Cf. section 7.7.1.

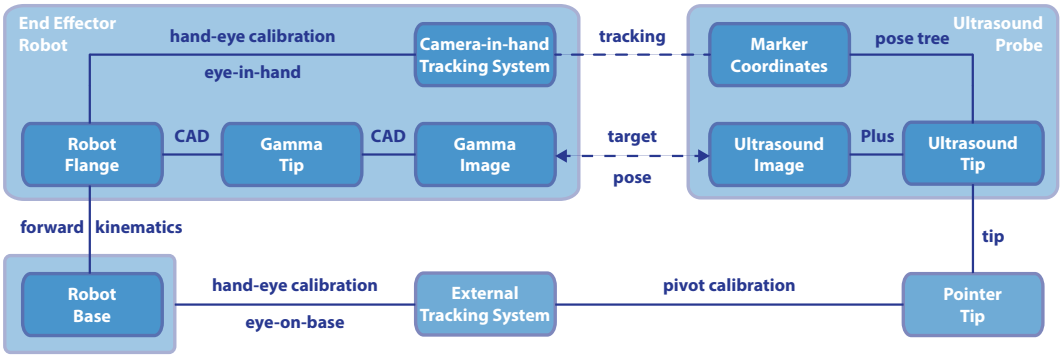


Fig. 10.12. **Relevant coordinate reference frames and calibration routines.** We illustrate the most notable coordinate systems and involved calibration processes. The bigger boxes define rigidly attached components for the robot (left) and the ultrasound transducer (right). The robot is attached to the robot base (lower left) and its dynamic arm with the associated forward kinematics defines the position of its end effector. At the robot flange, we rigidly attach the gamma camera as well as the camera-in-hand stereo vision system. The 3D mount and the gamma CAD model define the pose towards the gamma tip and its image. A hand-eye calibration (eye-in-hand variant) is used to calibrate the pose to the rigidly attached camera-in-hand system which is calibrated for its intrinsic and extrinsic parameters. It constantly tracks the pose of self-adhesive markers attached to the ultrasound probe (dashed line). An external tracking system (bottom, centre) is co-calibrated with it and thus with the robot base in a hand-eye calibration step (eye-on-base variant). A pointer with a rigidly attached spherical marker is calibrated with a pivot calibration and used to determine the poses on the ulrasound probe (right) between ultrasound tip and its image using Plus. The pose tree which starts at the static robot base can be used to determine the relative pose between marker coordinates and ultrasound tip. We can vary the target pose (dashed line in the centre) through motions of the robotic arm.

ultrasound tip and the ultrasound image by pointing to several locations within the ultrasound image. The metallic pointer is visible in the ultrasound image and we can leverage the freehand calibration provided in the Plus ultrasound framework.[49] The remaining pose between the trained ultrasound marker system and the ultrasound tip as pointed by the pointer is then calculated using the pose tree from robot base to the marker coordinates and from robot base to the ultrasound tip. The so calculated relative pose is determined via **pose averaging** with the Weiszfeld algorithm[50] and an input of ten poses to minimize error propagation.
The entire calibration process takes time and is valid until the individual components are detached. In a typical setup, the calibration of the camera intrinsics and extrinsics requires approximately 30 min, object training needs 5 min, and the hand-eye routines require 10 min each. The pivot calibration costs 10 min and the ultrasound calibration requires around 30 min. The remaining spatial calibration needs another 10 min. In total, the calibration procedure takes about 1 h 45 min if all components function perfectly without errors. This does not include the manual CAD estimates.

The external tracker can finally be neglected as it only helps to simplify the calibration with existing material such as the pointer. We end up with two relevant dynamic components which are illustrated in boxes in Fig. 10.12. On one hand, we can manoeuvre the ultrasound probe while the vision system can track its pose. This causes a change between the gamma image and the ultrasound image reference frame. On the other hand, we can actively steer the robotic arm which also changes this relative pose. The other relative displacements are static and remain the same. To support a medical expert in an anatomical scan, we do not intervene the

[49] Cf. Lasso et al. [236].
[50] Cf. Hartley, Aftab, and Trumpf [164].

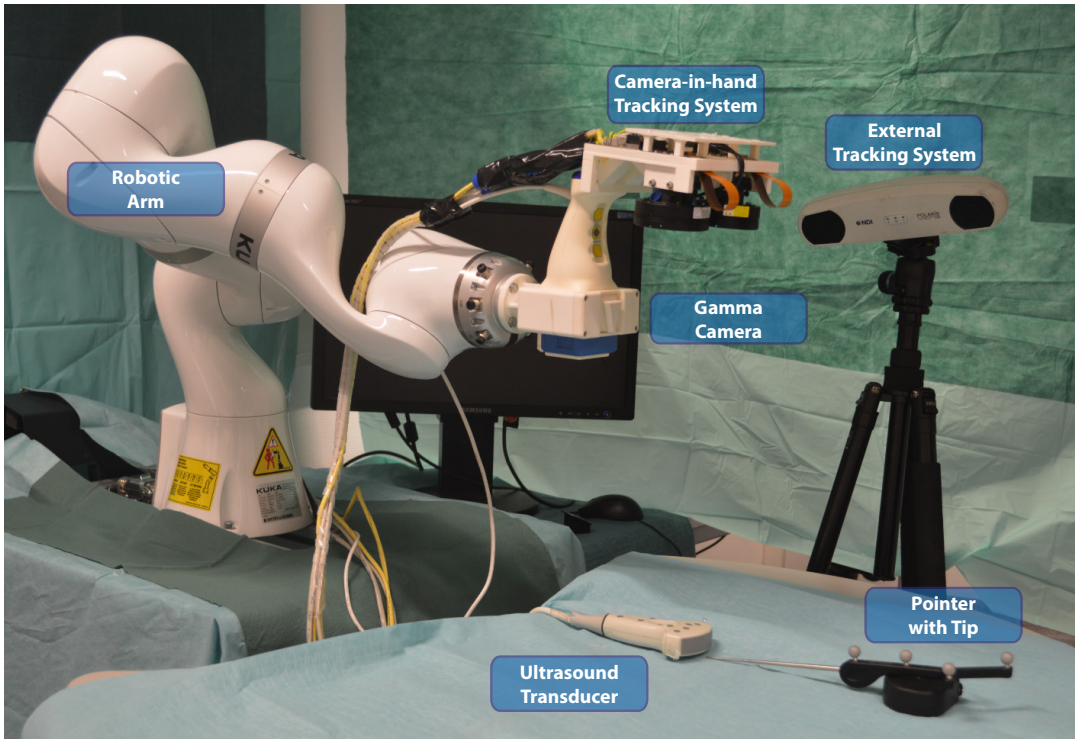


Fig. 10.13. **Calibration setup for component co-calibration.** The picture shows the setup for a calibration between the individual components. The robotic arm (left) is brought to different poses such that the tracked poses of the camera-in-hand tracker (top, centre) towards the self-adhesive markers on the ultrasound transducer vary and a hand-eye calibration (eye-in-hand variant) can be performed. An additional marker attached to the end effector allows for co-calibration with the external tracker (top right). The tip of the pointer (lower right) is calibrated with a pivot calibration such that the external tracker can determine its tip position via the spherical rigid body marker. Since the tip is visible in an ultrasound image, it can be used to calibrate the ultrasound image to its tip. The entire process requires multiple steps.

manual adjustment necessary for the anatomical scan, but **vary the robotic arm in such a way that the relative pose between the gamma image and the ultrasound image is kept** relatively **constant**.

We define the relative pose between them by aligning their centres and demand co-planarity between the image planes. Following the same convention as in chapter 8.1.2, we denote the transformation from A to B with ${}^{B}\mathbf{T}_{A}$. The target pose of the robotic flange with respect to the ultrasound image is then defined by

$$ {}^{\text{US}}\mathbf{T}_{\text{flange}} = {}^{\text{US}}\mathbf{T}_{\text{gamma}} \cdot {}^{\text{gamma}}\mathbf{T}_{\text{flange}}, \tag{10.1} $$

where the transformation ${}^{\text{US}}\mathbf{T}_{\text{gamma}}$ between the gamma camera and the ultrasound probe is given by the ideal plane-to-plane displacement and ${}^{\text{gamma}}\mathbf{T}_{\text{flange}}$ is static and predetermined by our CAD calibration.

The remaining relevant coordinate frames are illustrated in Fig. 10.14 and we can express the ultrasound reference frame at time t in robot base coordinates with

$$ {}^{\text{base}}\mathbf{T}_{\text{US}}(t) = {}^{\text{base}}\mathbf{T}_{\text{flange}}(t) \cdot {}^{\text{flange}}\mathbf{T}_{\text{OTS}} \cdot {}^{\text{OTS}}\mathbf{T}_{\text{marker}}(t) \cdot {}^{\text{marker}}\mathbf{T}_{\text{US}}, \tag{10.2} $$

where ${}^{\text{marker}}\mathbf{T}_{\text{US}}$ comes from the ultrasound and pose tree calibration, ${}^{\text{OTS}}\mathbf{T}_{\text{marker}}$ is provided by the optical tracker, ${}^{\text{flange}}\mathbf{T}_{\text{OTS}}$ comes from the hand-eye calibration, and ${}^{\text{base}}\mathbf{T}_{\text{flange}}$ is defined

through the forward kinematics of the manipulator. This allows to describe the next best robot pose of the robot flange at time $t+1$ in dependency of the quasi-static pose between gamma and ultrasound image and the latest pose estimation as

$$
\begin{aligned}
&{}^{\text{base}}\mathbf{T}_{\text{flange}}(t+1) && (10.3)\\
&= {}^{\text{base}}\mathbf{T}_{\text{US}}(t) \cdot {}^{\text{US}}\mathbf{T}_{\text{flange}} && (10.4)\\
&= {}^{\text{base}}\mathbf{T}_{\text{flange}}(t) \cdot {}^{\text{flange}}\mathbf{T}_{\text{OTS}} \cdot {}^{\text{OTS}}\mathbf{T}_{\text{marker}}(t) \cdot {}^{\text{marker}}\mathbf{T}_{\text{US}} \cdot {}^{\text{US}}\mathbf{T}_{\text{gamma}} \cdot {}^{\text{gamma}}\mathbf{T}_{\text{flange}}. && (10.5)
\end{aligned}
$$

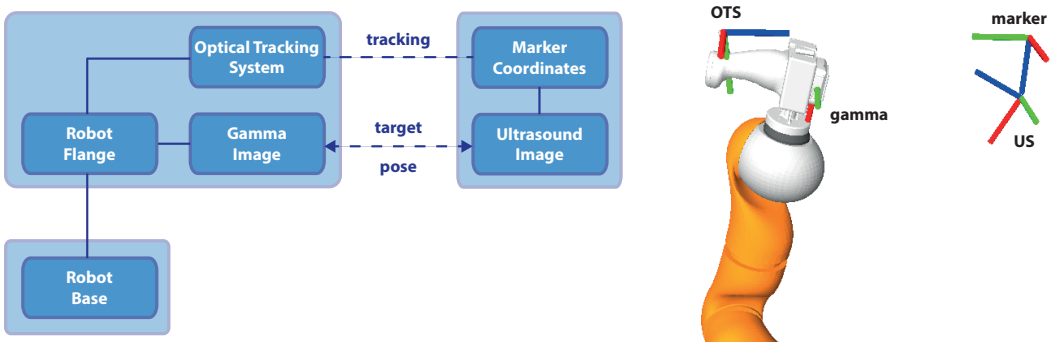


Fig. 10.14. **Reference frames for robot control.** The left side shows the relevant reference frames for the robot control where the three bigger blue boxes indicate rigidly attached coordinate systems. The robot base (lower left) is connected with the end effector that consists of the camera-in-hand optical tracking system (OTS) and the gamma camera which are both rigidly attached to the robot flange. The OTS tracks the ultrasound transducer via its attached markers (dashed line). The relative pose between gamma and US (indicated with the arrows) is required to be relatively constant. The coordinate system abbreviations of the non-robotic components are visualized on the right.

Constant updates of the next best robot motion are calculated with most recent tracking data and the current flange information. For safety reasons and to minimize vibrations and oscillations, we denoise the poses[51] with a pose average over a window of ten poses and actively manipulate the robot pose only if the computed residual displacement compared with the current robot pose exceeds a certain threshold. This decision does not harm the accuracy of our visualization as all the relative poses are known at any time in order for a correct spatial fusion of the modalities.

For the multi-modal visualization, we require also the **transformation between** the **image coordinates**. The gamma coordinates, and thus the radioactive events can be described in ultrasound coordinates as

$$
{}^{\text{US}}\mathbf{T}_{\text{gamma}}(t) = {}^{\text{US}}\mathbf{T}_{\text{marker}} \cdot {}^{\text{marker}}\mathbf{T}_{\text{OTS}}(t) \cdot {}^{\text{OTS}}\mathbf{T}_{\text{flange}} \cdot {}^{\text{flange}}\mathbf{T}_{\text{gamma}}. \qquad (10.6)
$$

The gamma camera is constructed out of multiple scintillation chambers arranged in a 2D array.[52] While it is possible to adopt complex gamma camera modeling[53] for the device at hand that target the needs of 3D reconstruction, we adopt a simple model motivated by the 2D detector array where we used a parallel projection orthogonal to the gamma image plane to project the measured events onto the ultrasound image plane with the help of the relative pose from equation (10.6). The idea is illustrated in Fig. 10.15 which also shows an augmentation

[51] Cf. section 9.2.
[52] Cf. Knoll et al. [224].
[53] Cf. Matthies et al. [278].

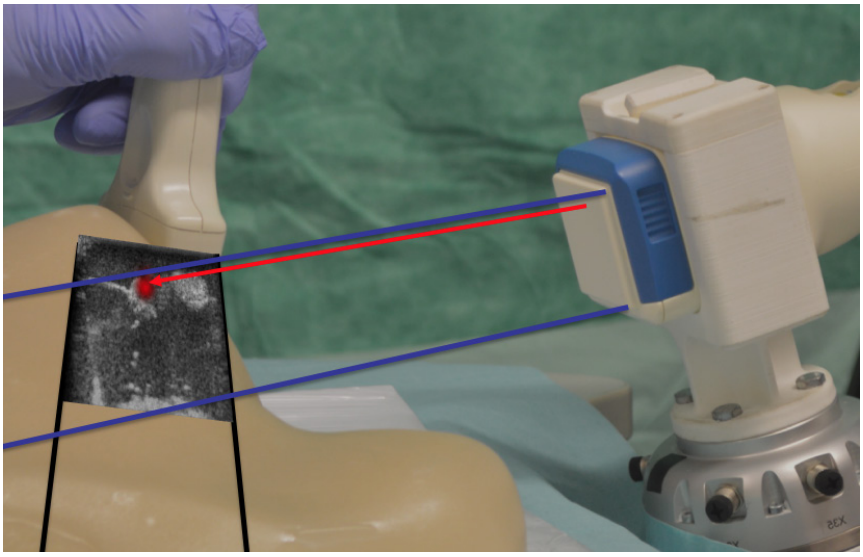


Fig. 10.15. **Projection of gamma camera events onto ultrasound plane.** The anatomical information is examined with an ultrasound probe as a slice through a phantom (left). The field of view of the ultrasound transducer is visualized with black lines. Measurements for various depths are shown with the overlayed US image illustrating two lymph node phantoms (bright blobs). The robotic arm holds the gamma camera (right). It measures the radioactivity with a set of detectors arranged in a 2D grid. We project detected events orthogonal to the grid onto the ultrasound image plane (red arrow) where we visualize them in red. It is visible that one of the nodes contains a radiotracer.

example. We empirically determine a valuable plane-to-plane distance for the relative pose as $d < 15$ cm to make a standard amount of radiotracer clearly visible and set $d = 13$ cm for all our experiments as a trade off between proximity and collision avoidance. A resulting position setup for the respective components is exemplified in our evaluation in Fig. 10.21.

10.3.2.2. Multi-modal Visualization

The events are made visible using the CAMPVis[54] framework with the **visualization pipeline** depicted in Fig. 10.16. The gamma camera is directly attached to the visualization workstation via USB and runs a method to provide the gamma events relevant for the rendering. The latest ultrasound image together with meta data is interfaced via OpenIGTLink from the ultrasound machine. And the relative pose for the co-modality fusion is provided out of ROS. We then augment the events by integration over time on the ultrasound image where we vary the integration time from 0.5 sec to 3 sec depending on the activity decay of our radio isotope. In proportion to the amount of measured events, we adjust the opacity of the augmentation and denoise the gamma signal by cutting off integrated measures below a specified threshold. To add a spatial distribution effect for the events, we apply a 2D Gaussian profile on the projected events that approximately represents the probability for the detected events in space and eases interpretation of the augmented image. The multi-modal visualization with a graphical user interface to manipulate relevant parameters is shown in Fig. 10.17. A consecutive validation evaluates the system.

[54] Cf. Schulte zu Berge et al. [371].

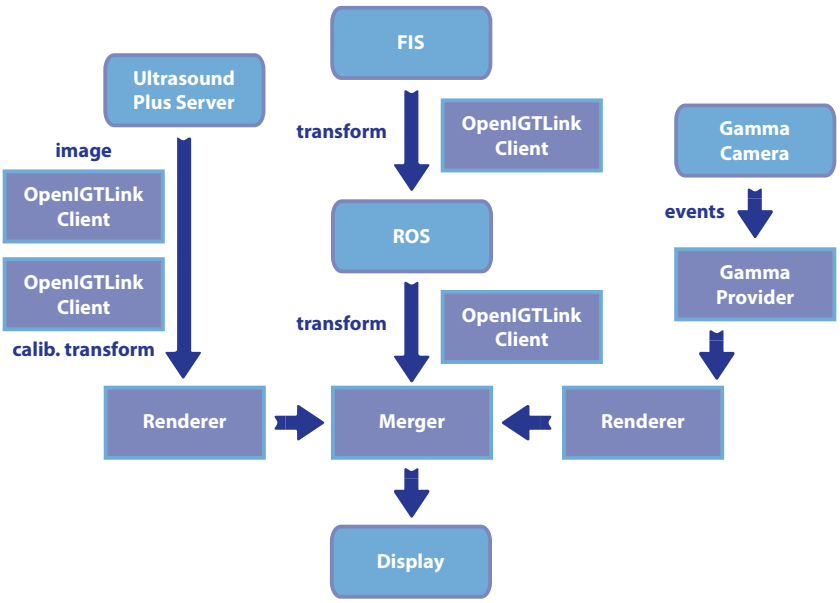


Fig. 10.16. **Visualization pipeline overview for US-Gamma augmentation.** The visualization framework is provided with an ultrasound image and its meta data (left) from an ultrasound server via OpenIGTLink. The poses are also communicated via OpenIGTLink (centre). Tracking is provided by the FRAMOS Imaging Systems (FIS) framework (top) to ROS which gives the necessary relative pose to a merger method. The gamma camera (right) sends events to a gamma provider which prepares the information for the gamma overlay. The merger method ultimately composes the augmented image. The entire pipeline operates in real-time.

Fig. 10.17. **Multi-modal visualization and GUI for spatial US-Gamma fusion.** The graphical user interface allows for runtime manipulation of specific pipeline properties (left). It is shown here that selected properties for the gamma camera rendering such as the integration time and threshold value can be changed during execution. The resulting visualization is shown on the right where the ultrasound image shows two phantom nodes (brighter blobs on top). The line of sight for the gamma camera projected onto the ultrasound image is indicated by a blue area and the radioactivity is augmented in red. The isotope can be localized in the right node.

10.3.3. Experimental Validation

We assess the use and quality of the proposed collaborative robotic assistant in a series of tests. A first test investigates the feasibility of functionality-enhanced ultrasound images in a toy setup which is quantitatively validated in a second study. A third experiment validates the use of the system for sentinel lymph node biopsy in an expert assessment where we compare the classical approach of cognitive fusion against the augmentation with spatial modality fusion on a phantom.

10.3.3.1. Feasibility Study

In a first feasibility study, we leverage a prototype setup to **understand** the **capabilities of joint US-Gamma imaging**.[55] The main goal of this preliminary test is to find the minimum distance for two phantom nodes that allows to distinguish which one of them uses a radiotracer. The experimental setup is illustrated in Fig. 10.18, where two neighbouring lymph nodes are simulated with spherical containers of 1 cm diameter each. One of them (the cold node) is filled with pure water while the other contains a fluid with 0.5 MBq of ^{99m}Tc to simulate a sentinel lymph node with a radiotracer (hot node). Both containers are put in a plastic box which is filled with water for acoustic ultrasound coupling and they are brought gradually closer to each other. As shown in Fig. 10.18, even when both spheres are put next to each other touching each others surface, the hot node can clearly be identified in the Gamma-enhanced ultrasound view indicating a reliable test setup for more complex procedures.

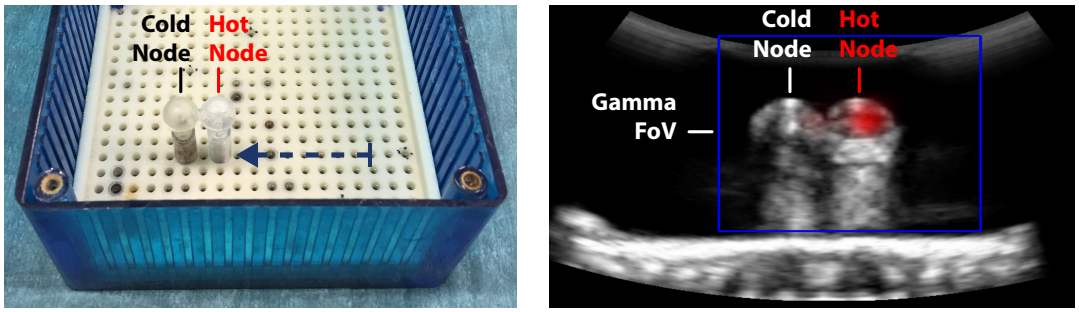


Fig. 10.18. **Feasibility test for multi-modal spatial fusion.** To spherical containers of 1 cm diameter are placed in a box (left) which is filled with water. One of them (hot node) is filled with a fluid with 0.5 MBq of ^{99m}Tc while the other (cold node) is filled with water. They are brought gradually closer to each other as indicated by the blue arrow. An ultrasound scan through the water is performed using a prototype of our system. The resulting visualization of spatial US-Gamma fusion is shown on the right where the Gamma field of view is indicated in blue. Even when the spheres touch each other, the hot node can clearly be distinguished from the cold node by looking at the overlayed radioactivity shown in red.

10.3.3.2. Accuracy Assessment

To **validate the accuracy** of the spatial fusion approach, we use a similar setup as for the time measurements with a second robot in chapter 7.9. We utilize the spheres and the box from

[55] Cf. Esposito et al. [101].

our feasibility study before. A UR5 (Univeral Robots, Odense, Denmark) robotic arm is set up to hold the ultrasound transducer in the experiment to reliably navigate the probe around the phantom in order to quantify the error of the fusion system while our robot assistant follows with the gamma camera. One sphere is filled again with a radiotracer. We use 3 MBq of ^{99m}Tc and navigate the robotic arm manually to various poses in which we acquire ten spatially fused US-Gamma images slicing through the hot node with an event integration time of 2 sec. The target is to compare an annotation of the sphere centre with an annotation of the point of highest radioactive measurement on the ultrasound image and investigate the overall deviation due to calibration and approximation steps.

After starting with measurements in ground position, we translate the ultrasound probe which is slightly immersed in water along the horizontal ultrasound image direction to both sides with a displacement of up to 21 mm compared to the ground position such that the sphere is still part of the ultrasound image and our robotic assistant does not collide with the box phantom. At the extreme points of the translational motion, we acquire ten joint images each. Then rotations along all three coordinate axes are performed where the ultrasound transducer is rotated clockwise and counterclockwise around the coordinate axes with a deviation of up to 22.4°. A total of 60 measurements are taken at the extreme positions. We note that the used robot poses reflect extreme positions of the ultrasound transducer beyond motions an examiner would require. The anatomical centres and points of highest activity of all measurements are annotated and the metric deviation is calculated from the pixel distances. We measure an average error of 0.7 mm in ground position. Table 10.1 summarizes the results under individual variation of translation and rotation. Over all tests, the average error can be reported as 1.12±0.57 mm with a median error of 1.00 mm. This is significantly lower than the minimum threshold of 5 mm for lymph nodes visibility over which the anatomical structure can be reliably detected in patients.[56] We therefore conclude that the influence of the system error is small enough to allow for an expert assessment with a synthetic breast phantom.

Variation in	Translation	Rotation
Average Error	1.05 mm	1.15 mm

Tab. 10.1. **Average deviation for multi-modal augmentation.** Compared is the average error under translation and rotation of the ultrasound probe.

10.3.3.3. Expert Validation with Biopsy

After the preliminary tests, we analyse an **expert assessment** with a biopsy phantom for which we ask five medical experts to perform a **punch needle biopsy on a phantom**. The subjects involved in the study are two professional gynaecologist and three independent medical researchers. We prepare two phantoms with synthetic hot and cold nodules and consecutively ask for a punch needle biopsy under functional and anatomical guidance. A first run is done with cognitive spatial fusion of the modalities where a handheld gamma camera and an ultrasound probe are used and a second run is performed with the help of our collaborative robot. We quantitatively compare both runs following a specified protocol.

[56] Cf. Obwegeser et al. [313].

The **two phantoms** are formed to represent the anatomy of the female upper torso around the axilla region. We build the phantoms out of a mixed gelatine-agar material[57] and put 11 (respectively 12 synthetic nodules) inside at different locations and distances. The reason for setting up two different phantoms is that we want to test the methods in two independent runs where it is not possible for the subjects to remember the nodule locations. To reliably simulate a haptic difference, we choose a higher concentration of agar-gelatine for the nodules and add a radiotracer to 4 (respectively 5) of them with 3 MBq of ^{99m}Tc each. Additionally, we mark the nodules with food colour to distinguish the subtracted tissue from the punch needle biopsy. We mark cold nodules with blue and hot nodules with red colour and insert them at a subcutaneous depth of approximately 2 cm for a realistic setup. We cut open one prototype phantom after the experiment and show it together with an example biopsy of a cold node in Fig. 10.19. In Fig. 10.20 a SPECT-CT scan of one of the phantoms is shown where the active nodes can be clearly distinguished.

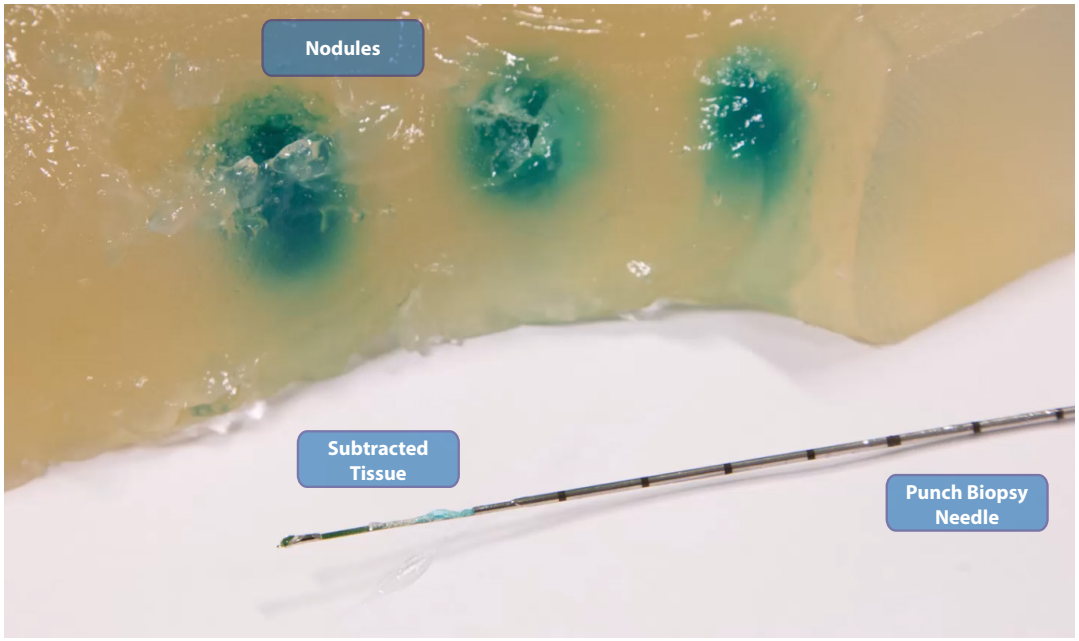


Fig. 10.19. **Open phantom with three cold nodules and a punch biopsy example.** We biopsied a prototype phantom with a punch biopsy needle (bottom) and cut it open for illustration of the three visible nodules (top). The subtracted phantom tissue leaves a visible blue colour in the open punch needle compartment (centre).

In the **first run**, each medical expert is given a gamma camera and a handheld ultrasound transducer to examine the first phantom. A punch biopsy needle for tissue extraction is also provided. Gamma and ultrasound images are visualized live on two different screens and a **cognitive spatial fusion** of the modalities is necessary in order to distinguish hot from cold nodules after finding the relevant anatomical structure. In order to free a hand, all subjects decided to perform the biopsy under US-guidance only while using the second hand for the biopsy. Our collaborative robot is used in a **second run** where it holds the gamma camera and the functional information is blended in our visualization framework on top of the ultrasound image on a single screen. The biopsy is then performed under **multi-modal guidance**.

[57] Cf. Dang et al. [79].

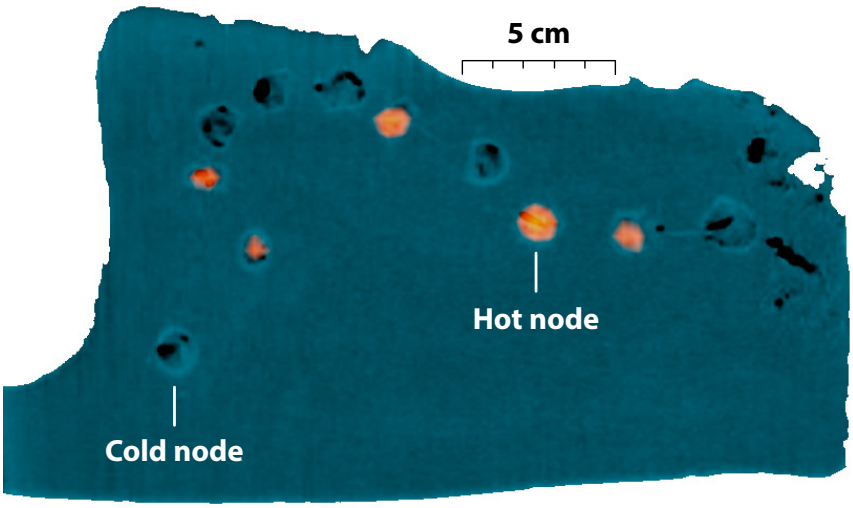


Fig. 10.20. SPECT-CT scan of breast-axilla phantom with lymph nodes. A SPECT-CT scan is shown for one of the phantoms with 12 lymph nodes. The 5 hot nodes (orange) can be clearly seen while the remaining ones are slightly indicated by a corona and density variations.

For all tests, the experts are given time to get familiar with the system and we initiate the experiments when the subject indicates to be ready to start. We then ask them to count the number of visible lymph nodes and identify hot nodes. Finally, a biopsy of a potentially hot node is performed which the expert can freely choose. We validate the outcome of the biopsy with the food colour and record the inspection time and the biopsy duration. Additionally, the subjective confidence for each hot node identification is put to protocol and we ask for a individual difficulty assessment of the entire procedure one and two where both values are measured on a scale from 0 to 5. The results are summarized in table 10.2 while Fig. 10.21 depicts the dimensions of the entire setup at the start of the second run.

Procedure	Cognitive Fusion	Our System	Improvement
Inspection Time [min : sec]	13 : 13	8 : 55	+32.5%
Identified Nodes on US	87.3%	85.0%	−2.3%
Correct Hot Nodes	75.0%	88.0%	+13.0%
False Positive Rate	43.3%	0%	+43.3%
Confidence	86.2%	88.0%	+1.8%
Biopsy Time [min : sec]	5 : 12	2 : 46	+46.8%
Success Rate	0%	80%	+80.0%
Subjective Ease-of-Use	1.8/5	4.4/5	+52.0%

Tab. 10.2. Average results of expert user biopsy. The table compares the result of our expert study with anatomical assessment and punch needle biopsy on a breast phantom for cognitive modality fusion versus multi-modal fusion with collaborative robotic assistant. Note the significant improvement of the procedure time and accuracy.

Besides a significantly faster identification of the nodes, also the biopsy time is improved by a large margin with the collaborative spatial fusion while at the same time the **accuracy of the**

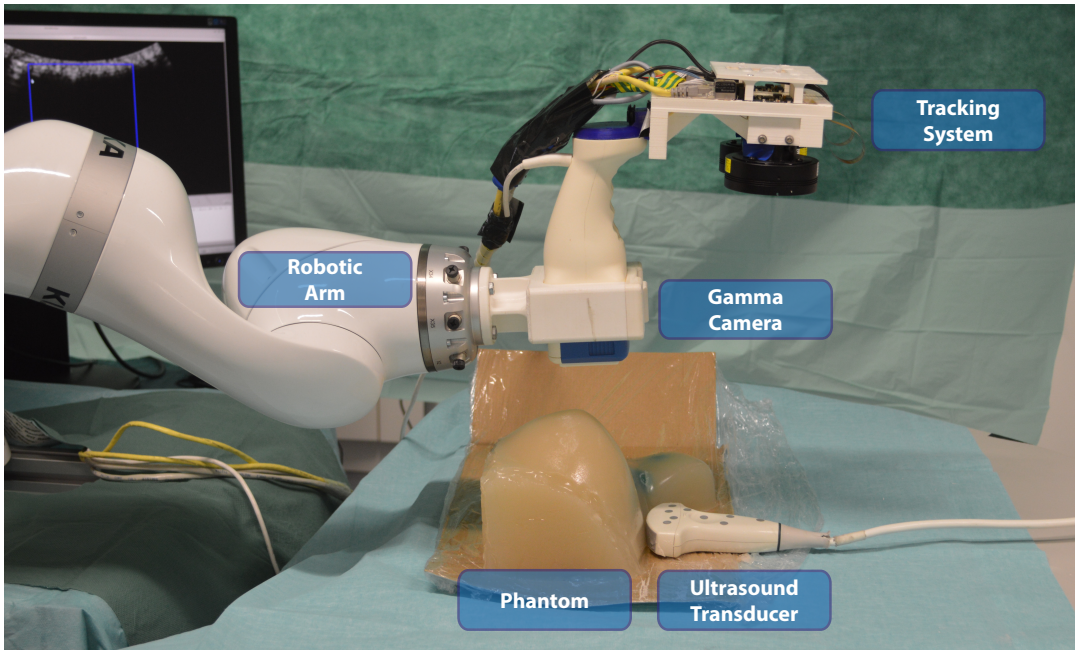


Fig. 10.21. **Setup of involved hardware components for expert validation.** At the start of the second run, the medical expert is given an ultrasound transducer to examine the phantom (bottom) while the robotic arm (left) assists the intervention by holding both the tracking system (top) and the gamma camera (centre) above the examined anatomy.

assessment increases. Summing the time spent for inspection and biopsy, we note that a total of 6 min 44 sec is saved in average with the novel procedure which is a **time improvement** of 36.6%. Moreover, no single node was incorrectly described as a hot sentinel node with the robotic setup. Besides the subjective simplification of the procedure with the assistant, the objective success rate of the biopsy improved by 80%. It is worth noting that the only incorrect biopsy was caused by a system failure where the robot reached its joint limits during the preparation for the biopsy by the subject. The medical expert then decided to perform the biopsy under ultrasound guidance only. This let to a confusion of two close nodes which resulted in a biopsy of the cold node. The fact that no punch needle biopsy in the first run could correctly subtract tissue from a hot node indicates the difficulty of the cognitive fusion process. The high confidence level for the hot node identification process paired with the false positive rate is alarming from a diagnostic perspective as it can ultimately lead to an incorrect treatment of early stage metastases.

Overall, this case study shows that **live spatial fusion of functional and anatomical data can have an impact on the quality and efficiency of medical procedures**. Accurate poses can help to reach a correct spatial alignment even in complex calibration chains as we have seen by the feasibility study. The second experiment revealed that the accuracy can be maintained even under changing poses of the ultrasound probe. The impact of the decrease for false positives and the encouraging feedback from the medical experts motivate to explore the procedure not only in this controlled synthetic environment but also in a larger medical study with patients being involved. An interesting question could be to assess the learning curve of novice users in this context. We believe that the intuitive use of the system that made it directly usable by all our subjects is a great factor for wider acceptance. A flexible real-time tracking system that can deal with partial marker occlusion usually present in handheld

scans is essential for this and can ultimately effect the impact on other medical treatments where the current hardware may still be too bulky. A reliable system can also enable safe robotic collaboration in the absence of a robotic expert. The significant improvements for complex spatial fusion systems, however, come with the burden of a large calibration overhead. Exploring alternative calibration procedures that do not require time-consuming preparation with specified calibration targets or investigations to update and maintain the calibration quality with online measurements seem a promising direction. Aside of an improved ergonomic handling, one can investigate miniaturization of the tracking design similar to the re-design approaches studied in chapter 7.8 and attach some needle guidance tool to simplify the biopsy procedure. A further step towards a safer and more intelligent robotic collaborator could also include a feature extension for spatial awareness through multi-view depth estimation that helps to avoid collisions.

With these case studies, we have seen that accurate real-time optical tracking systems can have a significant impact on practical problems in different sectors where time synchronization, pose interpolation and denoising play a substantial role. We want to conclude this thesis with a brief retrospective of the subjects touched here and a short summary of open problems in the field that are of applied and theoretical interest to indicate some potential future directions.

Part V

Conclusion & Outlook

11 Prospects

> *There is no real ending.*
> *It's just the place*
> *where you stop the story.*
>
> **– Frank Herbert**

We conclude this thesis with a brief summary and meditation about the proposed pipelines and results before we dialectically discuss their limitations and look ahead to potential solutions and interesting research directions to overcome these restrictions. A final thought on the practical relevance of 6D pose estimation for interdisciplinary projects in the future finishes our journey.

11.1. Retrospective

After an initial discussion of the basic hardware and software concepts used in computer vision systems (chapter 2), we detailed the steps of a **2D image processing** pipeline to robustly and reliably detect circular markers in an image with sub-pixel precision (chapter 3). We leveraged the algorithm in an optimization framework for accurate camera calibration (chapter 4) and described the fundamental concepts and training for data driven approaches with neural networks (chapter 5).

We then left the 2D image plane in chapter 6 and focused on 3D computer vision. After laying the foundation to describe 6D poses with dual quaternions, we further investigated the differential geometric aspects of the parameter space and reviewed classical and convolutional approaches for stereoscopic depth estimation with probabilistic concepts. Accurate binocular triangulation of markers served as the input for a robust optical tracking algorithm based on dual quaternions which we formulated and validated in chapter 7. The real-time optical tracking system (OTS) leverages a flexible online marker training which was prototyped with self-adhesive retro-reflective markers and IR illumination. We instantiated the hardware in two systems, an outside-in and a camera-in-hand OTS. A medical validation was done with the latter using a collaborative robot for movement therapy. Consecutively, we went one step further to markerless pose estimation (chapter 8) where we analyzed a miniaturized stereo system for inside-out tracking with SLAM in the operating room and consecutively redefined the standard paradigm for classical **6D pose estimation**. While detection or regression are common approaches to solve the problem, we leverage machine learning to estimate a next

best pose and reformulate the problem as a decision process. Using a reinforcement signal from a small neural network to move a virtual object gradually closer to an observation allowed to reach highly accurate monocular 6D poses even for unseen objects that were not used during the synthetic-only training.

Our final part focused on the poses of optical 6D estimation systems (chapter 9) and their application in practice (chapter 10). We first examine ways to efficiently interpolate poses to estimate displacements between measurements and pragmatically extrapolate future motion to decrease lags in real-time pipelines and to synchronize systems. We effectively use the Riemannian structure of pose space for denoising of estimation sequences in the presence of outlier measurements before we combine these concepts for **sensor fusion**.
To illustrate the wide applicability of the developed concepts, we chose three orthogonal applications where multiple modalities are combined with different goals in an industrial, medical and robotics setup. With our optical tracking system and the necessary accurate pose descriptions, we managed to showcase solutions to all three problems despite intricate co-calibration requirements.

11.2. Limitations & Future Directions

Even though we see a current trend towards more general forms of 6D pose estimation[1] where the **displacement of unseen object instances** can be measured, it remains an open problem to accurately estimate the poses of objects not present during training. These methods already generalize to different object instances of the same class, such as a laptop with a slightly different shape or a cup that has another form. However, they fail for completely new objects.
Our markerless method (chapter 8) provides a solution to track objects of an unknown class that are entirely different from the training examples as we change the pose estimation paradigm and learn a decision process comparing between a 3D model and an observation. The pose accuracy for known objects, however, is still superior compared to unseen examples and the requirements of a 3D model remains. It would be interesting to see investigations where the learning and testing processes are entangled in a joint dynamic system that refines, adjusts and updates a 3D model with current measurements aiming towards CAD-free pose estimation of unknown objects with high accuracy. One could try to generate the information and the updates even synthetically and save them directly in the 3D model files as variable appearance signatures using spatial descriptors. These ideas ultimately blend between the disciplines of object pose estimation and spatial reconstruction.
A simple online training step for the model is already part of our marker-based optical tracking algorithm (chapter 7) where we see a significant gap between marker approaches and markerless methods although RGB images provide many more pixel measurements. Even though the monocular methods without markers are very robust, this shows that the descriptive power of current pipelines still falls short compared to marker approaches. We believe that this gap will trigger a series of further investigations in the future **bringing markerless methods closer to** the pose estimation performance currently possible with **markers**.

[1] Cf. Wang et al. [443].

The core tracking backbone we designed in chapter 7 allows to relax its tolerance towards **deformations**. We demonstrated this flexibility with a use case for robot movement therapy. While this led to the necessary results for the application, the core pose description is still rigid and could be further explored towards **non-rigid motions** with higher degree of freedom that are also capable to describe motions of objects that change shape with different dynamics. This is of severe practical relevance as full rigidity is an approximation that only holds for certain objects and with human interaction, object appearance and geometry can change. The same dynamics hold true for the inside-out approaches discussed in chapter 8. While our intrinsic assumption is a mostly rigid environment, the situation can drastically change not only in an operating room, where people and tools move, but also beyond these scenes. Defining a fixed world anchor for SLAM approaches in **dynamic environments** is difficult and most methods nowadays either ignore non-rigid motion[2] or specifically model non-rigid object deformations.[3] This leaves space to explore joint approaches that combine the benefits of both worlds while still being able to run efficiently with the aim of camera self-localization.

With the **availability of depth** sensors even on mobile devices, it can also be an interesting direction to explore the geometric nature of scenes and their topology more thoroughly. At the moment, our pose pipelines fully rely on monocular or stereoscopic images where the designed pipelines triangulate point clouds either from markers or 2D features. An acquisition of full depth maps either directly with a depth camera or indirectly following multi-view cues could be leveraged to make the algorithm more aware of the environment or enable the use of geometric constraints for self-supervision through cycle consistencies. An outcome can be a data-driven approach which is free from the synthetic training where ground truth is required. Moreover, depth information has a direct impact on pose estimation applications in augmented reality where not only more realistic light changes can benefit from scene geometry understanding, but we can also use geometric cues for occlusion aware augmentations in mixed reality pipelines.
Applications can further benefit from exploring and improving the **efficiency and runtime** of the proposed pipelines. While we managed to design algorithms that run on a laptop (chapter 8) and improved the sampling rate of pose pipelines (chapter 9.1), real applications can further benefit from processing that reduces the interface lag by moving the computation closer to the edge. One direction could be to explore the possibility for pose estimation algorithms to either run on dedicated hardware such as an ASIC or FPGA or leverage embedded and mobile vision platforms with neural processing units.[4]

The sensor fusion approaches we consider in chapter 10 focus around spatial fusion of **multiple modalities**. Accurate fusion and the pose improvement strategies from chapter 9.3 enable a set of novel applications and improved augmentations where we use the estimated poses to combine different modalities. At the core, the myriad of sensing devices not only in medical applications but also in general multi-sensor setups is highly interconnected. From our point of view, a further explicit exploration of this **interconnectivity** can solve a set of problems and improve pose estimation accuracy and reliability far beyond what is possible with individual sensor treatment which can result in better modality combination.
A direction could be to aim for **global consistency** where all sensors in a specific environment contribute data and we look for plausible consistency cycles among them. An inside-out device

[2] Cf. Bescos et al. [22].
[3] Cf. Bozic et al. [36].
[4] Cf. Dinelli et al. [88].

that suffers from drift during its temporal self-localization can then benefit from a stable global outside-in tracker that observes its pose in particular displacements while another modality such as an ultrasound device that explores an anatomy must see similar structures if it is held in similar poses. Such a framework – if modeled probabilistically – can then be optimized not only amongst different sensors but also across different modalities ultimately outputting better individual poses. One can then retrieve the pose between a specific sensor and a target without direct line-of-sight contact even along an indirect path in a pose graph that does not involve this direct edge. Knowing and modeling the reliability of specific sensors for the task at hand then contributes in better overall poses. An IMU can for instance greatly improve the rotation accuracy measured by an outside-in tracker that observes the object it is attached to while its translation estimates may be erroneous. These could then again benefit from the measurements of outside-in tracking consequently leading to a better pose.

A future intelligent interaction between such devices in the OR could also greatly benefit the semantic understanding of the surrounding where tools provide their data to a global system that leverages poses and images to optimize for a specific task. While fully **autonomous understanding of the surrounding** actions may be a far fetched goal, a simpler framework could aim for improved self-localization with multi-modal sensor fusion for interactive use in collaborative environments. A robotic manipulator could thereby actively decide for the optimal position of tracking systems or a single camera to retrieve the best environment map for orientation or the best reconstruction of an object of interest by moving the camera or another sensor autonomously at specific locations that benefit the global understanding of the scene.

Such an intelligent, holistic environment and self-understanding can then not only benefit the poses measured but could lead to significant practical advantages. In all our implementations throughout this thesis, co-calibration was always necessary between sensors and we moved various calibration targets to different locations running a variety of hand-eye, tool, intrinsic and extrinsic calibration routines. If we have a **shared common map** that is distributed and optimized among the entire swarm of sensors, we can then also explore auto-calibration setups via this map where the calibration constraints could be deduced from the map itself containing scene geometry and appearance. Furthermore, re-calibration and online adjustments via such a shared and potentially multi-modal map can be investigated.

11.3. Epilogue

Rigid object pose estimation is an essential task for robotics, automation as well as augmented and mixed reality applications. We have used computer vision systems and explored the underlying mathematical and algorithmic concepts for pose computation in single and multi-modal setups that enabled high performance 3D vision systems. These systems can be used across scientific disciplines which we exemplified in various medical setups. Marker-based approaches benefit from the fundamental concepts of multi-view geometry and the precise mathematical description of camera sensing. Such pipelines can lead to highly accurate pose estimation results while data-driven algorithms can be the tool of choice to make these estimations robust and adjustable to specific objects. We strongly believe that a **synthesis of geometric knowledge with neural network support** can lead to significant improvements in the domain in the

future. The proposed pose treatments and changed tracking paradigm presented in this thesis can be a solid basis to further explore ideas in this direction while we think that the medical use cases can be a motivation to have not only scientific but also social impact. Moreover, the developed hardware and software that lead to the optical tracking system can be a baseline for future research.

During all our case studies, we noticed that current setups often suffer from the requirement of expert knowledge or need a multidisciplinary team in practice to be realized. An exploration of auto-calibration and map sharing of sensing tools can contribute towards a democratization of these technologies at the interface between academic and practical disciplines.

Part VI

Appendix

A

Mathematical Derivations & Complementary Results

A.1. Dual Quaternion Energy Functional

The energy functional for the pose estimation from section 7.3.7 can be rewritten in terms of quaternions. For this purpose it is beneficial to analyse the quaternion matrices

$$\mathbf{P}(\mathbf{r}) = \begin{pmatrix} r_4 & -r_3 & r_2 & r_1 \\ r_3 & r_4 & -r_1 & r_2 \\ -r_2 & r_1 & r_4 & r_3 \\ -r_1 & -r_2 & -r_3 & r_4 \end{pmatrix} \quad \text{and} \quad \mathbf{W}(\mathbf{r}) = \begin{pmatrix} r_4 & r_3 & -r_2 & r_1 \\ -r_3 & r_4 & r_1 & r_2 \\ r_2 & -r_1 & r_4 & r_3 \\ -r_1 & -r_2 & -r_3 & r_4 \end{pmatrix} \tag{A.1}$$

from (7.31) in more detail. We note that for two arbitrary quaternions $\mathbf{r}, \mathbf{s} \in \mathbb{H}$ it holds

$$\mathbf{P}(\mathbf{r})\,\mathbf{s} = \mathbf{W}(\mathbf{s})\,\mathbf{r} \tag{A.2}$$

$$\mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{W}(\mathbf{r}) = \mathbf{W}(\mathbf{r})\mathbf{W}(\mathbf{r})^{\mathrm{T}} = \mathbf{r}^{\mathrm{T}}\mathbf{r}\,\mathbf{I} \tag{A.3}$$

$$\mathbf{P}(\mathbf{r})\mathbf{W}(\mathbf{s})^{\mathrm{T}} = \mathbf{W}(\mathbf{s})^{\mathrm{T}}\mathbf{P}(\mathbf{r}) \tag{A.4}$$

with the identity matrix $\mathbf{I} \in \mathbb{R}^{4\times4}$.

Leaving out the constant, these equations can simplify the energy functional from equation (7.33) for the unit dual quaternion $\mathbf{Q} = \mathbf{r} + \varepsilon\mathbf{s}$ with

$$\mathrm{E}(\mathbf{r},\mathbf{s}) = \sum_{j,k} \mathrm{m}_{jk} \left\| y_k - \left(\mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{P}(\mathbf{r})x_j + \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s}\right)\right\|^2 \tag{A.5}$$

$$= \sum_{j,k} \mathrm{m}_{jk} \left(y_k - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s} - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{P}(\mathbf{r})x_j\right)^{\mathrm{T}} \left(y_k - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s} - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{P}(\mathbf{r})x_j\right) \tag{A.6}$$

$$\overset{(A.2)}{=} \sum_{j,k} \mathrm{m}_{jk} \left(y_k - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s} - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{W}\left(x_j\right)\mathbf{r}\right)^{\mathrm{T}} \left(y_k - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s} - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{W}\left(x_j\right)\mathbf{r}\right) \tag{A.7}$$

$$= \sum_{j,k} \mathrm{m}_{jk} \left(y_k^{\mathrm{T}} - \mathbf{s}^{\mathrm{T}}\mathbf{W}(\mathbf{r}) - \mathbf{r}^{\mathrm{T}}\mathbf{W}\left(x_j\right)^{\mathrm{T}}\mathbf{W}(\mathbf{r})\right) \left(y_k - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s} - \mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{W}\left(x_j\right)\mathbf{r}\right) \tag{A.8}$$

$$= \sum_{j,k} \mathrm{m}_{jk} \left(y_k^{\mathrm{T}}y_k - y_k^{\mathrm{T}}\mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s} - y_k^{\mathrm{T}}\mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{W}\left(x_j\right)\mathbf{r} - \mathbf{s}^{\mathrm{T}}\mathbf{W}(\mathbf{r})y_k + \mathbf{s}^{\mathrm{T}}\mathbf{W}(\mathbf{r})\mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s}\right. \tag{A.9}$$

$$+\mathbf{s}^{\mathrm{T}}\mathbf{W}(\mathbf{r})\mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{W}\left(x_j\right)\mathbf{r} - \mathbf{r}^{\mathrm{T}}\mathbf{W}\left(x_j\right)^{\mathrm{T}}\mathbf{W}(\mathbf{r})y_k + \mathbf{r}^{\mathrm{T}}\mathbf{W}\left(x_j\right)^{\mathrm{T}}\mathbf{W}(\mathbf{r})\mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{s} \tag{A.10}$$

$$\left.+\mathbf{r}^{\mathrm{T}}\mathbf{W}\left(x_j\right)^{\mathrm{T}}\mathbf{W}(\mathbf{r})\mathbf{W}(\mathbf{r})^{\mathrm{T}}\mathbf{W}\left(x_j\right)\mathbf{r}\right) \tag{A.11}$$

$$\overset{(A.2)}{\underset{(A.3)}{=}} \sum_{j,k} m_{jk} \left(y_k^T y_k - \mathbf{r}^T \mathbf{P}(y_k)^T \mathbf{s} - \mathbf{r}^T \mathbf{P}(y_k)^T \mathbf{W}(x_j)\mathbf{r} - \mathbf{s}^T \mathbf{P}(y_k)\mathbf{r} + \mathbf{s}^T \mathbf{r}^T \mathbf{r}\, \mathbf{I}\, \mathbf{s} \right. \tag{A.12}$$

$$+ \mathbf{s}^T \mathbf{r}^T \mathbf{r}\, \mathbf{I}\, \mathbf{W}(x_j)\mathbf{r} - \mathbf{r}^T \mathbf{W}(x_j)^T \mathbf{P}(y_k)\mathbf{r} + \mathbf{r}^T \mathbf{W}(x_j)^T \mathbf{r}^T \mathbf{r}\, \mathbf{I}\, \mathbf{s} \tag{A.13}$$

$$\left. + \mathbf{r}^T \mathbf{W}(x_j)^T \mathbf{r}^T \mathbf{r}\, \mathbf{I}\, \mathbf{W}(x_j)\mathbf{r} \right). \tag{A.14}$$

Using the constraint $\mathbf{r}^T\mathbf{r} = 1$ of the dual quaternion and equation (A.3) again gives further simplifications such that

$$\mathrm{E}(\mathbf{r}, \mathbf{s}) = \sum_{j,k} m_{jk} \left(y_k^T y_k - \mathbf{r}^T \mathbf{P}(y_k)^T \mathbf{s} - \mathbf{r}^T \mathbf{P}(y_k)^T \mathbf{W}(x_j)\mathbf{r} - \mathbf{s}^T \mathbf{P}(y_k)\mathbf{r} + \mathbf{s}^T \mathbf{s} \right. \tag{A.15}$$

$$\left. + \mathbf{s}^T \mathbf{W}(x_j)\mathbf{r} - \mathbf{r}^T \mathbf{W}(x_j)^T \mathbf{P}(y_k)\mathbf{r} + \mathbf{r}^T \mathbf{W}(x_j)^T \mathbf{s} + \mathbf{r}^T x_j^T x_j \mathbf{I}\, \mathbf{r} \right) \tag{A.16}$$

$$= \sum_{j,k} m_{jk} \left(\mathbf{s}^T \mathbf{s} - \mathbf{r}^T \left(\mathbf{P}(y_k)^T \mathbf{W}(x_j) + \mathbf{W}(x_j)^T \mathbf{P}(y_k) \right) \mathbf{r} \right. \tag{A.17}$$

$$\left. + \mathbf{s}^T \left(\mathbf{W}(x_j) - \mathbf{P}(y_k) \right) \mathbf{r} + \mathbf{r}^T \left(\mathbf{W}(x_j)^T - \mathbf{P}(y_k)^T \right) \mathbf{s} + x_j^T x_j + y_k^T y_k \right) \tag{A.18}$$

$$\overset{(A.4)}{=} \sum_{j,k} m_{jk} \left(\mathbf{s}^T \mathbf{s} - \mathbf{r}^T \left(\mathbf{P}(y_k)^T \mathbf{W}(x_j) + \mathbf{P}(y_k) \mathbf{W}(x_j)^T \right) \mathbf{r} \right. \tag{A.19}$$

$$\left. + 2\mathbf{s}^T \left(\mathbf{W}(x_j) - \mathbf{P}(y_k) \right) \mathbf{r} \right) + D \tag{A.20}$$

$$= \sum_{j,k} m_{jk} \left(\mathbf{s}^T \mathbf{s} - 2\mathbf{r}^T \mathbf{P}(y_k)^T \mathbf{W}(x_j)\mathbf{r} + 2\mathbf{s}^T \left(\mathbf{W}(x_j) - \mathbf{P}(y_k) \right) \mathbf{r} \right) + D \tag{A.21}$$

with the constant value $D \in \mathbb{R}$.

A.2. Additional YCB Comparison

Additional to Tab. 8.1, where the standard ADD metric[1] is analysed, we compare in Tables A.1, A.2 and A.3 the area under the ADD threshold curve (AUC) for varying absolute thresholds from zero to 0.1m in direct comparison with other pipelines.[2] In line with the previous results, our method compares favourable with respect to this metric on the standard benchmark (Ours OS) and significantly better with the shift-correction.

[1] Cf. Hinterstoisser et al. [174].
[2] Cf. Xiang et al. [454].

Model	3DC [454]	PC [454]	CPC [63]	PRBPF [82]	Ours OS	+ Shift
002_master_chef_can	12.30	50.90	62.32	63.30	65.61	91.15
003_cracker_box	16.80	51.70	66.69	77.80	84.34	90.74
004_sugar_box	28.70	68.60	67.19	79.60	78.43	91.05
005_tomato_soup_can	27.30	66.00	75.52	73.00	66.83	76.06
006_mustard_bottle	25.90	79.90	83.79	84.70	86.05	94.03
007_tuna_fish_can	5.40	70.40	60.98	64.20	65.90	69.12
008_pudding_box	14.90	62.90	62.17	64.50	79.00	83.01
009_gelatin_box	25.40	75.20	83.84	83.00	82.92	92.78
010_potted_meat_can	18.70	59.60	65.86	51.80	75.21	79.44
011_banana	3.20	72.30	37.74	18.40	84.99	90.19
019_pitcher_base	27.30	52.50	62.19	63.70	85.14	94.22
021_bleach_cleanser	25.20	50.50	55.14	60.50	89.27	90.68
024_bowl	2.70	6.50	3.55	28.40	85.89	87.03
025_mug	9.00	57.70	45.83	77.90	78.95	87.83
035_power_drill	18.00	55.10	76.47	71.80	76.56	91.95
036_wood_block	1.20	31.80	0.12	2.30	48.62	53.52
037_scissors	1.00	35.80	56.42	38.70	79.78	83.99
040_large_marker	0.20	58.00	55.26	67.10	73.27	75.31
051_large_clamp	6.90	25.00	29.73	38.30	56.09	65.97
052_extra_large_clamp	2.70	15.80	21.99	32.30	67.31	78.06
061_foam_brick	0.60	40.40	51.80	84.10	86.52	86.70
Average	13.02	51.74	53.55	58.35	76.03	83.47

Tab. A.1. Evaluation on the YCB dataset with our object-specific models, AUC01. We compare the area under the ADD threshold curve (AUC) for varying thresholds from zero to 0.1m. Symmetric objects are shown in italic letters.

Model	RKF [349]	HM [312]	R&C [328]	Dope [419]	Ours OS	+ Shift
002_master_chef_can	54.60	81.90	76.70	-	65.61	91.15
003_cracker_box	57.60	83.60	82.90	55.90	84.34	90.74
004_sugar_box	84.10	82.10	86.40	75.70	78.43	91.05
005_tomato_soup_can	68.30	79.80	57.40	76.10	66.83	76.06
006_mustard_bottle	79.00	91.50	86.70	81.90	86.05	94.03
007_tuna_fish_can	43.50	48.70	69.70	-	65.90	69.12
008_pudding_box	50.30	90.20	68.80	-	79.00	83.01
009_gelatin_box	74.80	93.70	73.00	-	82.92	92.78
010_potted_meat_can	50.30	79.10	74.60	39.40	75.21	79.44
011_banana	8.20	51.70	68.80	-	84.99	90.19
019_pitcher_base	77.80	69.40	83.80	-	85.14	94.22
021_bleach_cleanser	59.30	76.20	78.30	-	89.27	90.68
024_bowl	-	3.60	1.50	-	85.89	87.03
025_mug	69.10	53.90	57.90	-	78.95	87.83
035_power_drill	71.40	82.90	81.50	-	76.56	91.95
036_wood_block	-	0.00	0.00	-	48.62	53.52
037_scissors	-	65.30	75.40	-	79.78	83.99
040_large_marker	-	56.50	59.80	-	73.27	75.31
051_large_clamp	-	57.20	75.30	-	56.09	65.97
052_extra_large_clamp	-	23.60	20.40	-	67.31	78.06
061_foam_brick	-	32.10	37.00	-	86.52	86.70
Average	60.59	62.05	62.66	65.80	76.03	83.47

Tab. A.2. **Evaluation on the YCB dataset with our object-specific models, AUC02.** We compare the area under the ADD threshold curve (AUC) for varying thresholds from zero to 0.1m. Symmetric objects are shown in italic letters.

Model	HMP [125]	MT [442]	D-IM [251]	PV-N [326]	Ours OS	+ Shift
002_master_chef_can	75.80	62.70	71.20	81.60	65.61	91.15
003_cracker_box	78.00	80.90	83.60	80.50	84.34	90.74
004_sugar_box	76.50	83.80	94.10	84.90	78.43	91.05
005_tomato_soup_can	72.10	60.40	86.10	78.20	66.83	76.06
006_mustard_bottle	78.90	85.10	91.50	88.30	86.05	94.03
007_tuna_fish_can	51.60	75.40	87.70	62.20	65.90	69.12
008_pudding_box	85.60	17.70	82.70	85.20	79.00	83.01
009_gelatin_box	86.70	79.90	91.90	88.70	82.92	92.78
010_potted_meat_can	70.10	55.00	76.20	65.10	75.21	79.44
011_banana	47.90	59.60	81.20	51.80	84.99	90.19
019_pitcher_base	71.80	96.10	90.10	91.20	85.14	94.22
021_bleach_cleanser	69.10	89.40	81.20	74.80	89.27	90.68
024_bowl	-	49.50	8.60	-	85.89	87.03
025_mug	43.40	87.70	81.40	81.50	78.95	87.83
035_power_drill	76.80	96.40	85.50	83.40	76.56	91.95
036_wood_block	-	43.80	60.00	-	48.62	53.52
037_scissors	42.90	60.20	60.90	54.80	79.78	83.99
040_large_marker	47.60	87.50	75.60	35.80	73.27	75.31
051_large_clamp	-	90.70	48.40	-	56.09	65.97
052_extra_large_clamp	-	88.10	31.00	-	67.31	78.06
061_foam_brick	-	26.30	35.90	-	86.52	86.70
Average	67.18	70.30	71.66	74.25	76.03	83.47

Tab. A.3. Evaluation on the YCB dataset with our object-specific models, AUC03. We compare the area under the ADD threshold curve (AUC) for varying thresholds from zero to 0.1m. Symmetric objects are shown in italic letters.

B

List of Authored & Co-authored Publications

2020

[52] **Benjamin Busam**, Hyun Jun Jung, and Nassir Navab. "I Like to Move It: 6D Pose Estimation as an Action Decision Process". *[arXiv preprint], arXiv:2009.12678, 2020*.

[362] Mahdi Saleh, Shervin Dehghani, **Benjamin Busam**, Nassir Navab, and Federico Tombari. "Graphite: GRAPH-Induced feaTure Extraction for Point Cloud Registration". *International Conference on 3D Vision (3DV), 2020, Fukuoka, Japan (virtual)*. [**Oral Presentation**].

[261] Adrian Lopez-Rodriguez, **Benjamin Busam**, and Krystian Mikolajczyk. "Project to Adapt: Domain Adaptation for Depth Completion from Noisy and Sparse Sensor Data". *Asian Conference on Computer Vision (ACCV), 2020, Kyoto, Japan (virtual)*. [**Oral Presentation** and **Best Student Paper Award**].

[14] Axel Barroso-Laguna, Yannick Verdie, **Benjamin Busam**, and Krystian Mikolajczyk. "HDD-Net: Hybrid Detector Descriptor with Mutual Interactive Learning". *Asian Conference on Computer Vision (ACCV), 2020, Kyoto, Japan (virtual)*.

[358] Patrick Ruhkamp, Ruiqi Gong, Nassir Navab, and **Benjamin Busam**. "DynaMiTe: A Dynamic Local Motion Model with Temporal Constraints for Robust Real-Time Feature Matching". *[arXiv preprint], arXiv:2007.16005, 2020*.

[171] Daniel Hernandez-Juarez, Sarah Parisot, **Benjamin Busam**, Aleš Leonardis, Gregory Slabaugh, and Steven McDonagh. "A Multi-Hypothesis Approach to Color Constancy". *Conference on Computer Vision and Pattern Recognition (CVPR), 2020, Seattle, Washington, USA (virtual)*.

2019

[51] **Benjamin Busam**, Matthieu Hog, Steven McDonagh, and Gregory Slabaugh. "SteReFo: Efficient Image Refocusing with Stereo Vision". *International Conference on Computer Vision (ICCV), Advances in Image Manipulation (AIM), 2019, Seoul, Korea*. [**Oral Presentation**].

[272] Fabian Manhardt, Diego Martín Arroyo, Christian Rupprecht, **Benjamin Busam**, Tolga Birdal, Nassir Navab, and Federico Tombari. "Explaining the Ambiguity of Object Detection and 6D Pose from Visual Data". *International Conference on Computer Vision (ICCV), 2019, Seoul, Korea*.

[27] Tolga Birdal, **Benjamin Busam**, Nassir Navab, Slobodan Ilic, and Peter Sturm. "Generic Primitive Detection in Point Clouds Using Novel Minimal Quadric Fits". *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 2018, Volume 42, Issue 6*.

2018

[53] **Benjamin Busam**, Patrick Ruhkamp, Salvatore Virga, Beatrice Lentes, Julia Rackerseder, Nassir Navab, and Christoph Hennersperger. "Markerless Inside-Out Tracking for 3D Ultrasound Compounding". *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), Point-of-Care Ultrasound (POCUS), 2018, Granada, Spain*. [**Oral Presentation** and **Live Demonstration**].

[26] Tolga Birdal, **Benjamin Busam**, Nassir Navab, Slobodan Ilic, and Peter Sturm. "A Minimalist Approach to Type-Agnostic Detection of Quadrics in Point Clouds". *Conference on Computer Vision and Pattern Recognition (CVPR), 2018, Salt Lake City, Utah, USA*.

2017

[48] **Benjamin Busam**, Tolga Birdal, and Nassir Navab. "Camera Pose Filtering with Local Regression Geodesics on the Riemannian Manifold of Dual Quaternions". *International Conference on Computer Vision (ICCV), Multiview Relationships in 3D Data (MVR3D), 2017, Venice, Italy*. [**Oral Presentation** and **Best Student Paper Award**].

2016

[50] **Benjamin Busam**, Marco Esposito, Benjamin Frisch, and Nassir Navab. "Quaternionic Upsampling: Hyperspherical Techniques for 6 DoF Pose Tracking". *International Conference on 3D Vision (3DV), 2016, Stanford, California, USA*.

[102] Marco Esposito, **Benjamin Busam**, Christoph Hennersperger, Julia Rackerseder, Nassir Navab, and Benjamin Frisch. "Multimodal US–Gamma Imaging using Collaborative Robotics for Cancer Staging Biopsies". *International Journal of Computer Assisted Radiology and Surgery (IJCARS), 2016, Volume 11, Issue 9*. [**Best Paper Award**].

2015

[49] **Benjamin Busam**, Marco Esposito, Simon Che'Rose, Nassir Navab, and Benjamin Frisch. "A Stereo Vision Approach for Cooperative Robotic Movement Therapy". *International Conference on Computer Vision (ICCV), International Workshop on Assistive Computer Vision and Robotics (ACVR), 2015, Santiago, Chile*. [**Oral Presentation**, "Adaptable High-resolution Real-time Stereo Tracking" awarded with the **EMVA Young Professional Award 2015** from the European Machine Vision Association (EMVA)].

[101] Marco Esposito, **Benjamin Busam**, Christoph Hennersperger, Julia Rackerseder, An Lu, Nassir Navab, and Benjamin Frisch. “Cooperative Robotic Gamma Imaging: Enhancing US-guided Needle Biopsy”. *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), 2015, Munich, Germany*. [**Oral Presentation**].

C

List of Academic Projects & Research Funding

Academic Projects

2020

2020/21 WS Hyun Jun Jung. "Temporal Depth Sensor Fusion with Multiple Modalities". PhD project supervised by Nassir Navab, mentored by **Benjamin Busam**. *PhD Project, started in Winter Semester 2020/2021*.

2020/21 WS Daoyi Gao and Hanzhi Chen. "Temporally Consistent Video Depth". Practical project supervised by **Benjamin Busam**. *Advanced Topics in 3D Computer Vision, Winter Semester 2020/2021*.

2020 SS Barnabé Mas. "Self-supervised Depth Estimation from Polarization Imagery". Master thesis supervised by **Benjamin Busam**. Academic referent: Aymeric Dieuleveut. *Master Thesis, École Polytechnique, Department of Applied Mathematics, Summer Semester 2020*.

2020 SS Muhammad Faizan, Ihsan Balaban, and Jeremias Neth. "Depth-aware Mixed Reality: Capture the AR-Flag". Practical project supervised by **Benjamin Busam** and Patrick Ruhkamp. *Perception and Learning in Robotics and Augmented Reality, Summer Semester 2020*.

2019

2019/20 WS Hyun Jun Jung. "6D Pose Tracking as an Action Decision Process". Master thesis supervised by **Benjamin Busam** and Nassir Navab. *Master Thesis, Winter Semester 2019/2020*.

2019 SS Konstantinos Zacharis. "A Geometric View of Rotation Representations in Computer Vision and Deep Learning". Master thesis supervised by **Benjamin Busam** and Ulrich Bauer. *Master Thesis, Summer Semester 2019*.

2018

2018/19 WS Elias Marquart, Tobias Eder, and Varnika Tyagi. "Observing Environment Statistics – Anomaly Detection". Practical project supervised by **Benjamin Busam**. *Advanced Topics in 3D Computer Vision, Winter Semester 2018/2019*.

2018/19 WS Nour Al Orjany, Ritvik Ranadive, and Stefano Gasperini. “Mobile Indoor Navigation”. Practical project supervised by Patrick Ruhkamp and **Benjamin Busam**. *Advanced Topics in 3D Computer Vision, Winter Semester 2018/2019*.

2018/19 WS Hooriya Anam. “CodeSLAM – Learning a Compact, Optimisable Representation for Dense Visual SLAM”. Seminar presentation supervised by **Benjamin Busam**. *Recent Trends in 3D Computer Vision and Deep Learning, Winter Semester 2018/2019*.

2018/19 WS Hyun Jun Jung. “Spatio-temporal Depth Estimation in Real-Time”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Winter Semester 2018/2019*.

2018/19 WS Nitin Deshpande. “Context-Aware Robust SLAM Applications”. Master thesis supervised by Patrick Ruhkamp, **Benjamin Busam**, and Nassir Navab. *Master Thesis, Winter Semester 2018/2019*.

2018 SS Hyun Jun Jung. “Depth Accuracy Improvements for Active Stereo Sensing”. Working student supervised by **Benjamin Busam**. *Working Student, Summer Semester 2018*.

2018 SS Miguel Trasobares Baselga. “A modular Implementation of Feature based SLAM”. Working student supervised by **Benjamin Busam**. *Working Student, Summer Semester 2018*.

2018 SS Michael Haberl. “Object Pose Estimation with PointNet”. Master thesis supervised by **Benjamin Busam** and Ulrich Bauer. *Master Thesis, Summer Semester 2018*.

2018 SS Matthieu Hongtao Zhang. “Real-Time Gaze Estimation”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Summer Semester 2018*.

2018 SS Antoine Keller, Violeta Sofia Morales, and Lennart Bastian. “Motion Interpolation and Sensor Fusion”. Lab project supervised by Florian Lindemann and **Benjamin Busam**. *Case Studies in Non-linear Optimization, Summer Semester 2018*.

2017

2017/18 WS Mahdi Saleh. “Semi-supervised and Reinforcement Learning Techniques in 3D Computer Vision”. PhD project supervised by Nassir Navab, mentored by **Benjamin Busam**. *PhD Project, started in Winter Semester 2017/2018*.

2017/18 WS Patrick Ruhkamp. “Simultaneous Localization and Mapping in Challenging Environments”. PhD project supervised by Nassir Navab, mentored by **Benjamin Busam**. *PhD Project, started in Winter Semester 2017/2018*.

2017/18 WS Charalampos Papathanasis. “Hybrid Sensor Fusion with Dual Quaternion based EKF for Pose Estimation in Real-Time”. Master thesis supervised by **Benjamin Busam** and Nassir Navab. *Master Thesis, Winter Semester 2017/2018*.

2017/18 WS Thomas Sennebogen. “A mixed Reality Application for Needle Targeting with Trifocal Stereo in Real-Time”. Interdisciplinary project supervised by **Benjamin Busam**. *Interdisciplinary Project, Winter Semester 2017/2018*.

2017/18 WS Ruiqi Gong. “Dyna-Eye: A dynamic 2D-3D Stereo Viewer”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Winter Semester 2017/2018*.

2017/18 WS Tomas Bartipan. “Gaze Estimation with Pupil Tracking in Real-Time”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Winter Semester 2017/2018*.

2017/18 WS Tobias Valinski. “Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs”. Seminar presentation supervised by **Benjamin Busam**. *Recent Trends in 3D Computer Vision and Deep Learning, Winter Semester 2017/2018*.

2017/18 WS Felix Scheidhammer. “Pose-aware Rendering of live Ultrasound Data for mixed medical AR”. Bachelor thesis supervised by **Benjamin Busam** and Nassir Navab. *Bachelor Thesis, Winter Semester 2017/2018*.

2017 SS Lu Sang, Michael Haberl, and Raphael Ullmann. “Disparity Determination in Stereo Vision”. Lab project supervised by **Benjamin Busam** and Philipp Jarde. *Case Studies in Non-linear Optimization, Summer Semester 2017*.

2017 SS Aleksandr Bulankin. “Magic Eye: An adaptive 2D-3D Stereo Viewer”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Summer Semester 2017*.

2017 SS Ekaterina Kanaeva. “Pose Interpolation with Dual Quaternion Series”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Summer Semester 2017*.

2017 SS Ester Molero Hidalgo. “Activity Classification with Persistent Homology Barcodes”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Summer Semester 2017*.

2016

2016/17 Mahdi Saleh. “Natural Marker Extraction and Learning for binocular Images and 6D Pose Estimation”. Master thesis supervised by **Benjamin Busam** and Nassir Navab. *Master Thesis, 2016/2017*.

2016/17 Faisal Kalim. “Hybrid opto-inertial Tracking Systems”. Practical project with the Interdisciplinary Research Lab (IFL) at the university hospital Klinikum rechts der Isar supervised by **Benjamin Busam** and Benjamin Frisch. *Practical Project, 2016/2017*.

2016/17 WS Patrick Ruhkamp. “Monocular Reconstruction and Tracking in non-rigid Environments”. Master thesis supervised by **Benjamin Busam** and Nassir Navab. *Master Thesis, Winter Semester 2016/2017*.

2016/17 WS Charalampos Papathanasis. “Automatic Pose Synchronization”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Winter Semester 2016/2017*.

2016/17 WS Ramona Schneider. “FMtrack – Optical Tracking System for Medical Procedural Simulator”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Winter Semester 2016/2017*.

2016/17 WS Mahdi Hamad. “A virtual multi-view Optical Tracking System”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Winter Semester 2016/2017*.

2016/17 WS Felix Scheidhammer. “Fusion4D: Real-Time Performance Capture of Challenging Scenes”. Seminar presentation supervised by **Benjamin Busam**. *Recent Trends in 3D Computer Vision, Winter Semester 2016/2017*.

2016/17 WS Nan Yang. “Learning with Side Information through Modality Hallucination”. Seminar presentation supervised by **Benjamin Busam**. *Recent Trends in 3D Computer Vision, Winter Semester 2016/2017*.

2016 SS Faisal Kalim. “A hybrid opto-inertial Tracking System Prototype”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Summer Semester 2016*.

2016 SS Ahmed El-Gazzar. “A vision based anthropometric Scanner for dynamic Scenes”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Summer Semester 2016*.

2015

2015/16 WS Thomas Sennebogen. “Needle Tracking for Ultrasound-guided Biopsies with inside-out Vision”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Winter Semester 2015/2016*.

2015/16 WS Stefan Matl. “Visualizing the volumetric Accuracy of medical Tracking Solutions”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Winter Semester 2015/2016*.

2015/16 WS Christoph Baur. “Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture”. Seminar presentation supervised by **Benjamin Busam**. *Recent Trends in 3D Computer Vision, Winter Semester 2015/2016*.

2015/16 WS Markus Herb. “Computing the Stereo Matching Cost with a Convolutional Neural Network”. Seminar presentation supervised by **Benjamin Busam**. *Recent Trends in 3D Computer Vision, Winter Semester 2015/2016*.

2015 SS Mahdi Saleh. “Reducing the orthopaedic Risk of Cycling with a vision-based anthropometric Scanner”. Lab project supervised by **Benjamin Busam**. *Project Management and Software Development for Medical Applications, Summer Semester 2015*.

2014

2014/15 WS Christos Stoilas. "Rapid Object Detection using a boosted Cascade of simple Features". Seminar presentation supervised by **Benjamin Busam**. *Foundations of Computer Vision, Winter Semester 2014/2015*.

2014/15 WS Amin Ahantab. "Distinctive Image Features from scale-invariant Keypoints". Seminar presentation supervised by **Benjamin Busam**. *Foundations of Computer Vision, Winter Semester 2014/2015*.

2014 SS Ahmed Matar. "Monocular Real-Time Tracking with Passive Circular Markers". Bachelor thesis supervised by **Benjamin Busam**, Vasileios Belagiannis, Michael Friebe, and Nassir Navab. *Bachelor Thesis, Summer Semester 2014*.

Managed Research Funding & Acquired Grants

2017

From 09/2017 — IHATEC, BMVI. "IRiS – Interaktives Robotiksystem zur Entleerung von Seecontainern". Vision research and development for hard- and software components conducted by **Benjamin Busam**, Patrick Ruhkamp, and Mahdi Saleh. *Förderprogramm für Innovative Hafentechnologien (IHATEC), Bundesministerium für Verkehr und digitale Infrastruktur (BMVI), Project Duration: 09/2017 - 08/2020 (Benjamin Busam until 12/2018)*.

2016

From 01/2016 — ZIM, BMWi. "IOTMA – Inside-Out Tracking for Medical Applications". Vision research and development for both hard- and software components as well as management conducted by **Benjamin Busam**. *Zentrales Innovationsprogramm Mittelstand (ZIM), Bundeministerium für Wirtschaft und Energie (BMWi), Project Duration: 01/2016 - 12/2017*.

2015

From 10/2015 — Horizon 2020, EU-Project. "SYMBIONICA – Next Generation Bionics and Smart Prostetics". Hardware and software development for vision and monitoring systems as well as management partly conducted by **Benjamin Busam**. *Funded under: H2020-EU.2.1.5.1. - Technologies for Factories of the Future, Project Reference: 678144, Project Duration: 10/2015 - 09/2018*.

From 01/2015 — Horizon 2020, EU-Project. "BOREALIS – Enlightening Next Generation Material". Hardware and software development for vision and monitoring systems as well as management partly conducted by **Benjamin Busam**. *Funded under: H2020-EU.2.1.5.1. - Technologies for Factories of the Future, Project Reference: 636992, Project Duration: 01/2015 - 12/2017*.

2014

From 02/2014 — FP7, EU-Project. "MANufacturing through ergonoMic and safe Anthropocentric aDaptive workplacEs for context aware factories in EUROPE". Computer vision algorithms and sensing system hardware developed by **Benjamin Busam** and Paul Kreuzer. *Funded under: FP7-NMP, Project Reference: 609073, Project Duration: 09/2013 - 08/2016*.

List of Algorithms

List of Tables

List of Figures

Literature

[1] Rafal Ablamowicz and Garret Sobczyk. *Lectures on Clifford (geometric) algebras and applications*. Springer Science & Business Media, 2004. (see p. 97)

[2] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. "Building rome in a day" in: *12th International Conference on Computer Vision*. IEEE 2009. 72–79 (see p. 126)

[3] Sung Joon Ahn, Wolfgang Rauh, and Hans-Jürgen Warnecke. Least-squares orthogonal distances fitting of circle, sphere, ellipse, hyperbola, and parabola. *Pattern Recognition*, **34**: 2283–2303, 2001. (see p. 51)

[4] Sharath Akkaladevi, Martin Ankerl, Christoph Heindl, and Andreas Pichler. "Tracking multiple rigid symmetric and non-symmetric objects in real-time using depth data" in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2016. 5644–5649 (see p. 229)

[5] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. "Freak: Fast retina keypoint" in: *Conference on Computer Vision and Pattern Recognition*. IEEE 2012. 510–517 (see p. 212)

[6] James Andrews and Carlo H. Séquin. Type-constrained direct fitting of quadric surfaces. *Computer-Aided Design and Applications*, 2014. (see pp. 206, 208)

[7] Vladimir I. Arnol'd. The geometry of spherical curves and the algebra of quaternions. *Russian Mathematical Surveys*, **50**: 1, 1995. (see p. 93)

[8] Amir Atapour-Abarghouei and Toby P. Breckon. "Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018. 2800–2810 (see p. 126)

[9] Michael Bajura, Henry Fuchs, and Ryutarou Ohbuchi. Merging virtual objects with the real world: Seeing ultrasound imagery within the patient. *ACM SIGGRAPH Computer Graphics*, **26**: 203–210, 1992. (see p. 274)

[10] Alan H. Barr, Bena Currin, Steven Gabriel, and John F. Hughes. "Smooth Interpolation of Orientations with Angular Velocity Constraints Using Quaternions" in: *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH New York, NY, USA: ACM, 1992. 313–320 (see p. 239)

[11] Jonathan T. Barron. "A general and adaptive robust loss function" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 4331–4339 (see p. 214)

[12] Jonathan T. Barron and Ben Poole. "The fast bilateral solver" in: *European Conference on Computer Vision*. Springer 2016. 617–632 (see p. 123)

[13] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. "Key.net: Keypoint detection by handcrafted and learned cnn filters" in: *Proceedings of the IEEE International Conference on Computer Vision*. 2019. 5836–5844 (see p. 212)

[14] Axel Barroso-Laguna, Yannick Verdie, Benjamin Busam, and Krystian Mikolajczyk. “HDD-Net: Hybrid Detector Descriptor with Mutual Interactive Learning” in: *Asian Conference on Computer Vision*. 2020. (see pp. 213, 311)

[15] Angelo Basteris, Sharon M. Nijenhuis, A. H. Stienen, Jaap H. Buurke, Gerdienke B. Prange, and Farshid Amirabdollahian. Training modalities in robot-mediated upper limb rehabilitation in stroke: a framework for classification based on a systematic review. *Journal of Neuroengineering and Rehabilitation*, **11**: 111, 2014. (see pp. 182, 187, 188)

[16] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. “Surf: Speeded up robust features” in: *European conference on computer vision*. Springer 2006. 404–417 (see p. 212)

[17] Daniel Beale, Yong-Liang Yang, Neill Campbell, Darren Cosker, and Peter Hall. Fitting quadrics with a Bayesian prior. *Computational Visual Media*, 2016. (see p. 207)

[18] Jeffrey S. Beis and David G. Lowe. “Shape indexing using approximate nearest-neighbour search in high-dimensional spaces” in: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE 1997. 1000–1006 (see p. 213)

[19] Calin Belta and Vijay Kumar. An SVD-based projection method for interpolation on SE (3). *IEEE transactions on Robotics and Automation*, **18**: 334–345, 2002. (see p. 254)

[20] Filippo Bergamasco, Andrea Albarelli, Luca Cosmo, Emanuele Rodola, and Andrea Torsello. An accurate and robust artificial marker based on cyclic codes. *IEEE transactions on pattern analysis and machine intelligence*, **38**: 2359–2373, 2016. (see p. 140)

[21] Filippo Bergamasco, Andrea Albarelli, and Andrea Torsello. Pi-tag: a fast image-space marker design based on projective invariants. *Machine vision and applications*, **24**: 1295–1310, 2013. (see p. 140)

[22] Berta Bescos, José M. Fácil, Javier Civera, and José Neira. DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes. *IEEE Robotics and Automation Letters*, **3**: 4076–4083, 2018. (see p. 299)

[23] Paul J. Besl and Neil D. McKay. “Method for registration of 3-D shapes” in: *Sensor fusion IV: control paradigms and data structures*. vol. 1611 International Society for Optics and Photonics 1992. 586–606 (see pp. 5, 144)

[24] Jia-Wang Bian, Wen-Yan Lin, Yasuyuki Matsushita, Sai-Kit Yeung, Tan-Dat Nguyen, and Ming-Ming Cheng. “Gms: Grid-based motion statistics for fast, ultra-robust feature correspondence” in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017. 4181–4190 (see p. 213)

[25] Stan Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**: 401–406, 1998. (see p. 123)

[26] Tolga Birdal, Benjamin Busam, Nassir Navab, Slobodan Ilic, and Peter Sturm. “A minimalist approach to type-agnostic detection of quadrics in point clouds” in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018. 3530–3540 (see pp. 206, 207, 312)

[27] Tolga Birdal, Benjamin Busam, Nassir Navab, Slobodan Ilic, and Peter Sturm. Generic primitive detection in point clouds using novel minimal quadric fits. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **42**: 1333–1347, 2019. (see pp. 207, 312)

[28] Tolga Birdal, Ievgeniia Dobryden, and Slobodan Ilic. “X-tag: A fiducial tag for flexible and accurate bundle adjustment” in: *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE 2016. 556–564 (see pp. 140, 211, 218)

[29] Tolga Birdal and Slobodan Ilic. “Point pair features based object detection and pose estimation revisited” in: *International Conference on 3D Vision (3DV)*. IEEE 2015. (see p. 206)

[30] Michael M. Blane, Zhibin Lei, Hakan Çivi, and David B. Cooper. The 3L algorithm for fitting implicit polynomial curves and surfaces to data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000. (see p. 207)

[31] Amy A. Blank, James A. French, Ali Utku Pehlivan, and Marcia K. O'Malley. Current trends in robot-assisted upper-limb stroke rehabilitation: promoting patient engagement in therapy. *Current Physical Medicine and Rehabilitation Reports*, **2**: 184–195, 2014. (see p. 182)

[32] Michael Bloesch, Michael Burri, Sammy Omari, Marco Hutter, and Roland Siegwart. Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback. *The International Journal of Robotics Research*, **36**: 1053–1072, 2017. (see p. 181)

[33] Ludwig Boltzmann. Studien uber das Gleichgewicht der lebenden Kraft. *Wissenschafiliche Abhandlungen*, **1**: 49–96, 1868. (see p. 150)

[34] Dorit Borrmann, Jan Elseberg, Kai Lingemann, and Andreas Nüchter. The 3d hough transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, **2**: 2011. (see p. 207)

[35] Paul Bourke *2 Dimensional FFT* 1998 URL: http://paulbourke.net/miscellaneous/dft/ (visited on Sept. 1, 2020) (see p. 30)

[36] Aljaz Bozic, Michael Zollhofer, Christian Theobalt, and Matthias Nießner. "Deepdeform: Learning non-rigid rgb-d reconstruction with semi-supervised data" in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020. 7002–7012 (see p. 299)

[37] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. "Learning 6d object pose estimation using 3d object coordinates" in: *European Conference on Computer Vision*. Springer 2014. 536–551 (see pp. 208, 214, 218, 219)

[38] Gary Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. (see pp. 140, 198)

[39] Alex Bricou, Marie-Alix Duval, Yves Charon, and Emmanuel Barranger. Mobile gamma cameras in breast cancer care – a review. *European Journal of Surgical Oncology (EJSO)*, **39**: 409–416, 2013. (see p. 280)

[40] John S. Bridle. "Training Stochastic Model Recognition Algorithms as Networks can Lead to Maximum Mutual Information Estimation of Parameters" in: *Advances in Neural Information Processing Systems*. Denver. Colorado. USA, Nov. 1989. 211–217 (see p. 150)

[41] Brockhaus, ed. *Der Brockhaus multimedial 2009 premium* Elektronische Enzyklopädie auf DVD 2009 (see p. 14)

[42] Ilja N. Bronstein, Konstantin A. Semendjaew, Gerhard Musiol, and Heiner Mühlig. *Taschenbuch der Mathematik*. 6th ed. Deutsch, 2006. (see p. 16)

[43] Recommendation ITU-R BT. Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios. 1990. (see pp. 45, 46)

[44] Patrick Buehler, Mark Everingham, Daniel P. Huttenlocher, and Andrew Zisserman. Upper Body Detection and Tracking in Extended Signing Sequences. *International Journal of Computer Vision*, **95**: 180–197, 2011. (see p. 188)

[45] Sarah Louise Bugby, John E. Lees, and Alan C. Perkins. Hybrid intraoperative imaging techniques in radioguided surgery: present clinical applications and future outlook. *Clinical and Translational Imaging*, **5**: 323–341, 2017. (see p. 280)

[46] Wilhelm Burger and Mark J. Burge. *Digital Image Processing*. First Edit Springer Science+Business Media, LLC, 2008. (see p. 40)

[47] Benjamin Busam. *Projective Geometry and 3D point cloud matching*. MA thesis. Zentrum für Mathematik, Technische Universität München, 85747 Garching bei München, Germany: Technische Universität München, 2014. (see pp. 11, 21, 87, 104, 136)

[48] Benjamin Busam, Tolga Birdal, and Nassir Navab. “Camera Pose Filtering with Local Regression Geodesics on the Riemannian Manifold of Dual Quaternions” in: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2017. 2436–2445 (see pp. 8, 92, 312)

[49] Benjamin Busam, Marco Esposito, Simon Che’Rose, Nassir Navab, and Benjamin Frisch. “A Stereo Vision Approach for Cooperative Robotic Movement Therapy” in: *IEEE International Conference on Computer Vision Workshop (ICCVW)*. 2015. 519–527 (see pp. 6, 7, 134, 193, 211, 218, 247, 248, 282, 312)

[50] Benjamin Busam, Marco Esposito, Benjamin Frisch, and Nassir Navab. “Quaternionic upsampling: Hyperspherical techniques for 6 dof pose tracking” in: *Fourth International Conference on 3D Vision (3DV)*. IEEE, Oct. 2016. 629–638 (see pp. 8, 92, 203, 214, 239, 260, 261, 312)

[51] Benjamin Busam, Matthieu Hog, Steven McDonagh, and Gregory Slabaugh. “SteReFo: Efficient Image Refocusing with Stereo Vision” in: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2019. (see pp. 218, 311)

[52] Benjamin Busam, Hyun Jun Jung, and Nassir Navab. I Like to Move It: 6D Pose Estimation as an Action Decision Process. *arXiv preprint arXiv:2009.12678*, 2020. (see pp. 7, 218, 311)

[53] Benjamin Busam, Patrick Ruhkamp, Salvatore Virga, Beatrice Lentes, Julia Rackerseder, Nassir Navab, and Christoph Hennersperger. “Markerless Inside-Out Tracking for 3D Ultrasound Compounding” in: *Simulation, Image Processing, and Ultrasound Systems for Assisted Diagnosis and Navigation*. Cham: Springer, 2018. 56–64 (see pp. 7, 312)

[54] Samuel R. Buss and Jay P. Fillmore. Spherical averages and applications to spherical splines and interpolation. *ACM Transactions on Graphics (TOG)*, **20**: 95–126, 2001. (see p. 257)

[55] Hongping Cai, Tomáš Werner, and Jiří Matas. “Fast detection of multiple textureless 3-D objects” in: *International Conference on Computer Vision Systems*. Springer 2013. 103–112 (see p. 214)

[56] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. “The ycb object and model set: Towards common benchmarks for manipulation research” in: *International Conference on Advanced Robotics (ICAR)*. IEEE 2015. 510–517 (see pp. 4, 210)

[57] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. “Brief: Binary robust independent elementary features” in: *European conference on computer vision*. Springer 2010. 778–792 (see p. 212)

[58] Carlos Campos, Richard Elvira, Juan J. Gómez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. *arXiv preprint arXiv:2007.11898*, 2020. (see pp. 4, 194, 212)

[59] Federico Camposeco and Marc Pollefeys. “Using vanishing points to improve visual-inertial odometry” in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2015. 5219–5225 (see p. 181)

[60] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 679–698, 1986. (see p. 45)

[61] Like Cao, Jie Ling, and Xiaohui Xiao. The WHU Rolling Shutter Visual-Inertial Dataset. *IEEE Access*, **8**: 50771–50779, 2020. (see p. 181)

[62] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Zhuoqing Morley Mao. “Adversarial sensor attack on lidar-based perception in autonomous driving” in: *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 2019. 2267–2281 (see p. 269)

[63] Catherine Capellen, Max Schwarz, and Sven Behnke. ConvPoseCNN: Dense Convolutional 6D Object Pose Estimation. *arXiv preprint arXiv:1912.07333*, 2019. (see p. 307)

[64] Fabrice Chassat and Stephane Lavallée. “Experimental protocol of accuracy evaluation of 6-D localizers for computer-integrated surgery: Application to four optical localizers” in: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer 1998. 277–284 (see p. 141)

[65] Changhao Chen, Stefano Rosa, Yishu Miao, Chris Xiaoxuan Lu, Wei Wu, Andrew Markham, and Niki Trigoni. “Selective sensor fusion for neural visual-inertial odometry” in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 10542–10551 (see p. 181)

[66] Homer H. Chen. “A screw motion approach to uniqueness analysis of head-eye geometry” in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. June 1991. 145–151 (see p. 102)

[67] Sachin Chitta, Ioan Sucan, and Steve Cousins. Moveit! [ros topics]. *IEEE Robotics & Automation Magazine*, **19**: 18–19, 2012. (see p. 185)

[68] Sungil Choi, Seungryong Kim, Kihong Park, and Kwanghoon Sohn. “Learning Descriptor, Confidence, and Depth Estimation in Multi-view Stereo” in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018. 276–282 (see p. 123)

[69] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. “The loss surfaces of multilayer networks” in: *Artificial Intelligence and Statistics*. 2015. 192–204 (see p. 81)

[70] Yim-Pan Chui and Pheng-Ann Heng. “Adaptive attitude dead-reckoning by cumulative polynomial extrapolation of quaternions” in: *Fifth IEEE International Workshop on Distributed Simulation and Real-Time Applications*. Aug. 2001. 45–52 (see p. 239)

[71] William Kingdon Clifford. Preliminary Sketch of Biquaternions. *Proceedings of the London Mathematical Society*, **1-4**: 381–395, 1871. (see p. 96)

[72] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. “The Cityscapes Dataset for Semantic Urban Scene Understanding” in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016. 3213–3223 (see p. 123)

[73] Peter I. Corke and Malcolm C. Good. “Dynamic effects in high-performance visual servoing” in: *Proceedings of the International Conference on Robotics and Automation*. IEEE 1992. 1838–1839 (see p. 283)

[74] Alberto Crivellaro, Mahdi Rad, Yannick Verdie, Kwang Moo Yi, Pascal Fua, and Vincent Lepetit. “A novel representation of parts for accurate 3D object detection and tracking in monocular images” in: *Proceedings of the IEEE international conference on computer vision*. 2015. 4391–4399 (see pp. 214, 232)

[75] Geoffrey Cross and Andrew Zisserman. “Quadric reconstruction from dual-space geometry” in: *International Conference on Computer Vision*. 1998. (see pp. 5, 206, 207)

[76] George Cybenko. Approximations by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, **2**: 183–192, 1989. (see p. 78)

[77] Thomas Czerniawski, Bharathwaj Sankaran, Mohammad Nahangi, Carl T. M. Haas, and Fernanda Lustosa Leite. 6D DBSCAN-based segmentation of building point clouds for planar object classification. *Automation in Construction*, **88**: 44–58, 2018. (see p. 208)

[78] Erik B. Dam, Martin Koch, and Martin Lillholm. *Quaternions, interpolation and animation*. Datalogisk Institut, Københavns Universitet, 1998. (see p. 239)

[79] Jun Dang, Benjamin Frisch, Philippe Lasaygues, Dachun Zhang, Stefaan Tavernier, Nicolas Felix, Paul Lecoq, Etiennette Auffray, Jennifer Varela, Serge Mensah, and Mingxi Wan. Development of an Anthropomorphic Breast Phantom for Combined PET, B-Mode Ultrasound and Elastographic Imaging. *IEEE Transactions on Nuclear Science*, **58**: 660–667, 2011. (see p. 291)

[80] Konstantinos Daniilidis. Hand-eye calibration using dual quaternions. *The International Journal of Robotics Research*, **18**: 286–298, 1999. (see pp. 101, 244)

[81] Christian Demant, Bernd Streicher-Abel, and Peter Waszkewitz. *Industrial Image Processing*. Springer Berlin Heidelberg, 1999. (see p. 22)

[82] Xinke Deng, Arsalan Mousavian, Yu Xiang, Fei Xia, Timothy Bretl, and Dieter Fox. "PoseRBPF: A Rao-Blackwellized Particle Filter for6D Object Pose Estimation" in: *Proceedings of Robotics: Science and Systems*. Freiburg im Breisgau, Germany, June 2019. (see pp. 217, 229, 232, 307)

[83] John S. Denker, W. R. Gardner, Hans Peter Graf, Donnie Henderson, Richard E. Howard, W. Hubbard, Lawrence D. Jackel, Henry S. Baird, and Isabelle Guyon. "Neural network recognizer for hand-written zip code digits" in: *Advances in Neural Information Processing Systems*. 1989. 323–331 (see p. 80)

[84] Simon Denman, Todd Lamb, Clinton Fookes, Vinod Chandran, and Sridha Sridharan. Multi-spectral fusion for surveillance systems. *Computers & Electrical Engineering*, **36**: 643–663, 2010. (see p. 269)

[85] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Toward geometric deep SLAM. *arXiv preprint arXiv:1707.07410*, 2017. (see p. 212)

[86] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. "Superpoint: Self-supervised interest point detection and description" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018. 224–236 (see pp. 4, 213)

[87] Peter Deuflhard and Andreas Hohmann. *Numerische Mathematik I.* 3., überarbeitete Version de Gruyter, 2002. (see pp. 30, 156)

[88] Gianmarco Dinelli, Gabriele Meoni, Emilio Rapuano, Gionata Benelli, and Luca Fanucci. An fpga-based hardware accelerator for cnns using on-chip memories only: Design and benchmarking with intel movidius neural compute stick. *International Journal of Reconfigurable Computing*, **2019**: 2019. (see p. 299)

[89] Thanh-Toan Do, Ming Cai, Trung Pham, and Ian Reid. Deep-6DPose: Recovering 6D Object Pose from a Single RGB Image. *arXiv preprint arXiv:1802.10367*, 2018. (see p. 214)

[90] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. "CARLA: An Open Urban Driving Simulator" in: *Proceedings of the Conference on Robot Learning (CoRL)*. 2017. 1–16 (see p. 126)

[91] Andreas Doumanoglou, Rigas Kouskouridas, Sotiris Malassiotis, and Tae-Kyun Kim. "Recovering 6D object pose and predicting next-best-view in the crowd" in: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016. 3583–3592 (see p. 209)

[92] Bertram Drost, Markus Ulrich, Paul Bergmann, Philipp Hartinger, and Carsten Steger. "Introducing mvtec itodd-a dataset for 3d object recognition in industry" in: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017. 2200–2208 (see pp. 209, 218)

[93] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. "D2-net: A trainable cnn for joint description and detection of local features" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 8092–8101 (see p. 213)

[94] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. "Temporal Cycle-Consistency Learning" in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019. (see p. 266)

[95] David Eigen, Christian Puhrsch, and Rob Fergus. "Depth map prediction from a single image using a multi-scale deep network" in: *Advances in Neural Information Processing Systems*. 2014. 2366–2374 (see p. 127)

[96] Robert Elfring, Matías de la Fuente, and Klaus Radermacher. Assessment of optical localizer accuracy for computer aided surgery systems. *Computer Aided Surgery*, **15**: 1–12, 2010. (see pp. 4, 141, 159, 160)

[97] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, **40**: 611–625, 2018. (see pp. 4, 126, 181, 194, 196)

[98] Jakob Engel, Thomas Schöps, and Daniel Cremers. "LSD-SLAM: Large-scale direct monocular SLAM" in: *European Conference on Computer Vision*. Springer 2014. 834–849 (see pp. 4, 126, 194, 196)

[99] Jakob Engel, Jürgen Sturm, and Daniel Cremers. "Camera-based navigation of a low-cost quadrocopter" in: *International Conference on Intelligent Robots and Systems*. IEEE 2012. 2815–2821 (see p. 4)

[100] Zeynep Ercan and Salim Yüce. On Properties of the Dual Quaternions. *European Journal of Pure and Applied Mathematics*, **4**: 142–146, 2011. (see p. 96)

[101] Marco Esposito, Benjamin Busam, Christoph Hennersperger, Julia Rackerseder, An Lu, Nassir Navab, and Benjamin Frisch. "Cooperative robotic gamma imaging: Enhancing us-guided needle biopsy" in: *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer 2015. 611–618 (see pp. 8, 140, 193, 211, 218, 239, 278, 289, 313)

[102] Marco Esposito, Benjamin Busam, Christoph Hennersperger, Julia Rackerseder, Nassir Navab, and Benjamin Frisch. Multimodal US–gamma imaging using collaborative robotics for cancer staging biopsies. *International Journal of Computer Assisted Radiology and Surgery (IJCARS)*, **11**: 1561–1571, 2016. (see pp. 8, 211, 278, 312)

[103] Hao Fang, Florent Lafarge, and Mathieu Desbrun. "Planar Shape Detection at Structural Scales" in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018. (see p. 208)

[104] Michela Farenzena, Adrien Bartoli, and Youcef Mezouar. "Automatically smoothing camera pose using cross validation for sequential vision-based 3D mapping" in: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE 2008. 3616–3621 (see pp. 95, 253)

[105] Olivier D. Faugeras. *Three-Dimensional Computer Vision.* Fourth Print. The MIT Press, 2001. (see pp. 94, 105)

[106] Olivier D. Faugeras and Francis Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, **2**: 485–508, 1988. (see p. 126)

[107] Valery L. Feigin, Mohammad H. Forouzanfar, Rita Krishnamurthi, George A. Mensah, Myles Connor, Derrick A. Bennett, Andrew E. Moran, Ralph L. Sacco, Laurie Anderson, Thomas Truelsen, Martin O'Donnell, Narayanaswamy Venketasubramanian, Suzanne Barker-Collo, Carlene M. M. Lawes, Wenzhi Wang, Yukito Shinohara, Emma Witt, Majid Ezzati, Mohsen Naghavi, and Christopher Murray. Global and regional burden of stroke during 1990–2010: findings from the Global Burden of Disease Study 2010. *The Lancet*, **383**: 245–255, 2014. (see p. 182)

[108] Xiang Feng and Wanggen Wan. Dual Quaternion Blending Algorithm and Its Application in Character Animation. *Indonesian Journal of Electrical Engineering and Computer Science*, **11**: 5553–5562, 2013. (see p. 246)

[109] Aaron Fenster, Dónal Downey, and Neale Cardinal. Three-dimensional ultrasound imaging. *Physics in Medicine & Biology*, **46**: R67, 2001. (see p. 193)

[110] Ian S. Fentiman, Alain Fourquet, and Gabriel N. Hortobagyi. Male breast cancer. *The Lancet*, **367**: 595–604, 2006. (see p. 278)

[111] Mark Fiala. "ARTag, a fiducial marker system using digital techniques" in: *Conference on Computer Vision and Pattern Recognition (CVPR)*. vol. 2 IEEE 2005. 590–596 (see pp. 4, 140, 211)

[112] Mark Fiala. Designing Highly Reliable Fiducial Markers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**: 1317–1324, 2010. (see p. 188)

[113] Sanja Fidler, Marko Boben, and Aleš Leonardis. Learning a hierarchical compositional shape vocabulary for multi-class object representation. *arXiv preprint arXiv:1408.5516*, 2014. (see p. 207)

[114] Nuno Filipe, Michail Kontitsis, and Panagiotis Tsiotras. Extended Kalman filter for spacecraft pose estimation using dual quaternions. *Journal of Guidance, Control, and Dynamics*, **38**: 1625–1641, 2015. (see p. 253)

[115] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, **24**: 381–395, 1981. (see p. 213)

[116] Andrew W. Fitzgibbon. Robust registration of 2D and 3D point sets. *Image and Vision Computing*, **21**: 1145–1153, 2003. (see p. 144)

[117] Andrew W. Fitzgibbon, Maurizio Pilu, and Robert B. Fisher. Direct least square fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**: 476–480, 1999. (see pp. 52, 53)

[118] Tully Foote. "tf: The transform library" in: *Conference on Technologies for Practical Robot Applications (TePRA)*. IEEE 2013. 1–6 (see p. 185)

[119] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration theory for fast and accurate visual-inertial navigation. *IEEE Transactions on Robotics*, 1–18, 2015. (see p. 181)

[120] John Fox. Time-series regression and generalized least squares. *Appendix to An R S-PLUS Companion to Applied Regression. Thousand Oaks, CA*, 1–8, 2002. (see p. 252)

[121] Eric Foxlin, Yury Altshuler, Leonid Naimark, and Mike Harrington. "FlightTracker: A novel optical / inertial tracker for cockpit enhanced vision" in: *Proceedings of the third IEEE/ACM International Symposium on Mixed and Augmented Reality*. ISMAR 2004. 212–221 (see p. 239)

[122] Engelmann Francis, Kontogianni Theodora, Hermans Alexander, and Leibe Bastian. "Exploring Spatial Context for 3D Semantic Segmentation of Point Clouds" in: *Proceedings of the IEEE International Conference on Computer Vision, 3DRMS Workshop, ICCV*. 2017. (see p. 126)

[123] Alfred M. Franz, Tamas Haidegger, Wolfgang Birkfellner, Kevin Cleary, Terry M. Peters, and Lena Maier-Hein. Electromagnetic tracking in medicine – a review of technology, validation, and applications. *IEEE Transactions on Medical Imaging*, **33**: 1702–1725, 2014. (see p. 141)

[124] Rainer Gemma Frisius. *De radio astronomico et geometrico liber*. Bontius, 1545. 135 (see pp. 59, 60)

[125] Mingliang Fu and Weijia Zhou. DeepHMap++: Combined Projection Grouping and Correspondence Learning for Full DoF Pose Estimation. *Sensors*, **19**: 1032, 2019. (see pp. 215, 227, 228, 309)

[126] Keinosuke Fukunaga and Patrenahalli M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *IEEE Transactions on Computers*, **100**: 750–753, 1975. (see p. 213)

[127] Janez Funda, Russel H. Taylor, and Richard P. Paul. On homogeneous transforms, quaternions, and computational efficiency. *IEEE Transactions on Robotics and Automation*, **6**: 382–388, 1990. (see p. 244)

[128] Paul Furgale, Timothy D. Barfoot, and Gabe Sibley. "Continuous-time batch estimation using temporal basis functions" in: *International Conference on Robotics and Automation*. IEEE 2012. 2088–2095 (see p. 240)

[129] Paul Furgale, Chi Hay Tong, Timothy D. Barfoot, and Gabe Sibley. Continuous-time batch trajectory estimation using temporal basis functions. *The International Journal of Robotics Research*, **34**: 1688–1710, 2015. (see p. 240)

[130] Simon Henry Gage and Henry Phelps Gage. *Optic projection, principles, installation, and use of the magic lantern, projection microscope, reflecting lantern, moving picture machine*. Comstock, 1914. (see p. 59)

[131] Adrian Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. "Virtual Worlds as Proxy for Multi-Object Tracking Analysis" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016. (see p. 126)

[132] Guillermo Gallego, Tobi Delbruck, Garrick Michael Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jorg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1, 2020. (see p. 238)

[133] Jean Gallier. Notes on differential geometry and Lie groups. *University of Pennsylvannia*, 2012. (see p. 100)

[134] Ravi Garg, Vijay Kumar B.G., Gustavo Carneiro, and Ian Reid. "Unsupervised CNN for single view depth estimation: Geometry to the rescue" in: *European Conference on Computer Vision*. Springer 2016. 740–756 (see p. 127)

[135] Mathieu Garon and Jean-François Lalonde. Deep 6-DOF tracking. *IEEE transactions on visualization and computer graphics*, **23**: 2410–2418, 2017. (see pp. 210, 217, 229, 231)

[136] Mathieu Garon, Denis Laurendeau, and Jean-François Lalonde. "A Framework for Evaluating 6-DOF Object Trackers" in: *European Conference on Computer Vision*. 2018. (see pp. 4, 5, 210, 217–219, 226, 228, 229, 232)

[137] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, **47**: 2280–2292, 2014. (see pp. 4, 140, 188, 199, 208, 211)

[138] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, **32**: 1231–1237, 2013. (see p. 89)

[139] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite" in: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012. (see pp. 89, 123)

[140] Jason Geng. Structured-light 3D surface imaging: a tutorial. *Advances in Optics and Photonics*, **3**: 128–160, 2011. (see p. 88)

[141] Josiah Willard Gibbs. *Elementary principles in statistical mechanics: developed with especial reference to the rational foundation of thermodynamics*. C. Scribner's sons, 1902. (see p. 150)

[142] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Deep sparse rectifier neural networks" in: *Proceedings of the International Conference on Artificial Intelligence and Statistics*. 2011. 315–323 (see pp. 78, 83)

[143] Rüdiger Göbl, Nassir Navab, and Christoph Hennersperger. SUPRA: open-source software-defined ultrasound processing for real-time applications. *International Journal of Computer Assisted Radiology and Surgery*, 2018. (see p. 202)

[144] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017. 270–279 (see pp. 127, 194)

[145] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. "Digging into self-supervised monocular depth estimation" in: *Proceedings of the IEEE International Conference on Computer Vision*. 2019. 3828–3838 (see pp. 127, 194, 266)

[146] Steven Gold, Chien Ping Lu, Anand Rangarajan, Suguna Pappu, and Eric Mjolsness *Fast Algorithms for 2D and 3D Point Matching: Pose Estimation and Correspondence* Research Report YALEU/DCS/RR-1035 Department of Computer Science. Yale University. New Haven. CT 06520-8285, May 1994 (see p. 146)

[147] Steven Gold, Chien-Ping Lu, Anand Rangarajan, Suguna Pappu, and Eric Mjolsness. "New algorithms for 2D and 3D point matching: Pose estimation and correspondence" in: *Advances in Neural Information Processing Systems*. 1995. 957–964 (see p. 145)

[148] Steven Gold and Anand Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**: 377–388, 1996. (see p. 148)

[149] Steven Gold, Anand Rangarajan, Chien-Ping Lu, Suguna Pappu, and Eric Mjolsness. New algorithms for 2D and 3D point matching: pose estimation and correspondence. *Pattern Recognition*, **31**: 1019–1031, 1998. (see pp. 152, 156)

[150] Herbert Goldstein, Charles Poole, and John Safko. *Classical Mechanics*. Addison-Wesley, 1980. 426 (see p. 95)

[151] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Third Edition Pearson Prentice Hall, 2008. (see pp. 26, 31)

[152] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*. vol. 1 MIT press Cambridge, 2016. (see p. 75)

[153] Sébastien Granger and Xavier Pennec. "Multi-scale EM-ICP: A fast and robust approach for surface registration" in: *European Conference on Computer Vision*. Springer 2002. 418–432 (see p. 144)

[154] Gaël Guennebaud, Benoît Jacob, et al. *Eigen v3* http://eigen.tuxfamily.org 2010 (see p. 249)

[155] Chao X. Guo, Dimitrios G. Kottas, Ryan DuToit, Ahmed Ahmed, Ruipeng Li, and Stergios I. Roumeliotis. "Efficient Visual-Inertial Navigation using a Rolling-Shutter Camera with Inaccurate Timestamps." in: *Robotics: Science and Systems*. 2014. (see p. 181)

[156] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy Ren, and Xiaogang Wang. "Learning monocular depth by distilling cross-domain stereo networks" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 484–500 (see p. 127)

[157] Adrian Haarbach, Tolga Birdal, and Slobodan Ilic. "Survey of higher order rigid body motion interpolation methods for keyframe animation and continuous-time trajectory estimation" in: *International Conference on 3D Vision (3DV)*. IEEE 2018. 381–389 (see pp. 239, 240)

[158] John K. Haas *A history of the unity game engine* tech. rep. Worcester Polytechnic Institute, 2014 (see p. 224)

[159] Radim Halíř and Jan Flusser. "Numerically Stable Direct Least Squares Fitting of Ellipses" in: *The sixth International Conference in Central Europe on Computer Graphics and Visualization*. 1998. (see p. 53)

[160] William Rowan Hamilton. On quaternions; or on a new system of imaginaries in algebra. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, **25**: 10–13, 1844. (see pp. 93, 96)

[161] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J. Davison. "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM" in: *Proceedings of the International Conference on Robotics and Automation (ICRA)*. IEEE 2014. 1524–1531 (see p. 126)

[162] Christopher G. Harris and Mike Stephens. "A combined corner and edge detector" in: *Proceedings of the Fourth Alvey Vision Conference*. 1988. 147–151 (see p. 212)

[163] Richard Hartley. Theory and Practice of Projective Rectification. *International Journal of Computer Vision*, **35**: 115–127, 1999. (see pp. 114, 116)

[164] Richard Hartley, Khurrum Aftab, and Jochen Trumpf. "L1 rotation averaging using the Weiszfeld algorithm" in: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE 2011. 3041–3048 (see pp. 170, 284)

[165] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Second Edition Cambridge University Press, 2010. (see pp. 62, 71, 106, 109, 112, 116)

[166] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn" in: *International Conference on Computer Vision (ICCV)*. IEEE 2017. 2980–2988 (see pp. 32, 214)

[167] Donald Olding Hebb. *The organization of behavior: a neuropsychological theory*. J. Wiley, Chapman & Hall, 1949. (see p. 81)

[168] Janne Heikkila and Olli Silven. "A four-step camera calibration procedure with implicit image correction" in: *Conference on Computer Vision and Pattern Recognition*. IEEE 1997. 1106–1112 (see pp. 59, 66, 67)

[169] Robert Held, Ankit Gupta, Brian Curless, and Maneesh Agrawala. "3D puppetry: a kinect-based interface for 3D animation" in: *Proceedings of the 25th annual ACM Symposium on User Interface Software and Technology*. 2012. 423–434 (see p. 217)

[170] Christoph Hennersperger, Bernhard Fuerst, Salvatore Virga, Oliver Zettinig, Benjamin Frisch, Thomas Neff, and Nassir Navab. Towards MRI-Based Autonomous Robotic US Acquisitions: A First Feasibility Study. *IEEE Transactions on Medical Imaging*, **36**: 538–548, 2017. (see p. 192)

[171] Daniel Hernandez-Juarez, Sarah Parisot, Benjamin Busam, Aleš Leonardis, Gregory Slabaugh, and Steven McDonagh. "A Multi-Hypothesis Approach to Color Constancy" in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020. 2270–2280 (see p. 311)

[172] Joel A. Hesch and Stergios I. Roumeliotis. "A direct least-squares (DLS) method for PnP" in: *International Conference on Computer Vision*. IEEE 2011. 383–390 (see p. 213)

[173] Derrek A. Heuveling, K. Hakki Karagozoglu, Annelies Van Schie, Stijn Van Weert, A. Van Lingen, and Remco De Bree. Sentinel node biopsy using 3D lymphatic mapping by freehand SPECT in early stage oral cancer: a new technique. *Clinical otolaryngology: official journal of ENT-UK; official journal of Netherlands Society for Oto-Rhino-Laryngology & Cervico-Facial Surgery*, **37**: 89, 2012. (see p. 193)

[174] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes" in: *Asian Conference on Computer Vision*. Springer 2012. 548–562 (see pp. 5, 208, 214, 218, 219, 224, 227, 228, 306)

[175] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. (see p. 80)

[176] Heiko Hirschmüller. "Accurate and efficient stereo processing by semi-global matching and mutual information" in: *Conference on Computer Vision and Pattern Recognition (CVPR)*. vol. 2 IEEE 2005. 807–814 (see p. 123)

[177] Heiko Hirschmüller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, **30**: 328–341, 2007. (see pp. 123, 196)

[178] Tomáš Hodaň, Pavel Haluza, Štěpán Obdržálek, Jiří Matas, Manolis Lourakis, and Xenophon Zabulis. "T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects" in: *Winter Conference on Applications of Computer Vision (WACV)*. IEEE 2017. 880–888 (see pp. 5, 209)

[179] Tomáš Hodaň, Frank Michel, Eric Brachmann, Wadim Kehl, Anders GlentBuch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, Caner Sahin, Fabian Manhardt, Federico Tombari, Tae-Kyun Kim, Jiří Matas, and Carsten Rother. "Bop: Benchmark for 6d object pose estimation" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 19–34 (see pp. 209, 218)

[180] Tomáš Hodaň, Xenophon Zabulis, Manolis Lourakis, Štěpán Obdržálek, and Jiří Matas. "Detection and fine 3D pose estimation of texture-less objects in RGB-D images" in: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE 2015. 4421–4428 (see p. 214)

[181] Judy Hoffman, Saurabh Gupta, and Trevor Darrell. "Learning with side information through modality hallucination" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016. 826–834 (see p. 267)

[182] Aleksander Holynski and Johannes Kopf. "Fast depth densification for occlusion-aware augmented reality" in: *SIGGRAPH Asia Technical Papers*. ACM 2018. (see p. 218)

[183] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, **4**: 251–257, 1991. (see p. 78)

[184] Peter R. Hoskins, Kevin Martin, and Abigail Thrush. *Diagnostic ultrasound: physics and equipment*. CRC Press, 2019. (see p. 279)

[185] Po-Wei Hsu, Richard W. Prager, Andrew H. Gee, and Graham M. Treece. "Freehand 3D ultrasound calibration: a review" in: *IBM*. Springer, 2009. 47–84 (see p. 195)

[186] Danying Hu, Daniel DeTone, and Tomasz Malisiewicz. "Deep charuco: Dark charuco marker pose estimation" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 8436–8444 (see p. 140)

[187] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. "Segmentation-driven 6d object pose estimation" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 3385–3394 (see pp. 214, 227, 228)

[188] Thomas S. Huang and Arun N. Netravali. "Motion and structure from feature correspondences: A review" in: *Advances In Image Processing And Understanding: A Festschrift for Thomas S. Huang*. World Scientific, 2002. 331–347 (see p. 126)

[189] David H. Hubel and Torsten N. Wiesel. Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. *Journal of Neurophysiology*, **28**: 229–289, 1965. (see p. 79)

[190] David H. Hubel and Torsten N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, **195**: 215–243, 1968. (see p. 79)

[191] Bernd Jähne. *Digital Image Processing*. Sixth revised Edition Springer Berlin Heidelberg New York, 2005. (see pp. 42, 87)

[192] Brijnesh J. Jain and Michael Lappe. "Joining Softassign and Dynamic Programming for the Contact Map Overlap Problem" in: *Bioinformatics Research and Development*. Lecture Notes in Computer Science Springer Berlin Heidelberg, 2007. 410–423 (see p. 152)

[193] Mona Jalal, Josef Spjut, Ben Boudaoud, and Margrit Betke. "SIDOD: A Synthetic Image Dataset for 3D Object Pose Recognition With Distractors" in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. June 2019. (see p. 210)

[194] Maximilian Jaritz, Raoul De Charette, Emilie Wirbel, Xavier Perrotton, and Fawzi Nashashibi. "Sparse and dense data with cnns: Depth completion and semantic segmentation" in: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE 2018. 52–60 (see p. 128)

[195] Chao Jia and Brian L Evans. Constrained 3D Rotation Smoothing via Global Manifold Regression for Video Stabilization. *IEEE Transactions on Signal Processing*, **62**: 3293–3304, 2014. (see pp. 6, 95, 253)

[196] Jianbo Jiao, Ying Cao, Yibing Song, and Rynson Lau. "Look Deeper into Depth: Monocular Depth Estimation with Semantic Booster and Attention-Driven Loss" in: *The European Conference on Computer Vision (ECCV)*. Sept. 2018. (see p. 128)

[197] Bijoy Johnson and G. Somu. Robotic Telesurgery: Benefits Beyond Barriers. *BMH Medical Journal*, **3**: 2016. (see p. 238)

[198] Eagle S. Jones and Stefano Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *The International Journal of Robotics Research*, **30**: 407–430, 2011. (see p. 180)

[199] Fredrick Johnson Joseph, Alexander van Oepen, and Michael Friebe. Breast sentinel lymph node biopsy with imaging towards minimally invasive surgery. *Biomedical Engineering/Biomedizinische Technik*, **62**: 547–555, 2017. (see p. 279)

[200] David Joseph Tan, Federico Tombari, Slobodan Ilic, and Nassir Navab. "A versatile learning-based 3d temporal tracker: Scalable, robust, online" in: *Proceedings of the IEEE International Conference on Computer Vision*. 2015. 693–701 (see p. 217)

[201] Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*, 2018. (see p. 225)

[202] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, **32**: 922–923, 1976. (see p. 5)

[203] Roman Kaskman, Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. "Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects" in: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2019. (see pp. 5, 208, 216, 218)

[204] Hirokazu Kato and Mark Billinghurst. "Marker tracking and hmd calibration for a video-based augmented reality conferencing system" in: *Second International Workshop on Augmented Reality (IWAR)*. IEEE 1999. 85–94 (see pp. 4, 140, 211, 218)

[205] Ladislav Kavan. *Real-time Skeletal Animation*. PhD thesis. Faculty of Electrical Engineering, Czech Technical University in Prague, 2007. (see p. 239)

[206] Ladislav Kavan, Steven Collins, Carol O'Sullivan, and Jiří Žára. Dual quaternions for rigid transformation blending. *Trinity College Dublin, Tech. Rep. TCD-CS-2006-46*, 2006. (see pp. 104, 246)

[207] Ladislav Kavan, Steven Collins, Jiří Žára, and Carol O'Sullivan. "Skinning with dual quaternions" in: *Proceedings of the Symposium on Interactive 3D graphics and games*. ACM 2007. 39–46 (see pp. 6, 96, 239)

[208] Ladislav Kavan and Jiří Žára. "Spherical Blend Skinning: A Real-time Deformation of Articulated Models" in: *Proceedings of the Symposium on Interactive 3D Graphics and Games (I3D)*. Washington, District of Columbia: ACM, 2005. 9–16 (see p. 246)

[209] Tong Ke and Stergios I. Roumeliotis. "An efficient algebraic solution to the perspective-three-point problem" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017. 7225–7233 (see p. 213)

[210] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. "SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again" in: *Proceedings of the International Conference on Computer Vision (ICCV)*. 2017. 22–29 (see pp. 5, 214, 219, 224, 232)

[211] Wadim Kehl, Fausto Milletari, Federico Tombari, Slobodan Ilic, and Nassir Navab. "Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation" in: *European Conference on Computer Vision*. Springer 2016. 205–220 (see pp. 214, 218)

[212] Wadim Kehl, Federico Tombari, Slobodan Ilic, and Nassir Navab. "Real-time 3D model tracking in color and depth on a single CPU core" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017. 745–753 (see p. 217)

[213] Wadim Kehl, Federico Tombari, Nassir Navab, Slobodan Ilic, and Vincent Lepetit. "Hashmod: A Hashing Method for Scalable 3D Object Detection" in: *Proceedings of the British Machine Vision Conference (BMVC)*. Sept. 2015. 36.1–36.12 (see p. 214)

[214] Ben Kenwright. “A beginners guide to dual-quaternions: What they are, how they work, and how to use them for 3D character hierarchies” in: *20th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*. 2012. 1–10 (see p. 239)

[215] Ben Kenwrigth. Inverse kinematics with dual-quaternions, exponential-maps, and joint limits. *International Journal on Advances in Intelligent Systems*, **6**: 2013. (see pp. 6, 96)

[216] Rasool Khadem, Clement C. Yeh, Mohammad Sadeghi-Tehrani, Michael R. Bax, Jeremy A. Johnson, Jacqueline Nerney Welch, Eric P. Wilkinson, and Ramin Shahidi. Comparative tracking error analysis of five different optical tracking systems. *Computer Aided Surgery*, **5**: 98–107, 2000. (see p. 141)

[217] Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, and Shahram Izadi. “Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction” in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 573–590 (see p. 123)

[218] Myoung-Jun Kim, Myung-Soo Kim, and Sung Yong Shin. “A general construction scheme for unit quaternion curves with simple high order derivatives” in: *Proceedings of the 22nd annual Conference on Computer Graphics and Interactive Techniques*. 1995. 369–376 (see p. 239)

[219] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (see pp. 83, 225)

[220] Pavel Kirsanov, Airat Gaskarov, Filipp Konokhov, Konstantin Sofiiuk, Anna Vorontsova, Igor Slinko, Dmitry Zhukov, Sergey Bykov, Olga Barinova, and Anton Konushin. “DISCOMAN: Dataset of Indoor SCenes for Odometry, Mapping And Navigation” in: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE 2019. 2470–2477 (see p. 181)

[221] Gabriel Kiss, Sigurd Storve, Bjørn Olav Haugen, and Hans Torp. “Augmented reality based tools for echocardiographic acquisitions” in: *International Ultrasonics Symposium*. IEEE 2014. 695–698 (see p. 274)

[222] Maria Klodt and Andrea Vedaldi. “Supervising the new with the old: learning SFM from SFM” in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 698–713 (see p. 127)

[223] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, **36**: 78, 2017. (see pp. 126, 218)

[224] Peter Knoll, S. Mirzaei, K. Schwenkenbecher, and Thomas Barthel. Performance evaluation of a solid-state detector based handheld gamma camera system. *Frontiers in Biomedical Technologies*, **1**: 2014. (see p. 286)

[225] David N. Krag, Stewart J. Anderson, Thomas B. Julian, Ann M. Brown, Seth P. Harlow, Joseph P. Costantino, Takamaru Ashikaga, Donald L. Weaver, Eleftherios P. Mamounas, Lynne M. Jalovec, Thomas G. Frazier, R. Dirk Noyes, André Robidoux, Hugh M. C. Scarth, and Norman Wolmark. Sentinel-lymph-node resection compared with conventional axillary-lymph-node dissection in clinically node-negative patients with breast cancer: overall survival findings from the NSABP B-32 randomised phase 3 trial. *The Lancet Oncology*, **11**: 927–933, 2010. (see p. 279)

[226] David N. Krag, D. L. Weaver, J. C. Alex, and J. T. Fairbank. Surgical resection and radiolocalization of the sentinel lymph node in breast cancer using a gamma probe. *Surgical Oncology*, **2**: 335–340, 1993. (see p. 279)

[227] Florian Kral, Elisabeth Puschban, Herbert Riechelmann, and Wolfgang Freysinger. Comparison of optical and electromagnetic tracking for navigated lateral skull base surgery. *The International Journal of Medical Robotics and Computer Assisted Surgery*, **9**: 247–252, 2013. (see p. 192)

[228] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “Imagenet classification with deep convolutional neural networks” in: *Advances in Neural Information Processing Systems*. 2012. 1097–1105 (see p. 80)

[229] Jason Ku, Ali Harakeh, and Steven L. Waslander. “In defense of classical image processing: Fast depth completion on the cpu” in: *Proceedings of the Conference on Computer and Robot Vision (CRV)*. IEEE 2018. 16–22 (see p. 128)

[230] Ying Kuang, Aihua Mao, Guiqing Li, and Yunhui Xiong. “A strategy of real-time animation of clothed body movement” in: *International Conference on Multimedia Technology*. ICMT 2011. 4793–4797 (see p. 239)

[231] James J. Kuffner and Steven M. LaValle. “RRT-connect: An efficient approach to single-query path planning” in: *International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. vol. 2 IEEE 2000. 995–1001 (see p. 185)

[232] KUKA GmbH *KUKA LWR. User-friendly, sensitive and flexible.* 2012 (visited on Oct. 8, 2020) (see pp. 186, 248)

[233] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. “Deeper depth prediction with fully convolutional residual networks” in: *Fourth International Conference on 3D Vision (3DV)*. IEEE 2016. 239–248 (see p. 127)

[234] Pranav Lakshminarayanan *IGTLink4J* Johns Hopkins University. Baltimore, MD 2015 URL: `https://github.com/pranavl/igtlink4j/tree/master/lib/OpenIGTLink` (visited on Oct. 20, 2020) (see p. 276)

[235] Stefan Lanser. *Modellbasierte Lokalisation gestützt auf monokulare Videobilder*. PhD thesis. Fakultät für Informatik der Technischen Universität München, 1997. (see pp. 50, 54, 65, 69, 71)

[236] Andras Lasso, Tamas Heffter, Adam Rankin, Csaba Pinter, Tamas Ungi, and Gabor Fichtinger. PLUS: Open-Source Toolkit for Ultrasound-Guided Intervention Systems. *IEEE Transactions on Biomedical Engineering*, **61**: 2527–2537, 2014. (see pp. 202, 274, 282, 284)

[237] Joseph J. LaViola. “Double exponential smoothing: an alternative to Kalman filter-based predictive tracking” in: *Proceedings of the Workshop on Virtual Environments*. ACM 2003. 199–206 (see p. 253)

[238] Nalpantidis Lazaros, Georgios Christou Sirakoulis, and Antonios Gasteratos. Review of stereo vision algorithms: from software to hardware. *International Journal of Optomechatronics*, **2**: 435–462, 2008. (see p. 123)

[239] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, **521**: 436, 2015. (see p. 75)

[240] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. “Handwritten digit recognition with a back-propagation network” in: *Advances in Neural Information Processing Systems*. 1990. 396–404 (see p. 82)

[241] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**: 2278–2324, 1998. (see p. 80)

[242] Man Hee Lee and In Kyu Park. “Performance evaluation of local descriptors for affine invariant region detector” in: *Asian Conference on Computer Vision*. Springer 2014. 630–643 (see p. 212)

[243] Aleš Leonardis, Ales Jaklic, and Franc Solina. Superquadrics for segmenting and modeling range data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**: 1289–1295, 1997. (see pp. 5, 207)

[244] Vincent Lepetit and Pascal Fua. Monocular Model-Based 3D Tracking of Rigid Objects: A Survey. *Foundations and Trends® in Computer Graphics and Vision*, **1**: 1–89, 2005. (see pp. 93, 239)

[245] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. EPnP: An Accurate O(n) Solution to the PnP Problem. *International Journal of Computer Vision*, **81**: 155–166, 2009. (see pp. 213, 218)

[246] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. "BRISK: Binary robust invariant scalable keypoints" in: *International Conference on Computer Vision (ICCV)*. IEEE 2011. 2548–2555 (see p. 212)

[247] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Y. Siegwart, and Paul Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, **34**: 314–334, 2015. (see p. 181)

[248] Mingyang Li and Anastasios I. Mourikis. Vision-aided inertial navigation with rolling-shutter cameras. *The International Journal of Robotics Research*, **33**: 1490–1507, 2014. (see p. 181)

[249] Yali Li, Shengjin Wang, Qi Tian, and Xiaoqing Ding. A survey of recent advances in visual feature detection. *Neurocomputing*, **149**: 736–751, 2015. (see p. 212)

[250] Yangyan Li, Xiaokun Wu, Yiorgos Chrysathou, Andrei Sharf, Daniel Cohen-Or, and Niloy J. Mitra. "Globfit: Consistently fitting primitives by discovering global relations" in: *ACM Transactions on Graphics (TOG)*. 2011. (see p. 207)

[251] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. "Deepim: Deep iterative matching for 6d pose estimation" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 683–698 (see pp. 215, 222, 232, 309)

[252] Yunpeng Li, Noah Snavely, and Daniel P. Huttenlocher. "Location recognition using prioritized feature matching" in: *European Conference on Computer Vision*. Springer 2010. 791–804 (see p. 212)

[253] Zhengqi Li and Noah Snavely. "Megadepth: Learning single-view depth prediction from internet photos" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018. 2041–2050 (see p. 127)

[254] Zhigang Li, Gu Wang, and Xiangyang Ji. "CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation" in: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2019. (see p. 215)

[255] Paul P. Lin, David C. Allison, Jean Wainstock, Kathy D. Miller, William C. Dooley, Neil Friedman, and Ralph R. Baker. Impact of axillary lymph node dissection on the therapy of breast cancer patients. *Journal of Clinical Oncology*, **11**: 1536–1544, 1993. (see p. 279)

[256] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft coco: Common objects in context" in: *European Conference on Computer Vision*. Springer 2014. 740–755 (see p. 224)

[257] Tony Lindeberg. Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of Applied Statistics*, **21**: 225–270, 1994. (see p. 212)

[258] Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, **30**: 79–116, 1998. (see p. 212)

[259] Wenlei Liu, Sentang Wu, and Xiaolong Wu. Pose estimation method for planar mirror based on one-dimensional target. *Optical Engineering*, **57**: 073101, 2018. (see p. 178)

[260] Hugh Cristopher Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, **293**: 133–135, 1981. (see p. 112)

[261] Adrian Lopez-Rodriguez, Benjamin Busam, and Krystian Mikolajczyk. "Project to Adapt: Domain Adaptation for Depth Completion from Noisy and Sparse Sensor Data" in: *Asian Conference on Computer Vision*. 2020. (see pp. 266, 311)

[262] Ezequiel López-Rubio, Karl Thurnhofer-Hemsi, Óscar David de Cózar-Macías, Elidia Beatriz Blázquez-Parra, José Muñoz-Pérez, and Isidro Ladrón de Guevara-López. Robust Fitting of Ellipsoids by Separating Interior and Exterior Points During Optimization. *Journal of Mathematical Imaging and Vision*, 2016. (see p. 208)

[263] Steven Lovegrove, Alonso Patron-Perez, and Gabe Sibley. “Spline Fusion: A continuous-time representation for visual-inertial fusion with application to rolling shutter cameras.” in: *Proceedings of the British Machine Vision Conference (BMVC)*. 2013. (see pp. 181, 239)

[264] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, **60**: 91–110, 2004. (see pp. 4, 212)

[265] Siegrid Lowel and Wolf Singer. Selection of intrinsic horizontal connections in the visual cortex by correlated neuronal activity. *Science*, **255**: 209–212, 1992. (see p. 81)

[266] Miguel Angel Lozano and Francisco Escolano. “A Significant Improvement of Softassign with Diffusion Kernels” in: *Structural, Syntactic, and Statistical Pattern Recognition*. Lecture Notes in Computer Science Springer Berlin Heidelberg, 2004. 76–84 (see p. 152)

[267] Riccardo de Lutio, Stefano D’Aronco, Jan Dirk Wegner, and Konrad Schindler. “Guided Super-Resolution As Pixel-to-Pixel Transformation” in: *Proceedings of the IEEE International Conference on Computer Vision*. 2019. 8829–8837 (see p. 123)

[268] Fangchang Ma, Guilherme Venturelli Cavalheiro, and Sertac Karaman. “Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera” in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2019. 3288–3295 (see p. 128)

[269] Fangchang Ma and Sertac Karaman. “Sparse-to-dense: Depth prediction from sparse depth samples and a single image” in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2018. 1–8 (see p. 128)

[270] Lena Maier-Hein, Alfred Franz, Hans-Peter Meinzer, and Ivo Wolf. “Comparative assessment of optical tracking systems for soft tissue navigation with fiducial needles” in: *Medical Imaging 2008: Visualization, Image-Guided Procedures, and Modeling*. International Society for Optics and Photonics 2008. (see p. 141)

[271] Elmar Mair, Gregory D. Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. “Adaptive and generic corner detection based on the accelerated segment test” in: *European Conference on Computer Vision*. Springer 2010. 183–196 (see p. 212)

[272] Fabian Manhardt, Diego Martin Arroyo, Christian Rupprecht, Benjamin Busam, Tolga Birdal, Nassir Navab, and Federico Tombari. “Explaining the ambiguity of object detection and 6d pose from visual data” in: *Proceedings of the IEEE International Conference on Computer Vision*. 2019. 6841–6850 (see pp. 215, 311)

[273] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. “Deep model-based 6d pose refinement in rgb” in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 800–815 (see p. 232)

[274] Fabian Manhardt, Manuel Nickel, Sven Meier, Luca Minciullo, and Nassir Navab. CPS: Class-level 6D Pose and Shape Estimation From Monocular Images. *arXiv preprint arXiv:2003.05848*, 2020. (see p. 216)

[275] Éric Marchand, Fabien Spindler, and François Chaumette. ViSP for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics & Automation Magazine*, **12**: 40–52, 2005. (see pp. 176, 198, 248, 274)

[276] Eugenio Marinetto, David García-Mato, Alonso García, Santiago Martínez, Manuel Desco, and Javier Pascau. Multicamera Optical Tracker Assessment for Computer Aided Surgery Applications. *IEEE Access*, **6**: 64359–64370, 2018. (see pp. 4, 139, 141, 159)

[277] MathWorks *Documentation of vision.KalmanFilter class. Matlab Documentation Center* Image Processing Toolbox. Import, Export, and Conversion. Image Type Conversion. MathWorks 2016 URL: `https://de.mathworks.com/help/vision/ref/vision.kalmanfilter-class.html` (visited on Mar. 11, 2016) (see p. 260)

[278] Philipp Matthies, José Gardiazabal, Aslı Okur, Jakob Vogel, Tobias Lasser, and Nassir Navab. Mini gamma cameras for intra-operative nuclear tomographic reconstruction. *Medical Image Analysis*, **18**: 1329–1336, 2014. (see p. 286)

[279] Nikolaus Mayer, Eddy Ilg, Philipp Fischer, Caner Hazirbas, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. What makes good synthetic training data for learning disparity and optical flow estimation? *International Journal of Computer Vision (IJCV)*, **126**: 942–960, 2018. (see p. 126)

[280] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016. 4040–4048 (see p. 126)

[281] Ajmal Mian, Mohammed Bennamoun, and Robyn Owens. On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. *International Journal of Computer Vision*, **89**: 348–361, 2010. (see p. 213)

[282] Microsoft, ed. *Microsoft Encarta 2009 - Enzyklopädie* Elektronische Enzyklopädie auf DVD 2008 (see p. 39)

[283] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiří Matas, Frederik Schaffalitzky, Timor Kadir, and Luc Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, **65**: 43–72, 2005. (see p. 212)

[284] Ondrej Miksik and Krystian Mikolajczyk. "Evaluation of local detectors and descriptors for fast feature matching" in: *Proceedings of the 21st International Conference on Pattern Recognition (ICPR)*. IEEE 2012. 2681–2684 (see p. 212)

[285] James R. Miller. Analysis of quadric-surface-based solid models. *IEEE Computer Graphics and Applications*, **8**: 28–42, 1988. (see p. 206)

[286] David L. Mills. *Computer network time synchronization: the network time protocol on earth and in space*. CRC press, 2017. (see p. 247)

[287] Aitor Ruano Miralles. *An open-source development environment for self-driving vehicles*. MA thesis. Universitat Oberta de Catalunya, 2017. (see p. 126)

[288] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiří Matas. "Working hard to know your neighbor's margins: Local descriptor learning loss" in: *Advances in Neural Information Processing Systems*. 2017. 4826–4837 (see p. 213)

[289] Hans D. Mittelmann and Peter Spellucci *Decision Tree for Optimization Software* School of Mathematical & Statistical Sciences. Arizona State University 2013 URL: http://plato.asu.edu/guide.html (visited on Jan. 3, 2014) (see p. 147)

[290] Hans P. Moravec. "Towards Automatic Visual Obstacle Avoidance" in: *Proceedings of the fifth International Joint Conference on Artificial Intelligence*. vol. 584 1977. (see p. 212)

[291] Anastasios I. Mourikis and Stergios I. Roumeliotis. "A multi-state constraint Kalman filter for vision-aided inertial navigation" in: *International Conference on Robotics and Automation*. IEEE 2007. 3565–3572 (see p. 180)

[292] Marius Muja and David G. Lowe. "Flann, fast library for approximate nearest neighbors" in: *International Conference on Computer Vision Theory and Applications (VISAPP)*. vol. 3 INSTICC Press 2009. (see p. 213)

[293] Mahesh Chandra Mukkamala and Matthias Hein. "Variants of rmsprop and adagrad with logarithmic regret bounds" in: *Proceedings of the 34th International Conference on Machine Learning*. JMLR 2017. 2545–2553 (see p. 83)

[294] Ramakrishnan Mukundan. “Quaternions: From classical mechanics to computer graphics, and beyond” in: *Proceedings of the 7th Asian Technology Conference in Mathematics*. 2002. 97–105 (see p. 93)

[295] Johannes Müller and Christina Kuttler. Methods and models in mathematical biology. *Deterministic and Stochastic Approaches*, 2015. (see p. 15)

[296] Aaftab Munshi, Dan Ginsburg, and Dave Shreiner. *OpenGL ES 2.0 programming guide*. Pearson Education, 2008. (see p. 276)

[297] Raúl Mur-Artal, Jose Maria Martinez Montiel, and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, **31**: 1147–1163, 2015. (see pp. 4, 126, 194, 212, 218)

[298] Raúl Mur-Artal and Juan D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics (T-RO)*, **33**: 1255–1262, 2017. (see pp. 4, 126, 193, 194, 196, 212, 218)

[299] Raúl Mur-Artal and Juan D. Tardós. Visual-inertial monocular SLAM with map reuse. *IEEE Robotics and Automation Letters*, **2**: 796–803, 2017. (see p. 181)

[300] Richard M. Murray, Zexiang Li, and S. Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994. (see p. 100)

[301] Andriy Myronenko and Xubo Song. Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**: 2262–2275, 2010. (see p. 145)

[302] Andriy Myronenko, Xubo Song, and Miguel A. Carreira-Perpinán. “Non-rigid point set registration: Coherent point drift” in: *Advances in Neural Information Processing Systems*. 2007. 1009–1016 (see p. 145)

[303] Leonid Naimark and Eric Foxlin. “Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker” in: *Proceedings of the first International Symposium on Mixed and Augmented Reality*. IEEE Computer Society 2002. 27 (see pp. 4, 140, 188)

[304] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. “KinectFusion: Real-time dense surface mapping and tracking” in: *10th International Symposium on Mixed and Augmented Reality*. IEEE 2011. 127–136 (see p. 262)

[305] Yonhon Ng, Bomin Jiang, Changbin Yu, and Hongdong Li. “Non-iterative, fast SE (3) path smoothing” in: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE 2016. 3172–3179 (see p. 253)

[306] Gregory M. Nielson. v-Quaternion splines for the smooth interpolation of orientations. *IEEE Transactions on Visualization and Computer Graphics*, **10**: 224–229, 2004. (see p. 239)

[307] David Nistér, Oleg Naroditsky, and James Bergen. “Visual odometry” in: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE 2004. (see p. 4)

[308] Ciarán O’Conaire, Noel E. O’Connor, Eddie Cooke, and Alan F. Smeaton. “Comparison of fusion methods for thermo-visual surveillance tracking” in: *9th International Conference on Information Fusion*. IEEE 2006. 1–7 (see p. 269)

[309] Lawrence O’Gorman, Michael J. Sammon, and Michael Seul. *Practical Algorithms for Image Analysis*. Second Edition Cambridge University Press, 2008. (see p. 25)

[310] Paul O’Leary and Paul Zsombor-Murray. Direct and specific least-square fitting of hyperbolæ and ellipses. *Journal of Electronic Imaging*, **13**: 492–503, 2004. (see p. 53)

[311] John J. O’Connor and Edmund F. Robertson. Marius Sophus Lie. *The MacTutor History of Mathematics Archive*, 2000. (see p. 99)

[312] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. “Making deep heatmaps robust to partial occlusions for 3d object pose estimation” in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 119–134 (see pp. 215, 227, 228, 308)

[313] Reinhard Obwegeser, Katharina Lorenz, Maria Hohlagschwandtner, Klaus Czerwenka, Barbara Schneider, and Ernst Kubista. Axillary lymph nodes in breast cancer: is size related to metastatic involvement? *World Journal of Surgery*, **24**: 546–550, 2000. (see pp. 279, 290)

[314] Sven Oesau, Florent Lafarge, and Pierre Alliez. “Planar shape detection and regularization in tandem” in: *Computer Graphics Forum*. vol. 35 1 Wiley Online Library 2016. 203–215 (see p. 207)

[315] Aslı Okur, Christoph Hennersperger, Brent Runyan, José Gardiazabal, Matthias Keicher, Stefan Paepke, Thomas Wendler, and Nassir Navab. “FhSPECT-US guided needle biopsy of sentinel lymph nodes in the axilla: is it feasible?” in: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer 2014. 577–584 (see p. 280)

[316] Edwin Olson. “AprilTag: A robust and flexible visual fiducial system” in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2011. 3400–3407 (see pp. 4, 140, 211)

[317] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. “LF-Net: learning local features from images” in: *Advances in Neural Information Processing Systems*. 2018. 6234–6244 (see p. 213)

[318] Hannes Ovrén and Per-Erik Forssén. Trajectory representation and landmark projection for continuous-time structure from motion. *The International Journal of Robotics Research*, **38**: 686–701, 2019. (see p. 240)

[319] Cameron Lowell Palmer, Bjørn Olav Haugen, Eva Tegnander, Sturla H. Eik-Nes, Hans Torp, and Gabriel Kiss. “Mobile 3D augmented-reality system for ultrasound applications” in: *International Ultrasonics Symposium (IUS)*. IEEE 2015. 1–4 (see p. 274)

[320] Rick Parent. “Computer animation: algorithms and techniques – a historical review” in: *Proceedings Computer Animation*. IEEE 2000. 86–90 (see pp. 239, 252)

[321] Byeonghoon Park, Yongchan Keh, Donghi Lee, Yongkwan Kim, Sungsoon Kim, Kisuk Sung, Jungkee Lee, Donghoon Jang, and Youngkwon Yoon. “Outdoor Operation of Structured Light in Mobile Phone” in: *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*. Oct. 2017. (see p. 269)

[322] Andreas ten Pas and Robert Platt. Localizing grasp affordances in 3-d points clouds using taubin quadric fitting. *arXiv preprint arXiv:1311.3192*, 2013. (see p. 207)

[323] Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. “Superquadrics revisited: Learning 3d shape parsing beyond cuboids” in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 10344–10353 (see p. 207)

[324] Alonso Patron-Perez, Steven Lovegrove, and Gabe Sibley. A spline-based trajectory representation for sensor fusion and rolling shutter cameras. *International Journal of Computer Vision*, **113**: 208–219, 2015. (see p. 239)

[325] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G. Derpanis, and Kostas Daniilidis. “6-dof object pose from semantic keypoints” in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2017. 2011–2018 (see p. 216)

[326] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. “PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation” in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 4561–4570 (see pp. 215, 218, 309)

[327] Ettore Pennestrì and Pier Valentini. Dual quaternions as a tool for rigid body motion analysis: A tutorial with an application to biomechanics. *Archive of Mechanical Engineering*, **57**: 187–205, 2010. (see p. 239)

[328] Arul Selvam Periyasamy, Max Schwarz, and Sven Behnke. "Refining 6D Object Pose Predictions using Abstract Render-and-Compare" in: *19th International Conference on Humanoid Robots (Humanoids)*. IEEE 2019. 739–746 (see p. 308)

[329] Edward Pervin and Jon A. Webb *Quaternions in computer vision and robotics* tech. rep. Carnegie Mellon University. Computer Science Department, 1982 (see p. 93)

[330] Giorgia Pitteri, Slobodan Ilic, and Vincent Lepetit. "CorNet: Generic 3D Corners for 6D Pose Estimation of New Objects without Retraining" in: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2019. (see p. 216)

[331] Matteo Poggi, Fabio Tosi, and Stefano Mattoccia. "Learning Monocular Depth Estimation with Unsupervised Trinocular Assumptions" in: *International Conference on 3D Vision (3DV)*. 2018. 324–333 (see p. 127)

[332] Stephen B. Pollard, John E. W. Mayhew, and John P. Frisby. PMF: A stereo correspondence algorithm using a disparity gradient limit. *Perception*, **14**: 449–470, 1985. (see p. 129)

[333] Michael V. Potter, Lauro V. Ojeda, Noel C. Perkins, and Stephen M. Cain. Effect of IMU Design on IMU-Derived Stride Metrics for Running. *Sensors*, **19**: 2601, 2019. (see p. 238)

[334] Jennifer Pulman and Emily Buckley. Assessing the efficacy of different upper limb hemiparesis interventions on improving health-related quality of life in stroke patients: a systematic review. *Topics in Stroke Rehabilitation*, **20**: 171–188, 2013. (see p. 182)

[335] Dale Purves, Roberto Cabeza, Scott A. Huettel, Kevin S. LaBar, Michael L. Platt, Marty G. Woldorff, and Elizabeth M. Brannon. *Cognitive Neuroscience*. Sunderland: Sinauer Associates, Inc, 2008. (see p. 75)

[336] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, **34**: 1004–1020, 2018. (see p. 181)

[337] Rongqi Qiu, Qian-Yi Zhou, and Ulrich Neumann. "Pipe-run extraction and reconstruction from point clouds" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer 2014. (see p. 208)

[338] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. "ROS: an open-source Robot Operating System" in: *International Conference on Robotics and Automation (ICRA), Workshop on Open Source Software*. vol. 3 2009. 5 (see pp. 185, 248, 283)

[339] Mahdi Rad and Vincent Lepetit. "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth" in: *International Conference on Computer Vision (ICCV)*. 2017. (see p. 214)

[340] Mahdi Rad, Markus Oberweger, and Vincent Lepetit. "Domain transfer for 3d pose estimation from color images without manual annotations" in: *Asian Conference on Computer Vision*. Springer 2018. 69–84 (see pp. 5, 215)

[341] Mahdi Rad, Markus Oberweger, and Vincent Lepetit. "Feature mapping for learning fast and accurate 3d pose inference from synthetic images" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018. 4663–4672 (see p. 215)

[342] Anand Rangarajan, Haili Chui, and Fred L. Bookstein. "The softassign Procrustes matching algorithm" in: *Information Processing in Medical Imaging*. Lecture Notes in Computer Science Springer Berlin Heidelberg, 1997. 29–42 (see p. 152)

[343] Siddharth S. Rautaray and Anupam Agrawal. Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*, **43**: 1–54, 2015. (see p. 188)

[344] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016. 779–788 (see p. 214)

[345] Colin Rennie, Rahul Shome, Kostas E. Bekris, and Alberto F. De Souza. A dataset for improved rgbd-based object detection and pose estimation for warehouse pick-and-place. *IEEE Robotics and Automation Letters*, **1**: 1179–1185, 2016. (see p. 209)

[346] Jerome Revaud, Philippe Weinzaepfel, César Roberto de Souza, and Martin Humenberger. "R2D2: Repeatable and Reliable Detector and Descriptor" in: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019. (see p. 213)

[347] Jürgen Richter-Gebert. *Perspectives on Projective Geometry – A Guided Tour Through Real and Complex Geometry*. vol. 22 Springer, 2011. 571 (see p. 50)

[348] Jürgen Richter-Gebert and Thorsten Orendt. *Geometriekalküle*. vol. 11 Springer-Lehrbuch, 2009. 224 (see pp. 64, 93, 114)

[349] Jesse Richter-Klug and Udo Frese. "Towards Meaningful Uncertainty Information for CNN Based 6D Pose Estimates" in: *International Conference on Computer Vision Systems*. Springer 2019. 408–422 (see p. 308)

[350] Nuno Roma, José Santos-Victor, and José Tomé. "A comparative analysis of cross-correlation matching algorithms using a pyramidal resolution approach" in: *Empirical Evaluation Methods in Computer Vision*. World Scientific, 2002. 117–142 (see p. 123)

[351] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016. 3234–3243 (see p. 126)

[352] Adrian Rosebrock. Deep Learning for Computer Vision with Python. *PyImageSearch*, 2017. (see p. 75)

[353] Michael Rosenthal, Andrei State, Joohi Lee, Gentaro Hirota, Jeremy Ackerman, Kurtis Keller, Etta D. Pisano, Michael Jiroutek, Keith Muller, and Henry Fuchs. "Augmented Reality Guidance for Needle Biopsies: A Randomized, Controlled Trial in Phantoms" in: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer, 2001. 240–248 (see p. 274)

[354] Edward Rosten and Tom Drummond. "Machine learning for high-speed corner detection" in: *European Conference on Computer Vision*. Springer 2006. 430–443 (see p. 212)

[355] Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**: 105–119, 2008. (see p. 212)

[356] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. "ORB: An efficient alternative to SIFT or SURF" in: *International Conference on Computer Vision*. IEEE, Nov. 2011. 2564–2571 (see pp. 4, 212)

[357] Walter Rudin. *Reelle und Komplexe Analysis*. Oldenbourg Wissenschaftsverlag GmbH, 1999. (see pp. 16, 27)

[358] Patrick Ruhkamp, Ruiqi Gong, Nassir Navab, and Benjamin Busam. DynaMiTe: A Dynamic Local Motion Model with Temporal Constraints for Robust Real-Time Feature Matching. *arXiv preprint arXiv:2007.16005*, 2020. (see pp. 204, 213, 311)

[359] Szymon Rusinkiewicz and Marc Levoy. "Efficient variants of the ICP algorithm" in: *Proceedings of the third International Conference on 3D Digital Imaging and Modeling (3DIM)*. IEEE 2001. 145–152 (see pp. 5, 144, 217)

[360] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. "Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments" in: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE 2009. 1–6 (see p. 213)

[361] Ehab Salahat and Murad Qasaimeh. "Recent advances in features extraction and description algorithms: A comprehensive survey" in: *International Conference on Industrial Technology (ICIT)*. IEEE 2017. 1059–1063 (see p. 212)

[362] Mahdi Saleh, Shervin Dehghani, Benjamin Busam, Nassir Navab, and Federico Tombari. "Graphite: GRAPH-Induced feaTure Extraction for Point Cloud Registration" in: *International Conference on 3D Vision (3DV)*. 2020. (see pp. 213, 311)

[363] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. "Superglue: Learning feature matching with graph neural networks" in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020. 4938–4947 (see p. 213)

[364] Frank Schaeffel, Axel Telljohann, Ingmar Jahr, Karl Lenhardt, Robert Godding, Horst Mattfeldt, Tony Iglesias, Anita Salmon, Johann Scholtz, Robert Hedegore, Julianna Borgendale, Brent Runnels, Nathan McKimpson, Peter Waszkewitz, and Carsten Steger. *Handbook of Machine Vision*. Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, 2008. (see pp. 23, 45)

[365] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. "High-resolution stereo datasets with subpixel-accurate ground truth" in: *German Conference on Pattern Recognition*. Springer 2014. 31–42 (see p. 126)

[366] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, **47**: 7–42, 2002. (see p. 122)

[367] Cedric Scheerlinck, Nick Barnes, and Robert Mahony. "Continuous-time intensity estimation using event cameras" in: *Asian Conference on Computer Vision*. Springer 2018. 308–324 (see p. 238)

[368] Sebastien Schmerber and Fabrice Chassat. Accuracy evaluation of a CAS system: laboratory protocol and results with 6D localizers, and clinical experiences in otorhinolaryngology. *Computer Aided Surgery*, **6**: 1–13, 2001. (see p. 141)

[369] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, **37**: 151–172, 2000. (see p. 212)

[370] David Schubert, Thore Goll, Nikolaus Demmel, Vladyslav Usenko, Jörg Stückler, and Daniel Cremers. "The TUM VI benchmark for evaluating visual-inertial odometry" in: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE 2018. 1680–1687 (see p. 181)

[371] Christian Schulte zu Berge, Artur Grunau, Hossain Mahmud, and Nassir Navab *CAMPVis – a game engine-inspired research framework for medical imaging and visualization* tech. rep. Chair for Computer Aided Medical Procedures, Technische Universität München, 2014 (see pp. 283, 287)

[372] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. "Generalized-icp" in: *Robotics: Science and Systems*. vol. 2 Seattle, WA 2009. 435 (see p. 217)

[373] Jonathan M. Selig. Exponential and Cayley maps for dual quaternions. *Advances in Applied Clifford Algebras*, **20**: 923–936, 2010. (see p. 101)

[374] Ken Shoemake. "Animating rotation with quaternion curves" in: *Proceedings of the 12th annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. ACM Press, 1985. 245–254 (see pp. 93, 239, 244)

[375] Ken Shoemake. "Quaternion calculus and fast animation, computer animation: 3-D motion specification and control" in: *Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. 1987. (see p. 239)

[376] Matheen Siddiqui and Gérard Medioni. "Robust real-time upper body limb detection and tracking" in: *ACM International Workshop on Video Surveillance & Sensor Networks (VSSN)*. 2006. (see p. 188)

[377] Rebecca L. Siegel, Kimberly D. Miller, Stacey A. Fedewa, Dennis J. Ahnen, Reinier G. S. Meester, Afsaneh Barzi, and Ahmedin Jemal. Colorectal cancer statistics, 2017. *CA: a Cancer Journal for Clinicians*, **67**: 177–193, 2017. (see p. 278)

[378] Rebecca L. Siegel, Kimberly D. Miller, and Ahmedin Jemal. Cancer statistics, 2015. *CA: a Cancer Journal for Clinicians*, **65**: 5, 2015. (see p. 278)

[379] Chanop Silpa-Anan and Richard Hartley. "Optimised KD-trees for fast image descriptor matching" in: *Conference on Computer Vision and Pattern Recognition*. IEEE 2008. 1–8 (see p. 213)

[380] Arjun Singh, James Sha, Karthik S. Narayan, Tudor Achim, and Pieter Abbeel. "Bigbird: A large-scale 3d database of object instances" in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2014. 509–516 (see p. 210)

[381] Richard Sinkhorn. A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices. *The Annals of Mathematical Statistics*, **35**: pp. 876–879, 1964. (see pp. 148, 150)

[382] Marcos Slomp, Michihiro Mikamo, Bisser Raytchev, Toru Tamaki, and Kazufumi Kaneda. GPU-based SoftAssign for Maximizing Image Utilization in Photomosaics. *International Journal of Networking and Computing*, **1**: 2011. (see p. 152)

[383] Stephen M. Smith and J. Michael Brady. SUSAN – a new approach to low level image processing. *International Journal of Computer Vision*, **23**: 45–78, 1997. (see p. 212)

[384] Ruben Smits, Herman Bruyninckx, and Erwin Aertbeliën *KDL: Kinematics and dynamics library* 2011 URL: `https://www.orocos.org/kdl.html` (visited on Sept. 3, 2020) (see p. 248)

[385] Nikolai Smolyanskiy, Alexey Kamenev, and Stan Birchfield. "On the importance of stereo for accurate depth estimation: An efficient semi-supervised deep neural network approach" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018. 1007–1015 (see p. 128)

[386] Pierre Soille. *Morphological Image Analysis*. Second Edition Springer-Verlag Berlin Heidelberg, 2004. (see pp. 34, 36)

[387] Hannes Sommer, James Richard Forbes, Roland Siegwart, and Paul Furgale. Continuous-time estimation of attitude using b-splines on lie groups. *Journal of Guidance, Control, and Dynamics*, **39**: 242–261, 2016. (see p. 240)

[388] Hyewon Song, Suwoong Heo, Jiwoo Kang, and Sanghoon Lee. 3D Character Animation: A Brief Review. *Journal of International Society for Simulation Surgery*, **2**: 52–57, 2015. (see p. 239)

[389] Milan Šonka, Vaclav Hlavac, and Roger Boyle. *Image Processing, Analysis, and Machine Vision*. Third Edition Thomson Learning, 2008. (see pp. 32, 44, 62, 130)

[390] Rangaprasad Arun Srivatsan, Gillian T. Rosen, D. Feroze Mohamed Naina, and Howie Choset. Estimating SE (3) elements using a dual-quaternion based linear Kalman filter. *The proceedings of Robotics Science and Systems*, 2016. (see p. 253)

[391] Andrei State, Mark A. Livingston, William F. Garrett, Gentaro Hirota, Mary C. Whitton, Etta D. Pisano, and Henry Fuchs. "Technologies for augmented reality systems: realizing ultrasound-guided needle biopsies" in: *Proceedings of the 23rd annual Conference on Computer Graphics and Interactive Techniques*. 1996. 439–446 (see p. 274)

[392] Carsten Steger *An Unbiased Detector of Curvilinear Structures* tech. rep. FGBV–96–03 Technische Universität München: Forschungsgruppe Bildverstehen (FG BV), Informatik IX, 1996 (see p. 47)

[393] Carsten Steger. An Unbiased Detector of Curvilinear Structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**: 113–125, 1998. (see p. 47)

[394] Carsten Steger. *Unbiased Extraction of Curvilinear Structures from 2D and 3D Images*. PhD thesis. Fakultät für Informatik, Technische Universität München, 1998. (see p. 49)

[395] Carsten Steger, Markus Ulrich, and Christian Wiedemann. *Machine Vision Algorithms and Applications*. Wiley-VCH Verlag GmbH & Co. KGaA, 2008. (see pp. 11–13, 34, 41, 60, 63, 65, 68, 104)

[396] Philipp J. Stolka, Pezhman Foroughi, Matthew Rendina, Clifford R. Weiss, Gregory D. Hager, and Emad M. Boctor. "Needle guidance using handheld stereo vision and projection for ultrasound-based interventions" in: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer 2014. 684–691 (see p. 274)

[397] Eduard Study. Von den Bewegungen und Umlegungen: I. und II. Abhandlung. *Mathematische Annalen*, **39**: 441–565, 1891. (see p. 99)

[398] Ioan A. Sucan, Mark Moll, and Lydia E. Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, **19**: 72–82, 2012. (see p. 185)

[399] Shih-Yu Sun, Matthew Gilbertson, and Brian W. Anthony. "Probe localization for freehand 3D ultrasound by tracking skin features" in: *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer 2014. 365–372 (see p. 193)

[400] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. "Implicit 3D orientation learning for 6D object detection from RGB images" in: *European Conference on Computer Vision*. Springer 2018. 712–729 (see pp. 214, 219)

[401] Aksel Sveier, Adam Leon Kleppe, Lars Tingelstad, and Olav Egeland. Object Detection in Point Clouds Using Conformal Geometric Algebra. *Advances in Applied Clifford Algebras*, 1–16, 2017. (see p. 208)

[402] Toru Tamaki, Miho Abe, Bisser Raytchev, and Kazufumi Kaneda. "Softassign and EM-ICP on GPU" in: *First International Conference on Networking and Computing (ICNC)*. 2010. 179–183 (see p. 152)

[403] David J. Tan and Slobodan Ilic. "Multi-Forest Tracker: A Chameleon in Tracking" in: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2014. (see pp. 217, 218)

[404] Tolga Tasdizen. *Robust and repeatable fitting of implicit polynomial curves to point data sets and to intensity images*. PhD thesis. Brown University, 2001. (see p. 207)

[405] J. J. Tate, V. Lewis, T. Archer, P. G. Guyer, G. T. Royle, and I. Taylor. Ultrasound detection of axillary lymph node metastases in breast cancer. *European Journal of Surgical Oncology: the Journal of the European Society of Surgical Oncology and the British Association of Surgical Oncology*, **15**: 139–141, 1989. (see p. 279)

[406] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. "Cnn-slam: Real-time dense monocular slam with learned depth prediction" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017. 6243–6252 (see p. 4)

[407] Gabriel Taubin. Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **13**: 1115–1138, 1991. (see p. 207)

[408] Alykhan Tejani, Danhang Tang, Rigas Kouskouridas, and Tae-Kyun Kim. "Latent-class hough forests for 3d object detection and pose estimation" in: *European Conference on Computer Vision*. Springer 2014. 462–477 (see pp. 5, 209, 214)

[409] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. Real-time seamless single shot 6D object pose prediction. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. (see pp. 214, 218)

[410] Yurun Tian, Bin Fan, and Fuchao Wu. "L2-net: Deep learning of discriminative patch descriptor in euclidean space" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017. 661–669 (see p. 212)

[411] Paul A. Tipler and Gene Mosca. *Physik für Wissenschaftler und Ingenieure*. Spektrum Lehrbuch Elsevier Spektrum Akademischer Verlag, 2004. (see p. 60)

[412] Beau Tippetts, Dah Jye Lee, Kirt Lillywhite, and James Archibald. Review of stereo vision algorithms and their suitability for resource-limited systems. *Journal of Real-Time Image Processing*, **11**: 5–25, 2016. (see p. 123)

[413] Junichi Tokuda, Gregory S. Fischer, Xenophon Papademetris, Ziv Yaniv, Luis Ibanez, Patrick Cheng, Haiying Liu, Jack Blevins, Jumpei Arata, Alexandra J. Golby, Tina Kapur, Steve Pieper, Everette C. Burdette, Gabor Fichtinger, Clare M. Tempany, and Nobuhiko Hata. OpenIGTLink: an open network protocol for image-guided therapy environment. *The International Journal of Medical Robotics and Computer Assisted Surgery*, **5**: 423–434, 2009. (see pp. 171, 248)

[414] Federico Tombari, Samuele Salti, and Luigi Di Stefano. "Unique signatures of histograms for local surface description" in: *European Conference on Computer Vision*. Springer 2010. 356–369 (see p. 213)

[415] Alessio Tonioni, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. "Unsupervised adaptation for deep stereo" in: *Proceedings of the IEEE International Conference on Computer Vision*. 2017. 1605–1613 (see p. 126)

[416] Alessio Tonioni, Fabio Tosi, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. "Real-time self-adaptive deep stereo" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 195–204 (see p. 126)

[417] Andrea Torsello, Emanuele Rodola, and Andrea Albarelli. "Multiview registration via graph diffusion of dual quaternions" in: *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE 2011. 2441–2448 (see p. 253)

[418] Jonathan Tremblay, Thang To, and Stan Birchfield. "Falling things: A synthetic dataset for 3d object detection and pose estimation" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018. 2038–2041 (see p. 210)

[419] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. "Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects" in: *Conference on Robot Learning*. 2018. 306–316 (see p. 308)

[420] Roberto Tron, René Vidal, and Andreas Terzis. "Distributed pose averaging in camera networks via consensus on SE (3)" in: *Second International Conference on Distributed Smart Cameras (ICDSC)*. IEEE 2008. 1–10 (see p. 98)

[421] Roger Y. Tsai and Reimar K. Lenz. A new technique for fully autonomous and efficient 3 D robotics hand/eye calibration. *IEEE Transactions on Robotics and Automation*, **5**: 345–358, 1989. (see pp. 175, 198, 248)

[422] Yanghai Tsin and Takeo Kanade. "A correlation-based approach to robust point set registration" in: *European Conference on Computer Vision*. Springer 2004. 558–569 (see p. 145)

[423] Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. "Multi-view consistency as supervisory signal for learning shape and pose prediction" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018. 2897–2905 (see p. 266)

[424] Shubham Tulsiani, Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik. "Learning shape abstractions by assembling volumetric primitives" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017. 2635–2643 (see p. 207)

[425] Tinne Tuytelaars and Krystian Mikolajczyk. A survey on local invariant features. *Foundations and Trends in Computer Graphics and Vision*, **3**: 2008. (see p. 212)

[426] Hideaki Uchiyama and Hideo Saito. "Random dot markers" in: *Virtual Reality Conference*. IEEE 2011. 35–38 (see p. 140)

[427] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. "Sparsity invariant cnns" in: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE 2017. 11–20 (see pp. 128, 266)

[428] Scott E. Umbaugh. *Digital Image Processing and Analysis*. Second Edition CRC Press Taylor & Francis Group, 2011. (see p. 38)

[429] Unity Technologies *Unity* `https://unity3d.com/unity/whats-new/2018.3.6` version 2018.3.6f1 Accessed: 2019-06-10 (see p. 225)

[430] Vladyslav Usenko, Jakob Engel, Jörg Stückler, and Daniel Cremers. "Direct visual-inertial odometry with stereo cameras" in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2016. 1885–1892 (see p. 181)

[431] Soichiro Uto, Tokuo Tsuji, Kensuke Harada, Ryo Kurazume, and Tsutomu Hasegawa. "Grasp planning using quadric surface approximation for parallel grippers" in: *International Conference on Robotics and Biomimetics (ROBIO)*. 2013. (see p. 207)

[432] Pinuccia Valagussa, Gianni Bonadonna, and Umberto Veronesi. Patterns of relapse and survival following radical mastectomy. Analysis of 716 consecutive patients. *Cancer*, **41**: 1170–1178, 1978. (see p. 279)

[433] Ashley Varghese, M. Girish Chandra, and Kriti Kumar. "Dual quaternion based IMU and vision fusion framework for mobile augmented reality" in: *Proceedings of the 9th International Symposium on Intelligent Signal Processing (WISP)*. IEEE 2015. 1–6 (see pp. 6, 266)

[434] Yannick Verdie, Kwang Yi, Pascal Fua, and Vincent Lepetit. "Tilde: A temporally invariant learned detector" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015. 5279–5288 (see p. 212)

[435] Sergi Vidal-Sicart and Renato Valdés Olmos. Sentinel node mapping for breast cancer: current situation. *Journal of Oncology*, **2012**: 2012. (see p. 280)

[436] Martijn Visser, Stefano Stramigioli, and Cock Heemskerk. "Cayley-Hamilton for roboticists" in: *International Conference on Intelligent Robots and Systems*. IEEE 2006. 4187–4192 (see p. 100)

[437] Lukas Von Stumberg, Vladyslav Usenko, and Daniel Cremers. "Direct sparse visual-inertial odometry using dynamic marginalization" in: *International Conference on Robotics and Automation (ICRA)*. IEEE 2018. 2510–2517 (see p. 181)

[438] Joost R. van der Vorst, Boudewijn E. Schaafsma, Floris P. R. Verbeek, Merlijn Hutteman, J. Sven D. Mieog, Clemens W. G. M. Lowik, Gerrit-Jan Liefers, John V. Frangioni, Cornelis J. H. van de Velde, and Alexander L. Vahrmeijer. Randomized comparison of near-infrared fluorescence imaging using indocyanine green and 99 m technetium with or without patent blue for the sentinel lymph node procedure in breast cancer patients. *Annals of Surgical Oncology*, **19**: 4104–4111, 2012. (see pp. 4, 280)

[439] Daniel Wagner. "ARToolKitPlus for Pose Tracking on Mobile Devices" in: *Proceedings of the 12th Computer Vision Winter Workshop (CVWW)*. 2007. (see p. 209)

[440] Michael W. Walker, Lejun Shao, and Richard A. Volz. Estimating 3-D location parameters using dual number quaternions. *CVGIP: Image Understanding*, **54**: 358–367, 1991. (see p. 153)

[441] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. "Densefusion: 6d object pose estimation by iterative dense fusion" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 3343–3352 (see pp. 5, 214, 218)

[442] Cheng Wang, Yu Zhao, Jiabin Guo, Ling Pei, Yue Wang, and Haiwei Liu. "NEAR: The NetEase AR Oriented Visual Inertial Dataset" in: *International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE 2019. 366–371 (see pp. 181, 309)

[443] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J. Guibas. "Normalized object coordinate space for category-level 6d object pose and size estimation" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 2642–2651 (see pp. 5, 216, 298)

[444] Rui Wang, Martin Schworer, and Daniel Cremers. "Stereo DSO: Large-scale direct sparse visual odometry with stereo cameras" in: *Proceedings of the IEEE International Conference on Computer Vision*. 2017. 3903–3911 (see p. 196)

[445] Tao Wang and Haibin Ling. Gracker: A graph-based planar object tracker. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **40**: 1494–1501, 2017. (see p. 217)

[446] Jamie Watson, Michael Firman, Gabriel J. Brostow, and Daniyar Turmukhambetov. "Self-Supervised Monocular Depth Hints" in: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2019. 2162–2171 (see p. 127)

[447] Eric W. Weisstein *Ellipse* 2013 URL: http://mathworld.wolfram.com/Ellipse.html (visited on Sept. 13, 2020) (see p. 55)

[448] Greg Welch and Gary Bishop *An introduction to the Kalman filter* tech. rep. University of North Carolina at Chapel Hill, Department of Computer Science, 1995 (see p. 260)

[449] Thomas Wendler, Marco Feuerstein, Joerg Traub, Tobias Lasser, Jakob Vogel, Farhad Daghighian, Sibylle I. Ziegler, and Nassir Navab. "Real-time fusion of ultrasound and gamma probe for navigated localization of liver metastases" in: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer 2007. 252–260 (see pp. 4, 280)

[450] Paul Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis. Harvard University, 1974. (see p. 82)

[451] Timothy Whelan, Mark Levine, Amiram Gafni, Kenneth Sanders, Andrew Willan, Douglas Mirsky, Denise Schnider, David McCready, Susan Reid, Anna Kobylecky, and Kenneth Reed. Mastectomy or Lumpectomy? Helping Women Make Informed Choices. *Journal of Clinical Oncology*, **17**: 1727–1727, 1999. (see p. 279)

[452] Andrew D. Wiles, David G. Thompson, and Donald D. Frantz. "Accuracy assessment and interpretation for optical tracking systems" in: *Medical Imaging 2004: Visualization, Image-Guided Procedures, and Display*. vol. 5367 International Society for Optics and Photonics 2004. 421–432 (see p. 141)

[453] Changchang Wu, Friedrich Fraundorfer, Jan-Michael Frahm, and Marc Pollefeys. "3D model search and pose estimation from single images using VIP features" in: *Conference on Computer Vision and Pattern Recognition Workshops*. IEEE 2008. 1–8 (see p. 212)

[454] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. *Robotics: Science and Systems (RSS)*, 2018. (see pp. 5, 210, 214, 217–219, 222, 224, 226–228, 232, 306, 307)

[455] Junyuan Xie, Ross Girshick, and Ali Farhadi. "Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks" in: *European Conference on Computer Vision*. Springer 2016. 842–857 (see p. 127)

[456] Gang Xu and Zhengyou Zhang. *Epipolar Geometry in Stereo, Motion and Object Recognition*. vol. 6 Kluwer Academic Publishers, 1996. (see pp. 60, 108)

[457] Jiafeng Xu and Karl Henning Halse. Dual Quaternion Variational Integrator for Rigid Body Dynamic Simulation. *arXiv preprint arXiv:1611.00616*, 2016. (see p. 96)

[458] Dong-Ming Yan, Yang Liu, and Wenping Wang. "Quadric surface extraction by variational shape approximation" in: *International Conference on Geometric Modeling and Processing*. Springer 2006. 73–86 (see p. 207)

[459] Jiaolong Yang, Hongdong Li, Dylan Campbell, and Yunde Jia. Go-ICP: A globally optimal solution to 3D ICP point-set registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **38**: 2241–2254, 2015. (see p. 144)

[460] Jiaolong Yang, Hongdong Li, and Yunde Jia. "Go-icp: Solving 3d registration efficiently and globally optimally" in: *Proceedings of the IEEE International Conference on Computer Vision*. 2013. 1457–1464 (see p. 144)

[461] Nan Yang, Rui Wang, Jörg Stückler, and Daniel Cremers. "Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 817–833 (see p. 127)

[462] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. "Mvsnet: Depth inference for unstructured multi-view stereo" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 767–783 (see p. 123)

[463] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. "Lift: Learned invariant feature transform" in: *European Conference on Computer Vision*. Springer 2016. 467–483 (see p. 213)

[464] Zhichao Yin and Jianping Shi. "Geonet: Unsupervised learning of dense depth, optical flow and camera pose" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018. 1983–1992 (see p. 127)

[465] Shaodi You and Diming Zhang. Think locally, fit globally: Robust and fast 3D shape matching via adaptive algebraic fitting. *Neurocomputing*, 2017. (see p. 207)

[466] Carl Yuheng Ren, Victor Prisacariu, David Murray, and Ian Reid. "Star3d: Simultaneous tracking and reconstruction of 3d objects using rgb-d data" in: *Proceedings of the IEEE International Conference on Computer Vision*. 2013. 1561–1568 (see p. 217)

[467] Sangdoo Yun, Jongwon Choi, Youngjoon Yoo, Kimin Yun, and Jin Young Choi. "Action-Decision Networks for Visual Tracking With Deep Reinforcement Learning" in: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017. (see p. 219)

[468] Xiaoping Yun and Eric R. Bachmann. Design, implementation, and experimental results of a quaternion-based Kalman filter for human body motion tracking. *IEEE Transactions on Robotics*, **22**: 1216–1227, 2006. (see p. 253)

[469] Sergey Zakharov, Wadim Kehl, and Slobodan Ilic. "Deceptionnet: Network-driven domain randomization" in: *Proceedings of the IEEE International Conference on Computer Vision*. 2019. 532–541 (see p. 210)

[470] Sergey Zakharov, Benjamin Planche, Ziyan Wu, Andreas Hutter, Harald Kosch, and Slobodan Ilic. "Keep it unreal: Bridging the realism gap for 2.5 d recognition with geometry priors only" in: *International Conference on 3D Vision (3DV)*. IEEE 2018. 1–11 (see p. 210)

[471] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. "DPOD: 6D Pose Object Detector and Refiner" in: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2019. (see pp. 215, 218, 219)

[472] Amir R. Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J. Guibas. "Robust Learning Through Cross-Task Consistency" in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020. 11197–11206 (see p. 266)

[473] Amir R. Zamir, Alexander Sax, William Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. "Taskonomy: Disentangling Task Transfer Learning" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018. (see p. 266)

[474] Jure Zbontar and Yann LeCun. "Computing the stereo matching cost with a convolutional neural network" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015. 1592–1599 (see pp. 123–125)

[475] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. "Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018. 340–349 (see p. 127)

[476] Xiang Zhang, Stephan Fronz, and Nassir Navab. "Visual marker detection and decoding in AR systems: a comparative study" in: *International Symposium on Mixed and Augmented Reality*. 2002. 97–106 (see p. 188)

[477] Yinda Zhang and Thomas Funkhouser. "Deep Depth Completion of a Single RGB-D Image" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018. (see p. 266)

[478] Yinda Zhang, Sameh Khamis, Christoph Rhemann, Julien Valentin, Adarsh Kowdle, Vladimir Tankovich, Michael Schoenberg, Shahram Izadi, Thomas Funkhouser, and Sean Fanello. "ActiveStereoNet: End-to-end self-supervised learning for active stereo systems" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 784–801 (see p. 123)

[479] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, **13**: 119–152, 1994. (see p. 217)

[480] Zhengyou Zhang. Determining the Epipolar Geometry and its Uncertainty: A Review. *International Journal of Computer Vision*, **27**: 161–195, 1998. (see pp. 111, 112)

[481] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**: 2000. (see pp. 59, 198)

[482] Cheng Zhao, Li Sun, Pulak Purkait, Tom Duckett, and Rustam Stolkin. "Learning monocular visual odometry with dense 3D mapping from dense 3D flow" in: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE 2018. 6864–6871 (see p. 127)

[483] Hongwei Zhao, Ding Yuan, Hongmei Zhu, and Jihao Yin. "3-D point cloud normal estimation based on fitting algebraic spheres" in: *International Conference on Image Processing (ICIP)*. IEEE 2016. 2589–2592 (see p. 207)

[484] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. "Deeptam: Deep tracking and mapping" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 822–838 (see p. 127)

[485] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. "Unsupervised learning of depth and ego-motion from video" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017. 1851–1858 (see p. 194)

[486] Tinghui Zhou, Philipp Krahenbuhl, Mathieu Aubry, Qixing Huang, and Alexei A. Efros. "Learning Dense Correspondence via 3D-Guided Cycle Consistency" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016. (see p. 266)

[487] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. "On the continuity of rotation representations in neural networks" in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. 5745–5753 (see p. 214)

[488] Yuliang Zou, Zelun Luo, and Jia-Bin Huang. "Df-net: Unsupervised joint learning of depth and flow using cross-task consistency" in: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. 36–53 (see pp. 128, 266)