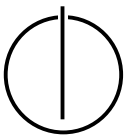
INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN



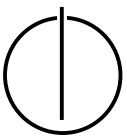
Fast 3D Object Detection and Pose Estimation for Augmented Reality Systems

Dissertation

Seyed Hesameddin NAJAFI SHOUSHTARI

2006

INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN



Fast 3D Object Detection and Pose Estimation for Augmented Reality Systems

Seyed Hesameddin NAJAFI SHOUSHTARI

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:		Univ.-Prof. Dr. Darius Burschka
Prüfer der Dissertation:	1.	Univ.-Prof. Gudrun J. Klinker, Ph.D.
	2.	Univ.-Prof. Nassir Navab, Ph.D.

Die Dissertation wurde am 21. September 2006 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 7. Dezember 2006 angenommen.

Abstract

Many problems in computer vision and augmented reality (AR) require the estimation of the pose of objects or mobile users in real-time. While reliable solutions have been proposed for pose estimation given correspondences and feature-based 3D tracking, fast and fully automated initialization for tracking, i.e. estimation of the initial pose is still an open problem. The difficulty stems from the need for fast and robust detection of known objects in the scene. This dissertation presents a fast and automated object detection and pose estimation system capable of working with large amounts of background clutter, severe occlusions, and strong viewpoint and scale changes.

The thesis builds upon existing algorithms introduced in the last few years and develops various novel techniques which significantly improve both time and functional performance for detection and pose estimation. The advances can be summarized as two main contributions.

First, a method for a scalable, statistical and compact representation of the object of interest is introduced based on fusion of 3D geometric and appearance information. This is achieved during an offline learning process by a statistical analysis and evaluation of distinctive features of the target object. This representation is then used at run-time during the matching and pose estimation processes to limit the number of hypothesis by incorporating both photometric and 3D geometric consistency constraints. This allows to reduce the effect of the complexity of the 3D model on the run-time performance and makes the method especially for large environments very powerful.

The second contribution consists of a novel sensor fusion framework for pose estimation based on a coarse to fine strategy capable of incorporating multiple sensors, e.g. mobile and stationary cameras. This relies on a statistical analysis, probabilistic estimation and fusion of the uncertainties of the sensors.

Furthermore, this system is integrated into an AR tracking framework for the initialization of a marker-less real-time tracking system, which proves to be fast and reliable enough for industrial AR applications.

Zusammenfassung

Viele Probleme in der Computer Vision und erweiterter Realität (AR) benötigen die Berechnung der Position und Orientierung von Objekten oder mobilen Benutzern in Echtzeit. Während hierfür viele präzise Trackingsysteme in den letzten Jahren hinweg entwickelt wurden, bleibt schnelle und völlig automatisierte Initialisierung und Reinitialisierung der Systeme noch ein offenes Problem. Die Hauptschwierigkeit is eine schnelle und robuste Detektion der schon bekannten Objekte aus der Szene. Diese Dissertation stellt ein schnelles, völlig automatisiertes und vor allem skalierbares Objekdetektionssystem vor, welches im Stande ist, mit starken Blickwinkel- und Skalierungsänderungen, partielle Verdeckungen und grossen Mengen von Hintergrundclutter zurecht zu kommen.

Die Arbeit baut auf bereits vorhandene Algorithmen auf, die in den letzten Jahren eingeführt wurden, und schlägt verschiedene neue Methoden vor, die sowohl die Zeit-, als auch die funktionelle Performance entscheidend verbessern. Die Weiterentwicklungen beinhalten unter anderem eine skalierbare und statistische Objektrepräsentation, die auf einer Fusion der 3-dimnsionalen Form und Erscheinungsmodelle basiert, ein Sensorfusionssystem, das verschiende Sensoren integriert, und ein Tracking Management System, das gezeigt hat, dass es für industrielle AR Anwendungen schnell und verlässlich genug ist.

Acknowledgments

First and foremost, I wish to thank my supervisors Prof. Gudrun Klinker, Ph.D. and Prof. Dr. Nassir Navab who have both continuously shared their support and enthusiasm. Working with them has been a great pleasure. I gratefully acknowledge Gudrun's constant and invaluable academic and personal support and guidance throughout the last years. I thank Nassir for sharing with me his unique way of looking at things and approaching a research problem. He has been a constant source of inspiration during this endeavor.

I have spent the last year and a half of my Ph.D. working at Siemens Corporate Research Center in Princeton, NJ. I am thankful to Yakup Genc, Ph.D. for giving me the opportunity to work in the 3D Vision and Augmented Reality Group. I greatly appreciate his insightful comments, his constructive criticism and his valuable suggestions. I also wish to thank Dr. Visvanathan Ramesh for the fruitful discussions during the course of this work. I have enjoyed many useful discussions with some of my present and former colleagues at the Real-time Vision Department, in particular, Yanghai Tsin, Ph.D., Xiang Zhang, Ph.D., Matthias Voigt, Anurag Mittal, Ph.D., Jan Neumann, Ph.D., Ying Zhu, Ph.D., Maneesh Singh, Ph.D., Dr. Claus Bahlmann, and Ali Khamene, Ph.D. I thank my colleague Mehdi Hamadou from Siemens A&D for the very interesting conversations and his valuable feedbacks and support in the ARTESAS project.

I would like to say a big 'thank-you' to all my former colleagues from the Augmented Reality Research Group at Technische Universität München (TUM), in particular, Martin Bauer, Verena Broy, Stephan Huber, Dr. Asa MacWilliams, Dr. Thomas Reicher, Dr. Christian Sandor, and Dr. Martin Wagner. It has been a great pleasure for me to work with you and I wish you all the best for your academic and professional pursuits. I also thank Prof. Bernd Brügge, Ph.D., for providing the opportunity to work at his chair.

Special thanks to all the students I have been working with at TUM, in particular, Stefan Hinterstoisser, Fabian Sturm, Christian Trübswetter, Franz Mader, Andreas Haug, and Musafa Isic.

A great thank you to my friends from the CAMP+AR Group at TUM, Dr. Selim BenHimane, Florian Echtler, Jörg Traub, Marco Feuerstein, Martin Groher, Hauke Heibel, Martin Horn, Daniel Pustka, Tobias Sielhorst, and Wolfgang Wein.

There are many other people that I will not list, with whom I have had useful and inspiring conversations about my work and related topics.

During the course of this work, I have been supported by a scholarship funded from Siemens Corporate Research. I gratefully acknowledge this support.

This thesis is, if indirectly, the fruit of my parents. I would like to thank them for their endless patience, continual support, and unflagging empathy and encouragement. My mother has always supported my dreams and aspirations. I would like to thank her

for all she is, and all she has done for me. Finally, my heartfelt thanks go to my father, Dr. S.M. Bagher Najafi-S. who gave everything for his children and their education and always encouraged them to pursue the quest for careers in science. He lives in my heart forever. I dedicate this thesis to him.

Contents

List of Figures

Chapter 1

Introduction

No problem is too small or too trivial if we can really do something about it.

– Richard Feynman (1918 – 1988)

1.1 Motivation

The problem addressed in this thesis is the automatic recovery of the three-dimensional pose of an object of interest from a single image.

Tracking objects through image sequences is one of the fundamental problems in computer vision, and estimating the motion and pose has been an area of research for many years. In many recent applications ranging from Robot Navigation, Surveillance to Augmented Reality real-time performance is of critical importance. Many reliable solutions have been proposed for real-time pose estimation given correspondences [30, 113, 109, 48] and feature-based 3D tracking [146, 43, 28]. Traditional tracking approaches make use of a strong prior on the pose for each new frame. Imposing temporal continuity constraints across frames increases the quality and robustness of the results. During the last years vision based tracking systems have reached a maturity level capable of tracking complex 3D objects very accurately [60, 32, 58, 59].

However, in practice robust object tracking in real-time is quite challenging and remains an open problem. Reasons for that are manifold. First of all tracking systems require an initialization, i.e. providing the system with the initial pose of the object or camera. Once tracking is started, rapid motions causes image features to undergo large motion between frames which can cause visual tracking systems to fail. Furthermore, it can cause motion blurred images where extraction of feature correspondences may fail. The lighting during a shot can change significantly; reflections and specularities may confuse the tracker. Finally, complete or partial occlusions as well as large amounts of background clutter may result in tracking failure. Once tracking fails the system must be re-initialized in the same manner. Therefore, one of the main challenges in tracking remains automated initialization and recovery. Systems that are initialized by hand or require the camera to be very close to a specified position are not desirable options.

The lack of a fast, reliable and automated initialization method is one of the main crucial problems that make many current tracking systems useless in the industry as well as in computer aided minimally invasive surgery. This has led to increased popularity of fiducial- or marker-based tracking system such as ARToolKit [56], or A.R.T. system [6], where the 3D tracking task is simplified to overcome the tracking limitations by detecting predefined artificial markers in every frame independently without constrains on camera pose. Using markers increases robustness and reduces computational requirements. However, it requires engineering the environment, which is not accepted in many industrial applications by the end-users and is sometimes even impossible, e.g. in outdoor environments. This has led to the development of systems in hybrid configurations involving expensive magnetic or inertial trackers [60].

The difficulty of overcoming the vision based tracking limitations as described above stems from the need for fast and robust detection and pose estimation of objects in the scene from a single image without priors on the pose.

The main aim of this thesis is to study the problem of automated initialization in particular for Augmented Reality systems. Next section gives a brief introduction into this field. In the following sections the problem definition is given in that context and some industrial applications are described.

1.1.1 An introduction to Augmented Reality

Augmented reality (AR) is a technology by which a user's perception of the real world is augmented with additional information generated from a computer model in real time [135, 21, 9, 8]. The visual enhancements may include virtual explanatory labels, three-dimensional rendered models, and shading and illumination changes. AR allows a user to work with and examine real objects while receiving additional information about them through a display. Augmented Reality can be applied in many fields including sectors of industry, e.g. for repairing and maintenance of complex machines and facilities, visualizing sensor or simulation data, and for interior or exterior design. Other fields of applications include medical applications such as computer aided minimally invasive surgery.

The displays for viewing the merged virtual and real environments can be classified into three categories: head mounted, hand held, and projective displays [9, 8]. Independent of which display technology is used, one of the most crucial problems that remains challenging to this day is the *registration problem*. The objects in the real and virtual worlds must be

properly aligned with respect to each other, or the illusion that the two worlds coexist will be compromised. Many applications such as industrial or medical applications demand accurate registration. For example, if the instruction to click a button is not shown on the right button in the machine, it would be useless. More seriously, in the needle biopsy application [104], if the virtual object is not where the real tumor is, the surgeon will miss the tumor and the biopsy will fail. Without accurate registration, Augmented Reality will not be accepted in many applications.

In the case of using a head mounted display (HMD), a common approach to registration is to mount a sensor on the user's head. Assuming the position of the sensor relative to the display is fixed, registration may be split into two parts: an accurate calibration of each eye's display relative to the sensor, and robust and accurate *tracking* of the sensor's position in the world. The *tracking problem* for AR is defined as the task of estimating the 6 DOF *pose*, i.e. three-dimensional position and orientation, of an object, e.g. the HMD, in a given coordinate system.

A wide variety of tracking technologies have been applied to AR and are described in recent surveys [118, 152]. Many tracking technologies have been used in AR applications. These include mechanical, magnetic, ultrasound, inertial, vision-based and hybrid trackers. Magnetic and ultrasound sensors have been used successfully, but confine the user to an instrumented working volume which may be very small. Magnetic trackers are vulnerable to distortion by metal in the environment. Ultrasound trackers suffer from noise and are difficult to make accurate at long ranges because of variations in the ambient temperature. Inertial trackers drift with time.

Vision based trackers such as mobile cameras on the other hand can operate without external beacons, and video capture capability is becoming a standard feature in off-the-shelf PCs. Consequently, the use of cameras as sensors for tracking has been the subject of substantial research and many reliable solutions have been proposed in the literature.

1.1.2 Visual tracking for Augmented Reality

Tracking mobile users and objects is a central part of every augmented reality system. Among many available tracking technologies the vision-based tracking systems are in many application fields the method of choice due to their accuracy as well as flexibility and ease of use. Many tracking techniques have been proposed that allow alignment of real and virtual worlds in real-time using images acquired by a camera.

Some of the vision-based trackers used in AR are based on tracking of a set of predefined patterns, artificial landmarks or fiducial markers (see Figure 1.1(a)-(c)). These markers need to be placed in the environment and calibrated, i.e. their three dimensional positions have to be computed. Each marker with its respective predefined pattern is then detected in the camera images and used to estimate the pose of the camera.

The use of markers increases robustness and reduces computational requirements. However, their use can be complicated as they require certain maintenance. For instance, placing a marker in the workspace of the user can be intrusive and the markers may from time to time need re-calibration. This is in particular within industrial environments not easily accepted by the end users. Therefore, direct use of scene features for tracking instead of the markers is much desirable, especially when certain parts of the workspace do not change in time. For example, a control panel, as shown in see Figure 1.2(a), has

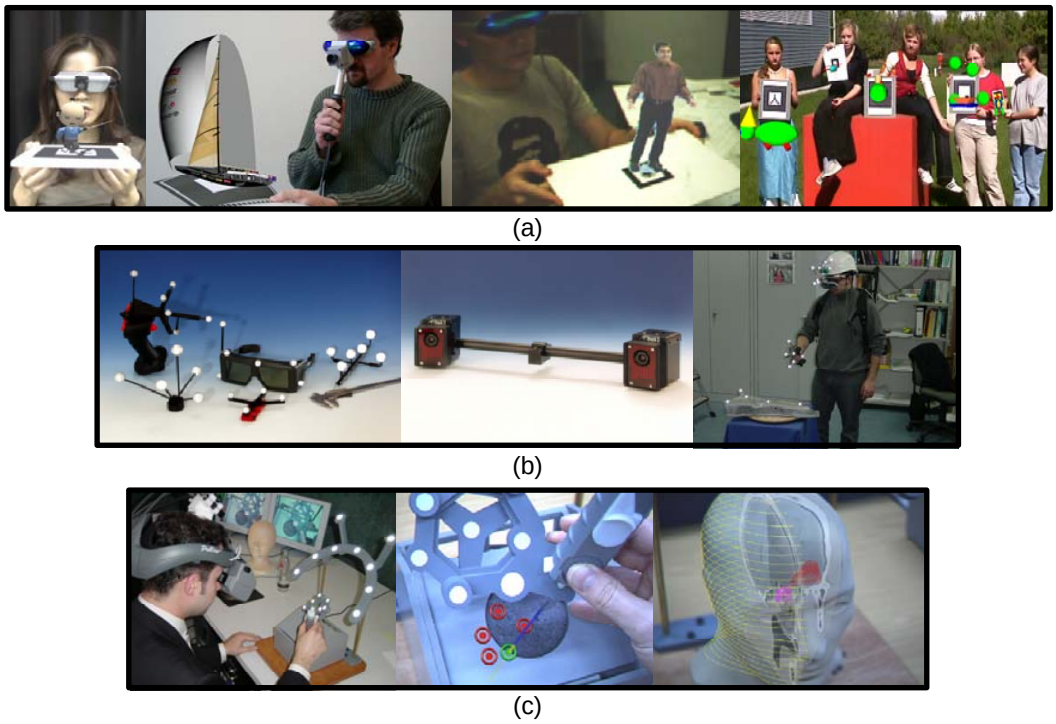


Figure 1.1: Marker-based tracking systems. (a) ARToolKit [56], courtesy of M. Billinghurst. (b) A.R.T. tracking system [6]. (c) RAMP system [123].

fixed buttons and knobs that remain the same over its lifetime. The use of these rigid and unchanging features for tracking simplifies the preparation of the scenarios for scene augmentation as well.

1.1.3 Marker-less tracking and the initialization problem

The marker-less tracking systems rely on natural features present on the object of interest or in the scene, such as corner points, edges, line segments, conics, blobs or regions, cylindrical objects [103], etc. There have been some efforts addressing marker-less tracking in the computer vision literature [43, 146, 23, 100, 132, 108]. The main advantages of these marker-less tracking approaches are their accuracy and speed which is required for AR applications. However, in practice robust object tracking in real-time is difficult and remains an open problem. Reasons for that are manifold. First of all all tracking approaches do require an initialization, i.e. providing the system with the initial pose of the object or camera. Once tracking is started, rapid camera or object movements causes image features to undergo large motion between frames which, can cause visual tracking systems to fail. Furthermore, it can cause motion blurred images where extraction of feature correspondences may fail. The lighting during a shot can change significantly; reflections and specularities may confuse the tracker. Finally, complete or partial occlusions as well as large amounts of background clutter may result in tracking failure. On the other side, the optimization or minimization algorithm may be subject to local minimum or divergence.

Figure 1.3 shows an example of an edge-based tracker that fails due to fast movements

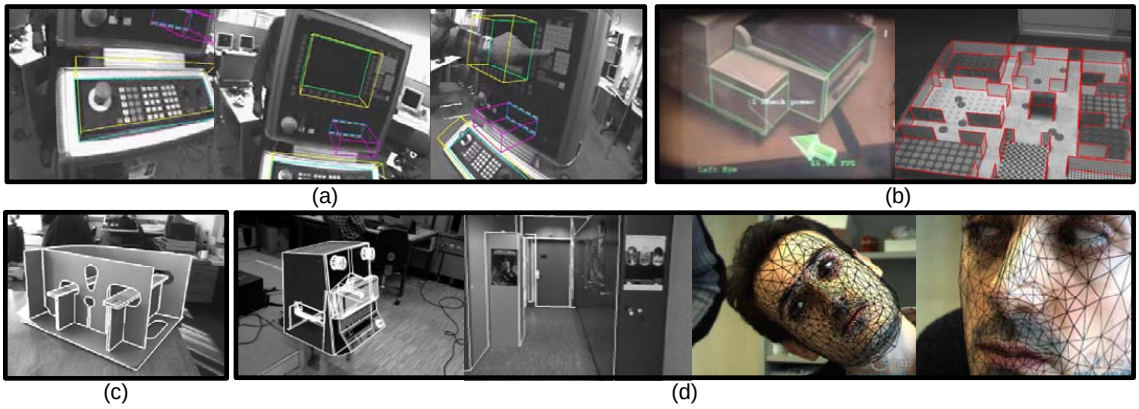


Figure 1.2: Marker-less tracking systems. (a) Genc et al [43]. (b) Klein [58], courtesy of T. Drummond. (c) Drummond et al [32], courtesy of T. Drummond and R. Cipolla. (d) Vachetti et al [146], courtesy of P. Fua.

causing motion blur. Once the tracker has failed, a re-initialization is required to continue tracking. As a consequence, a main crucial problem of the current marker-less tracking systems is the *automated initialization*, i.e. providing the system automatically with the initial pose of the user's view or camera. This procedure needs to be done each time the system looses track. This has led to the development of systems in hybrid configurations involving expensive magnetic or inertial trackers [59]. However, in this thesis we address the problem of automatic initialization based on camera images only.

The lack of a fast, reliable and automated initialization method is one of the main crucial problems that make many current marker-less tracking systems useless in the industry as well as in computer aided minimally invasive surgery. In our particular case, namely industrial applications of augmented reality, where the user wears a head mounted camera, this problem becomes very challenging. Due to fast head movements of user's head while working in a collaborative industrial environment and e.g. talking to co-workers, frequent and fast initialization is required. The main aim of this thesis is to study the problem of maintaining an estimate of the relative position between a monocular camera sensor and the position of 3D objects via the use of visual information.

1.1.4 Applications

This section describes some industrial applications of AR. Other potential AR applications that have been explored include medical applications, robot path planning, annotation, entertainment, and military aircraft navigation and targeting. Some medical AR projects the author has been involved in are described in appendix A.

Industrial AR

AR has been applied to a few sectors of industry mainly in order to improve manual work progress. One of the first industrial AR applications, the AR wire bundle assembly has been developed at Boing by David Mizell [21]. This project aimed at making the use of prefabricated guidance board for wiring obsolete by using optical see-through HMDs and

Figure 1.3: The initialization problem. Each time tracking is lost, e.g. due to fast camera motion causing blurred images, the tracking system needs to be initialized again.

wearable computers instead. The largest industrial AR (IAR) consortium, ARVIKA, was funded by Germany's Federal Ministry of Education and Research organization, which supported the Augmented Reality for Development, Production, and Servicing. In the ARVIKA project many solutions for using AR in manufacturing, similar to the wire bundle assembly have been proposed. But unfortunately most of them have stalled at the prototype level, and have not found regular use in production units yet. One of the few solutions for manufacturing that has been accepted by the end-users and installed in real industrial setting is the *intelligent welding gun* (Echtler et al [34]), developed in the PAARTI project by TU Munich for BMW within the ARVIKA consortium. The system was designed to help welders in the automotive industry shoot studs with high precision in the body of early automobile prototypes. The intelligent welding gun is a regular gun with a display attachment, a few buttons for user interactions, and reflective markers to track the gun position and orientation from stationary cameras (see Figure 1.4 (a)). While welders operate and move the gun, the display shows three-dimensional stud locations on the car frame relative to the current gun position. Navigational metaphors, such as notch and bead and a compass, are used to help workers place the gun at the planned stud positions with required precision. See [34] for more detail.

In the FixIt [33, 63] project, the objective was assisting workers in diagnosing machine malfunctions using AR. The idea is that by overlaying virtual information of a control system directly onto the machine while it is in operation, AR has the potential to help workers obtain a better understanding of the reasons for malfunctions. This will be described in more details in section D.

One of the other very promising application areas of Augmented Reality (AR) is the design of new products, such as cars or buildings [61]. Klinker et al describe in [62] the efforts in the project *Fata Morgana* to develop a proof-of-concept AR system for car designers. Figure 1.4(b) shows a demonstration system for presenting cars on a designer's desk.

Another branch of industrial applications involves integrating AR into the monitoring and control processes in industrial settings. Nassir Navab developed with his colleagues at

Figure 1.4: Industrial AR Applications. (a) The intelligent welding gun [34]. (b) Augmented desktop of a car modeler [62].

Siemens a system that aims at integrating 3D models, industrial drawings, factory images, AR visualization, computer-assisted tracking and localization, wireless communication and data access, and speech-based interaction to provide end-to-end solutions that empower wireless workers [106]. The system is based on an IAR software solution coined CyliCon that has been created for as built reconstruction of industrial pipelines (see Figure 1.5 (a)). Thereby virtual models of the areas of interest are reconstructed using AR techniques and augmented along with industrial drawings on the views of the real scene. These so called COP (*coregistered orthographic and perspective*) images [102, 5] make the design of new calibration and 3D reconstruction algorithms possible while reducing the need for calibration markers [105, 4]. Combining this solution with a user localization and tracking system in large environments, provides a very reliable, user-friendly, cost-effective and most important scalable solution for AR visualization and data access (see Figure 1.5 (b)).

Chapter 7 will discuss further IAR applications of the proposed detection and pose estimation system.

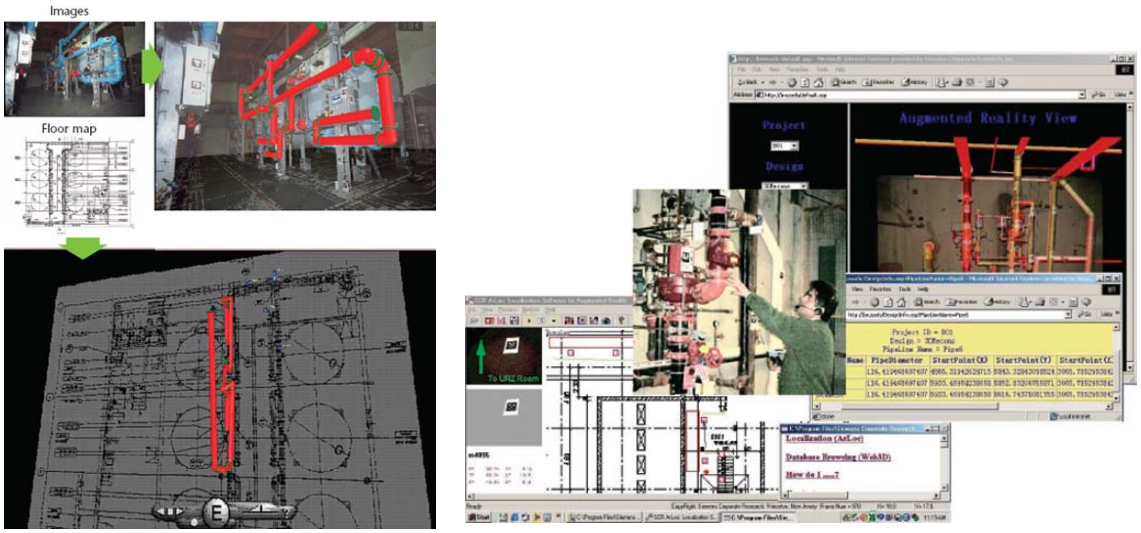
1.2 Thesis overview

This section states the main contributions of this thesis, and gives a brief outline of the following chapters.

1.2.1 Contributions

The main contributions of this thesis are

- A novel sensor fusion framework for pose estimation based on a coarse to fine strategy capable of incorporating multiple sensors (Chapter 4). It is not limited in tracking range and working environment, given a 3D model of the objects or the real scene. This is achieved based on a statistical analysis and probabilistic estimation of the uncertainties of the tracking sensors. The explicit representation of the error distribution allows the fusion of different sensor data in order to estimate the initial pose and improve the registration accuracy. Although the methodology presented here is not restricted to a particular tracking technology, for this work it was applied to an augmented reality system, using a mobile camera and several stationary tracking sensors, and can be easily extended to the case of any additional sensor. In order to



(a) (b)

Figure 1.5: CyliCon System [106] provides beside as-built reconstruction functionalities, augmented reality visualization of images, drawings, and virtual models (a). Mobile augmented reality allows the user to access the data through augmented views of the real environment (b).

solve the initialization problem, we adapt, modify and integrate advanced techniques such as plenoptic viewing, intensity-based registration, and ICP. Thereby the registration error is minimized in 3D object space rather than in 2D image. Experimental results show how complex objects can be registered efficiently and accurately to a single image.

- A fast, robust and automated object detection and pose estimation system which counters problems related to the limited repeatability of the feature detectors, and the difficulty of matching, in the presence of large amounts of background clutter, severe occlusions and particularly challenging viewing conditions (Chapters 5 and 6). We propose a method that conducts a statistical analysis of the appearance of object features from all possible viewpoints in the scene and incorporates the 3D geometry during both matching and the pose estimation processes. Thereby the appearance information from the 3D model and real images are combined with synthesized images in order to learn the variations in the multiple view feature descriptors using PCA. Furthermore, by analyzing the computed visibility distribution of each patch from different viewpoints, a reliability measure for each patch is estimated. This reliability measure is used to further constrain the classification problem. This results in a scalable representation reducing the effect of the complexity of the 3D model on the run-time matching performance. Moreover, as required in many real-time applications this approach can yield a reliability measure for the estimated pose. Experimental results show how the pose of complex objects can be estimated efficiently from a single test image (Chapter 6).

- Transfer to Industry: A final contribution of this thesis involves the transfer of

marker-less augmented reality technology to industry. Our object detection system has been integrated into an AR tracking framework for the initialization of a marker-less real-time tracking system. We introduce a tracking management framework designed for single and multiple object tracking which has proved to be fast and reliable enough for industrial AR applications (Chapter 7). This was carried out within the ARTESAS project funded by the German Ministry for Education and Research and supervised by the DLR (German Aerospace Center). ARTESAS aims at the exploration and evaluation of Augmented Reality base technologies for applications in industrial service environments.

1.2.2 Organization of the thesis

The structure of the thesis is as follows. First the mathematical framework and problem definition is given in chapter 2. In chapter 3 we discuss the related work and summarize the state of the art. Chapter 4 introduces a sensor fusion framework for automated initialization of marker-less tracking systems. A brief introduction to the problem of finding corresponding features, and the basic ideas behind the scale invariant and affine covariant features are given in chapter 5. Chapter 6 presents a method for fast and robust object detection and pose estimation by fusing both the underlying 3D and appearance models. In chapter 7 we discuss further experimental results and potential industrial AR applications of the object detection system. Moreover, we describe the tools that have been developed for the offline training process. Finally, the integration of our detection system into an AR tracking framework is described. Thereby, we introduce a tracking management framework designed for single as well as multiple object tracking. Conclusions, limitations as well as future work are provided in chapter 8.

Since every chapter is relatively self-contained, it directly includes the related results, followed by a discussion and conclusion. Thus, the reader interested in only one technique should be able to understand it by considering a single chapter.

Chapter 2

Mathematical Framework

There is no "royal road" to geometry.

– Euclid (325–265 BC), to king Ptolemy I

This chapter describes the geometry of a camera and introduces the mathematical problem definition. Throughout this thesis, we will refer to *internal camera calibration* as the process of determining the internal camera geometric and optical characteristics (*internal parameters*), such as the focal length and aspect ratio. The *external camera calibration* or *3D pose estimation* refers to determining the 3D position and orientation of the camera frame relative to a certain world or object coordinate system (*external parameters*). Next section describes a perspective camera model. A simple camera calibration algorithm is briefly reviewed in section 2.2. For textbooks on this topic, the reader is referred to [48, 35, 45].

2.1 Camera geometry

A camera is a mapping between the 3D world or object space and a 2D image [48]. One of the most common geometric camera models used in computer vision for CCD like sensors is the *pinhole model* which is the subject of the next section.

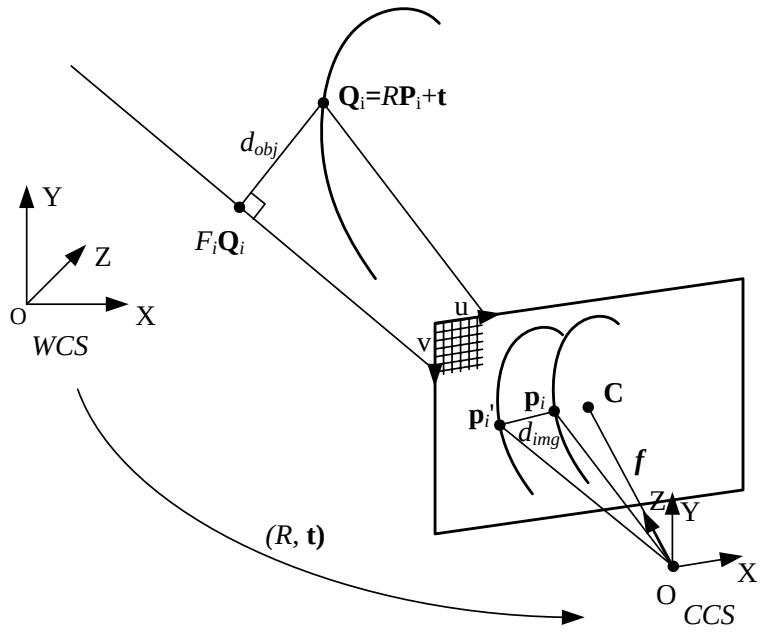


Figure 2.1: Pinhole camera geometry. The collinearity errors in object and image space are denotes as d_{obj} and d_img, respectively.

2.1.1 Pinhole camera model

Figure 2.1 illustrates the basic geometry of the pinhole camera model. We choose the projection center of the camera as the origin of an Euclidean coordinate system called the *camera coordinate system* (CCS) with the optical axis pointing in the positive z direction. Under the pinhole camera model, an object point with the 3D coordinates $(X, Y, Z)^T$ is mapped to the point $(fX/Z, fY/Z)^T$ on the image plane $z = f$ where a line joining the object point to the center of projection meets the image plane. This describes the central projection mapping from object to image coordinates. The parameter f is the *focal length* of the camera. If this parameter is known, for the sake of simplicity we consider the *normalized image plane* with $z = f = 1$ in the camera coordinate system.

In general, points in space will be expressed in terms of a different Euclidean coordinate frame, known as the *world coordinate system* (WCS). Given a set $\{\mathbf{P}_i | 0 < i \leq n\}$ of n object points $\mathbf{P}_i = (X_i, Y_i, Z_i)^t$, in the world coordinate system, the set of corresponding coordinates $\mathbf{Q}_i = (X_i, Y_i, Z_i)^t$ in the camera coordinate system, are related by a rigid transformation

$$\mathbf{Q}_i = R\mathbf{P}_i + \mathbf{t},$$

where $R = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]^t$ is a 3×3 rotation matrix and $\mathbf{t} = (t_x, t_y, t_z)^t$ is a translation vector.

The image point $\mathbf{p}_i = (u_i, v_i, 1)^t$ is the perspective projection of the object point $\mathbf{P}_i$ to the normalized image plane according to the following equation.

$$\mathbf{p}_i = \frac{R\mathbf{P}_i + \mathbf{t}}{\mathbf{r}_3\mathbf{P}_i + t_z}. \tag{2.1}$$

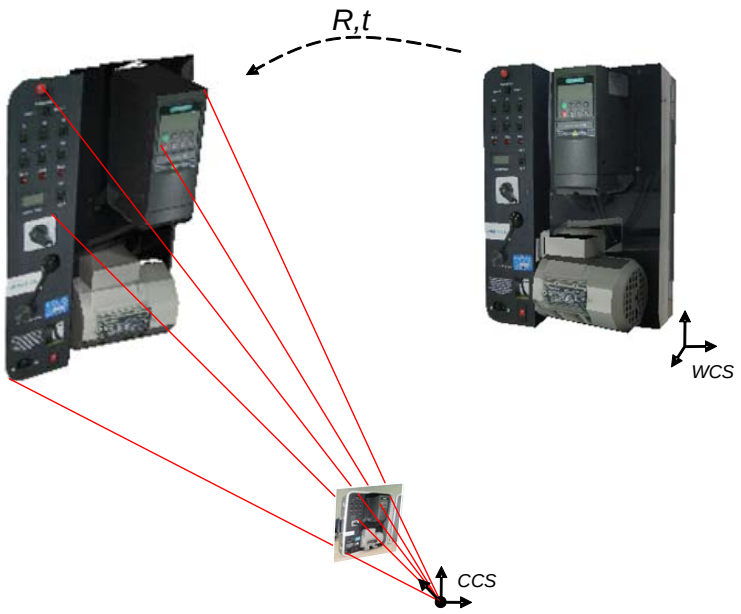


Figure 2.2: The 2D-3D pose estimation problem is to find the rigid transformation (R, t) which leads to a best fit between the object and the image data.

This equation is called the *collinearity equation* and says that $\mathbf{p}_i$, $\mathbf{Q}_i$ and the projection center of the camera $\mathbf{O}$ are collinear.

However, another way of thinking of collinearity is that the orthogonal projection of $\mathbf{Q}_i$ on the projection ray of $\mathbf{p}_i$ is equal to $\mathbf{Q}_i$ itself. This can be formulated as

$$R\mathbf{P}_i + \mathbf{t} = F_i(R\mathbf{P}_i + \mathbf{t}), \tag{2.2}$$

where F_i is a projection operator [82] and is defined as

$$F_i = \frac{\mathbf{p}_i\mathbf{p}_i^t}{\mathbf{p}_i^t\mathbf{p}_i} = \frac{1}{\|\mathbf{p}_i\|^2}\begin{pmatrix} u_i^2 & u_iv_i & u_i \\ u_iv_i & v_i^2 & v_i \\ u_i & v_i & 1 \end{pmatrix}. \tag{2.3}$$

We refer to (2.1) as the *image space collinearity equation* and (2.2) as the *object space collinearity equation*. The parameters of the rigid transform $(R, \mathbf{t})$ which relate the camera orientation and position to a world coordinate system are called the *external parameters* or the *exterior orientation*.

The *2D-3D pose estimation problem* is to find the rigid transform $(R, \mathbf{t})$ that best fits the known 3D object data with the observed 2D image data (see Figure 2.2). Usually this is achieved by minimizing some form of accumulation of errors (least squares methods) based on one of the collinearity equations in object or image space.

An alternative representation of the pose is by a six element vector $\mathbf{s} = (t_x, t_y, t_z,$-$\theta, \phi, \psi)^t$ containing the three translational parameters and three angles of rotation around the three main axes. Equivalently quaternion representation can be used for orientation.

The line from the projection center perpendicular to the image plane is called the *principal axis*, and the point where this axis meets the image plane is called the *principal*

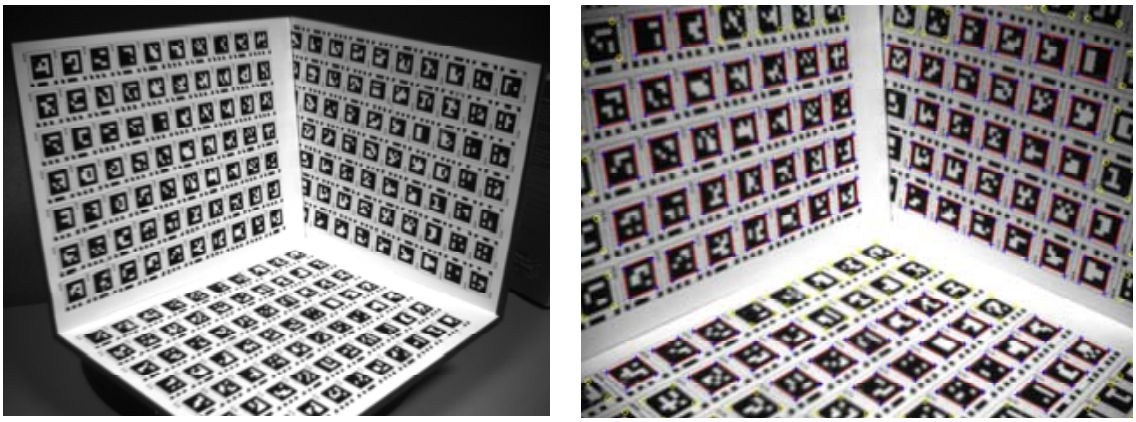


Figure 2.3: Camera calibration: (a) The calibration grid. (b) Camera image where the markers are detected using a marker detection system.

point. So far we have assumed that the origin of coordinates in image plane is at the principal point. This may not be the case in practice. In general, a shift is needed to the coordinates $(c_x, c_y)^T$ of the principal point $\mathbf{C}$ (see Figure 2.1).

Furthermore, we have assumed that the image coordinates are Euclidean coordinates having equal scales in both axial directions. However, if CCD cameras are used, the pixels might be non-square. We therefore need to take unequal scale factors in each direction into account. Suppose m_x and m_y are the number of pixels per unit distance in image coordinates. Then, the transformation from the world coordinates to pixel coordinates is obtained by multiplying 2.1 on the left by the following so called calibration matrix

$$K = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}, \tag{2.4}$$

where $f_x = fm_x$ and $f_y = fm_y$ is the focal length of the camera in terms of pixel dimensions in both directions, respectively. The parameter s is the skew parameter.

Using the notation P for the world point represented in homogeneous 4-vector $(X, Y, Z, 1)$, and p for the image point represented by a homogeneous 3-vector, then we can write 2.1 in a compact way as

$$\mathbf{p} = M\mathbf{P}, \tag{2.5}$$

with M is the *camera projection matrix*

$$M = K[R|\mathbf{t}]. \tag{2.6}$$

2.2 Camera calibration

The problem of camera calibration is to compute the camera projection matrix, i.e. intrinsic and extrinsic parameters based on a number of points whose object coordinates

in the WCS are known and whose image coordinates are measured. For this purpose we use a single camera image taken from a calibration grid as shown in Figure 2.3(a). The calibration grid consists of three flat planes covered by a set of coded visual markers (Huffman markers) whose 3D coordinates are pre-measured accurately and recorded. Given a single image from the marker grid each marker is detected automatically using a marker detection system. Having sufficiently many 3D-2D correspondences the camera matrix M may be determined. The internal parameters of K can then be extracted from the matrix M by decomposition.

2.2.1 Estimating the camera matrix

Given a number of point correspondences $\mathbf{P}_i \leftrightarrow \mathbf{p}_i$ between 3D points and 2D image points, we are looking for a 3x4 camera matrix M, such that $\mathbf{p}_i = M\mathbf{P}_i$, $\forall i$. In order to determine M different cost functions may be minimized. One solution is based on the measurement of geometric distance in the image, and minimization of the difference between the measured and estimated image coordinates. The geometric error in the image is $\sum_i d(\mathbf{p}_i, \hat{\mathbf{p}}_i)^2$, where d represents the Euclidean distance between two point and $\hat{\mathbf{p}}_i$ is the point $M\mathbf{P}_i$. Assuming that the measurement errors are Gaussian the solution to

$$\min_M \sum_i d(\mathbf{p}_i, M\mathbf{P}_i)^2 \tag{2.7}$$

is the Maximum Likelihood estimate of M. The minimization of this non-linear error function can be achieved with iterative techniques, such as Levenberg-Marquardt [48]. However, as a starting point for the iterative minimization an initial solution is required. A minimal solution can be obtained by using a linear method for minimizing an algebraic error.

For this purpose, the equation (2.5) can be written in terms of the vector cross product as $\mathbf{p}_i \times M\mathbf{P}_i = 0$. This form enables a simple linear solution for M to be derived. This can be written explicitly as

$$\mathbf{p}_i \times M\mathbf{P}_i = \begin{pmatrix} v_i\mathbf{m}^{3t}\mathbf{P}_i - \mathbf{m}^{2T}\mathbf{P}_i \\ \mathbf{m}^{1T}\mathbf{P}_i - u_i\mathbf{m}^{3T}\mathbf{P}_i \\ u_i\mathbf{m}^{2T}\mathbf{P}_i - v_i\mathbf{m}^{1T}\mathbf{P}_i \end{pmatrix} \tag{2.8}$$

where $\mathbf{m}^{jT}$ denotes the j-th row of the matrix M. Since this matrix M has 12 entries and ignoring the scale 11 degrees of freedom, 11 equations are required to solve for M. Each point correspondence leads to three linearly dependent equations. Choosing the first two yields

$$\begin{pmatrix} \mathbf{O}^T & -\mathbf{P}_i^T & v_i\mathbf{P}_i^T \\ \mathbf{P}_i^T & \mathbf{O}^T & -u_i\mathbf{P}_i^T \end{pmatrix} \begin{pmatrix} \mathbf{m}^1 \\ \mathbf{m}^2 \\ \mathbf{m}^3 \end{pmatrix} = 0. \tag{2.9}$$

For a set of n point correspondences we obtain a $2n \times 12$ matrix A by stacking up the equations (2.9) for each correspondence. This gives a linear equation system of the form $A\mathbf{m} = 0$, where $\mathbf{m}$ is the vector containing the 12 entries of the camera matrix M. The camera matrix M can be obtained by minimizing the residual $||A\mathbf{m}||$ called the *algebraic*

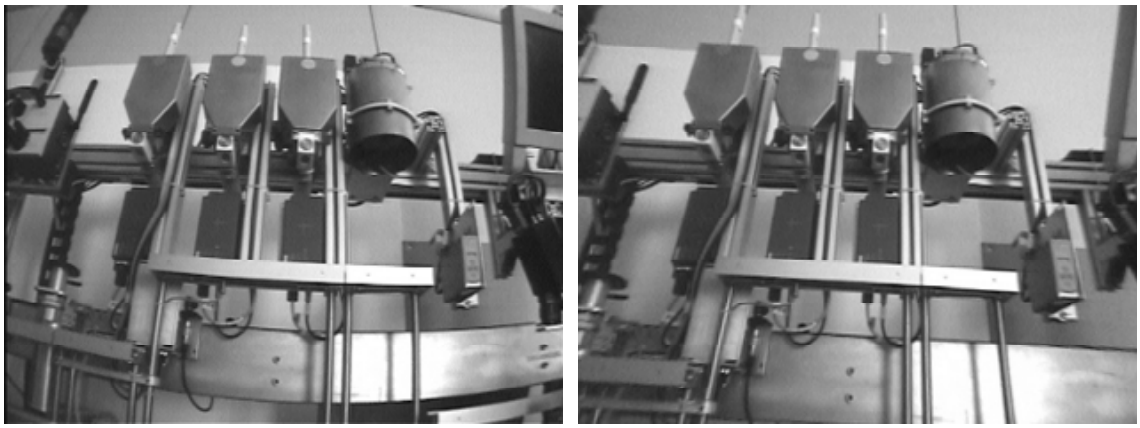


Figure 2.4: Radial distortion. (a) The oroginal image with significant radial distortion. (b) The image warped to remove the radial distortion. Note that the curved lines in (a) which are straight on the object are now straight.

error subject to some normalization constraint, e.g. $||\mathbf{m}|| = 1$. A solution is obtained from the unit singular vector of A corresponding to the smallest singular value. In order to avoid numerical instability it is important to carry out some sort of data normalization. The points in the image can be appropriately normalized by a translation so that their centroid is at the origin, and scaled so that their RMS distance from the origin is $\sqrt{2}$ [48].

2.2.2 Decomposition of the camera matrix

Once the camera matrix M is estimated, it can be decomposed into the form of equation 2.6 to find the camera center, the orientation and the internal parameters of the camera. The camera center $\mathbf{t}$ is the right-null-vector of $M\mathbf{t} = 0$ and can be recovered from the SVD decomposition [48] of M. Furthermore the camera matrix can be decomposed to an upper-triangular and orthogonal matrix to obtain the internal camera matrix K and the rotation matrix R.

2.3 Radial distortion

In the previous section the assumption has been that the linear model based on the collinearity constraint is an accurate model of the imaging process. However, for some lenses this assumption does not hold. The most deviation from the linear model is the radial distortion (see Fig. 2.4(a)). In practice for wide angle cameras with short focal lengths that we will use in our experiments (see section 7) this error becomes quite significant. Therefore, the image measurements need to be corrected to those that would have been obtained under an perfect undistorted linear camera model. Lens distortion takes place during the initial projection of the world onto image plane (see equation 2.1). Suppose (x_u, y_u) denote the image coordinates of a point under non-distorted pinhole projection. The actual point (x_d, y_d) after the radial displacement is given by the following equation:

$$\begin{aligned} x_d &= x_u + L(r)x_u \\ y_d &= y_u + L(r)y_u, \end{aligned}$$

where r is the radial distance $\sqrt{x_u^2 + y_u^2}$ from the center for radial distortion, and $L(r)$ is a distortion function of the radius r. Normally, for the distortion function an infinite series is required. However, Tsai [139] showed in his experiments that only two terms are needed. Any more elaborate modeling would cause numerical instability during the calibration process. The radial distortion is therefore modeled as

$$L(r) = \kappa_1 r^2 + \kappa_2 r^4, \tag{2.10}$$

where κ_1 and κ_2 are the *distortion coefficients*.

In order to correct the distortion in the image the distortion parameters need to be recovered during the camera calibration procedure. Figure 2.4(b) shows the un-distorted image, i.e. the image after the correction of radial distortion.

2.4 Discussion

This chapter described the geometry of a single perspective camera and defined the pose estimation problem. Furthermore, a numerical method has been described for estimating the camera projection matrix from corresponding object and image entities. Throughout this thesis we will assume the camera to be used is already internally calibrated. This only needs to be done once as part of the offline procedure which will be described in the next chapters. For this purpose, we use for our experiments the calibration grid as described in section 2.2. The focal lengths of the cameras is supposed to be fixed and does not change at runtime.

Chapter 3

Related Work

Study the past if you would define the future.
– Confucius (551 BC – 479 BC)

This chapter gives an overview of the developments and the current state-of-the-art in fast 3D object detection and tracking.

3.1 3D Object detection

For a long time the work on object detection could be divided into two streams of research, purely *geometric-based* and purely *appearance-based* approaches. In the geometric-based paradigm, the object of interest is represented by a 3D model of its shape [51, 78, 130]. Detection of the object in a given test image amounts to evaluating if the test image could contain a projection of the model. During this evaluation the system tries to find the object pose which would generate the features, e.g. edges or line segments observed in the test image. These approaches are somewhat robust under changes in illumination since the image brightness is discarded in favor of binary features. However, it is inherently difficult to interpret geometric features of the test image as projections of a 3D model, in particular within scenes with cluttered background, severe partial occlusions and strong viewpoint and scale changes.

In the appearance-based methods, the detection problem is formulated as a statistical classification problem. Existing approaches can be placed on a continuum ranging from purely *global* ones to purely *local* one. Global methods build an object representation by integrating information from several training images. Then, a classifier is learned from this set of training data. Examples for extracted information are color histograms [136] or contours, e.g. 'shape-context' [14] and 'shock-graphs' [128]. The first global approaches were proposed by Turk and Pentland [141] in the face recognition domain and later by Murase and Nayar [96] in a more general context.

Some methods make use of statistical classification techniques such as Principal Component Analysis (PCA) to compare the test image with a set of calibrated training images [107]. The basic idea is to inject a set of training images taken from many different known viewpoints into a PCA, so as to obtain a small number of *eigenimages* capturing most of the appearance variation. Thus, the multi view appearance of the object of interest becomes a low-dimensional manifold in the 'eigenspace' spanned by the eigenimages. Detection is done by projecting the test image into a 'point' in the eigenspace and finding the closest point on the manifold that represents the pose of the closest training image. More sophisticated methods use AdaBoost and classifiers cascade to achieve fast detection performance [39, 149]. These approaches give very good results for recognition of instances of generic object classes such as faces, and can handle tens of instances of the object class in one test image. However, global representation is unfortunately very sensitive to background clutter and partial occlusions, or even poses different from those in the training set. The main reason for the latter one is because the global object appearance varies in a complex and unpredictable way since the object's geometry is not considered.

By contrast to the global approaches, local approaches counter the problems caused by clutter and occlusion by decomposing an image in a collection of relatively small elements, called *local features*. Since these features are extracted based only on local information in the image, partial occlusion does not affect features in the visible part. Therefore, as long as enough features are found and matched the object can still be detected. Moreover, clutter in the test image only results in additional spurious features without compromising the ones on the object. During detection the features are extracted and matched with the ones from the training images. To reject mismatches, the coherence of the spatial arrangements of matched features can be verified in an additional stage. This can be achieved by enforcing geometric constraints such as epipolar constraints [124] or side constraints [72, 36]. Furthermore, a global score can be computed and used as detection criteria which is typically based on the number of verified correspondences.

3.1.1 Local feature detectors

In order to be able to handle as wide as possible a range of viewing conditions, the feature extraction and characterization have to be insensitive to viewpoint, scale and illumination variations. Several kinds of local features have been proposed in the literature. While some approaches use simple features such as corners or edge segments [114], local contour groups [129] or small multicolored image patches [84], more sophisticated approaches rely on local feature neighborhoods or *regions*, which are insensitive to viewpoint and illumination changes [80, 92, 144, 85, 54, 73, 127, 79, 148, 91, 124].

One of the first local feature regions introduced by Schmid et al. [126] was an 'interest

point' (e.g. Harris corners) augmented by a description of its pixel neighborhood, called *descriptor*. Both this interest point detector and descriptor are invariant under rotation. Lindeberg showed [73] that scale-invariant point detection can be achieved by using the local extrema of a Laplacian-of-Gaussian pyramid in scale space as interest points. Lowe [79] approximated the laplcaian by a Difference-of-Gaussian to increase the computational efficiency. Recently, detection under general viewpoint changes was achieved by several authors [92, 144, 85, 54], with affine invariant region detectors. The idea is that even though the global appearance variation of 3D objects is very complex under viewpoint changes, it can be approximated by a simple affine transformation at the local scale because a local feature region can be considered to be approximately planar.

The shape of each local feature is not fixed, but automatically adapts based on the underlying image intensities so as to always cover the same physical surface. Comparison is done using the rectified patches determined by applying the affine transformation that makes equal the two eigenvalues of the second moment matrix. For this type of features Mikolajczyk et al [93] suggest to use the term *covariant* regions rather than invariant, since the regions found in a picture deformed by some transformation are the images of the regions found in the original picture, deformed under the same transformation. However, the normalized image patterns they cover and the respective feature descriptors are typically invariant. Beside the geometric transformations, affine covariant regions do also cope with the photometric deformations between wide baseline images. Using these invariants as a local representation of the shape and appearance of an object provide a bridge between the appearance-based and geometric approaches. Tuytelaars et al [143] presented an edge- and an intensity extrema-based detector. The edge-based detector starts from a corner point and exploits nearby edges, whereas the intensity based one starts from intensity extremum and studies the intensity pattern along rays emanating from this point. Affine covariant features are then constructed by fitting an ellipse to the local structure. Matas et al [85] introduced a fast algorithm to extract Maximally Stable Extremal Regions (MSER). A MSER is a connected component of an appropriately thresholded image. Mikolajczyk et al [93] presented a state of the art on affine covariant region detectors and have compared their performance. Chapter 5 gives an extensive overview on covariant feature regions that have been used in the detection algorithms presented in this thesis.

3.2 Wide baseline matching

Once local feature regions have been extracted correspondences can be established to the features in the database by directly comparing the corresponding feature descriptors. A feature descriptor is a characterization of the feature, which is computed from the intensity patterns within the regions, respectively. The descriptor needs to be not only invariant to viewpoint and illumination changes, but also has to be distinctive to discriminate between the regions. Many different descriptors have been proposed in the literature[127, 143, 94, 80]. Schmid et al [127] compute rotation invariant descriptors as functions of relatively high order image derivatives. Tuytelaars et al [143] use the Generalized Color Moments as a descriptor. David Lowe [80] have introduced the so called SIFT descriptors that tolerates significant local deformations. Among the existing descriptors, a comparison by Mikolajczik et al [89] showed that SIFT descriptors [80] are the most dis-

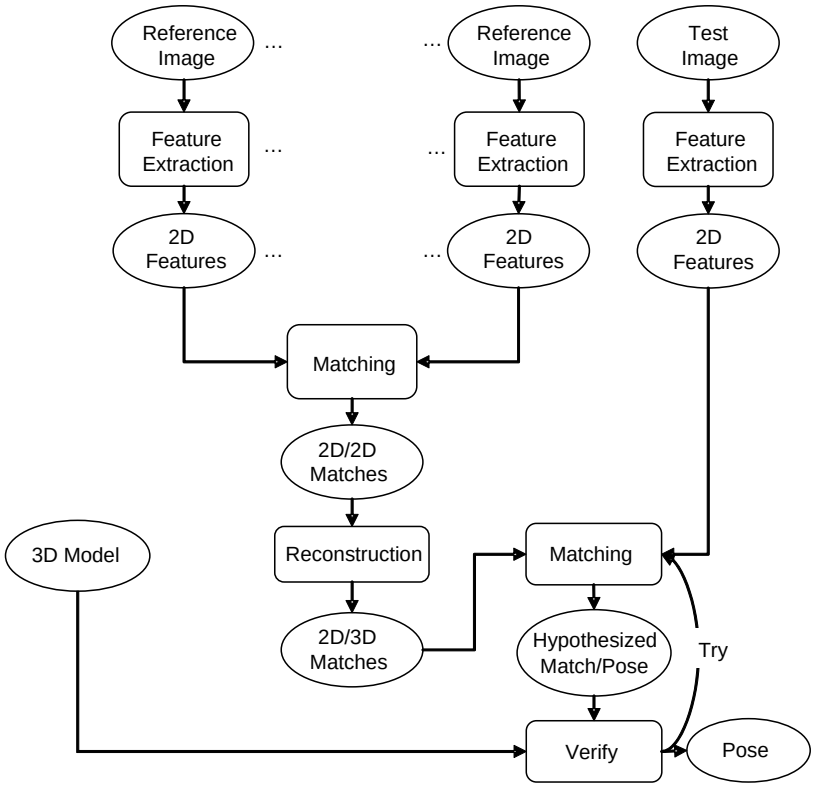


Figure 3.1: Overview of the conventional feature-based approaches for 3D object detection and pose estimation.

tinctive and effective. The distinctiveness is achieved by using a high-dimensional vector representing multiple image gradient histograms in the local neighborhood for each feature point. The reader is referred to section 5.7.2 for implementation details on the SIFT descriptor. This technique has been shown to perform quite good under severe clutter and viewpoint changes [80]. However, the high dimensional SIFT descriptors (128-dimensional) are computationally expensive to match even when using an efficient data structure [13]. Some approaches use dimension reduction techniques, such as PCA, in order to project high-dimensional samples onto a low-dimensional features space. The idea of applying PCA to SIFT feature descriptor was introduced by Ke et al. [57]. They showed that PCA is well-suited to representing key point patches, this representation is more distinctive and therefore improves the matching performance. Furthermore it is more robust and more compact than the 128-dimensional SIFT descriptor.

The use of local descriptors and the matching algorithms vary slightly between different methods for 3D object detection [127, 79, 72, 70, 146, 120, 143, 2, 52, 94]. During an offline training stage, a database is built of interest features lying on the object whose position on the object surface can be computed. For this purpose, usually a set of images are used which are manually calibrated. At runtime, local features are then extracted from the current test image and matched against the database (see Fig. 6.1). The pose is usually estimated from those correspondences using a robust technique such as RANSAC

to eliminate mismatches. A crucial problem that makes this approach difficult for practical use is because of the fact that the images used for building the database and the test images may have been taken from very different viewpoints. Therefore, in this context the wide baseline matching problem becomes a very critical issue that needs to be addressed.

Rothganger et al. [120] introduced a 3D object modeling and recognition algorithm for affine viewing conditions. The objects are represented as a collection of small planar patches, their invariants, and a description of their 3D spatial relationship. The local invariants used in their work are the affine covariant descriptions of the image brightness pattern in the neighborhood of salient image features developed by Lindeberg and Garding [75, 74] and by Mikolajczyk and Schmid [91]. Photometrically and geometrically consistent matches are selected in a two step RANSAC-based pose estimation procedure. Even though this method achieves good results for 3D object detection in difficult viewing conditions, as the authors confirm it is too slow for real-time applications.

In [116] similar results are obtained by training the system using multiple views of the object of interest, and storing all the SIFT features extracted from these views and matching against all of them. However, matching is usually performed by nearest neighbor search, which is computationally expensive as mentioned before, even when using efficient data structures [13]. On the other side, in this framework the descriptors are predefined and do not adapt to the specific images under consideration. In [87] first the set of image neighborhoods of features is built by tracking them over a sequence. Then, a descriptor for each feature is computed by applying Kernel PCA on each set, respectively.

3.2.1 Statistical classification techniques

Lepetit et al. [70] treat wide baseline matching of key points as a classification problem, where each class corresponds to the set of all possible views of each point. If the object of interest can be assumed to be locally planar, the system synthesizes a number of views or image patches by warping image patches around the points under affine transformation.

Once potential matches have been established they apply a plain RANSAC method to recover the 3D pose. Recently they introduced an approach for object pose estimation in real-time [69], where randomized trees [3] are used as the classification technique which is both generic and faster. The basic idea is that since the set of possible patches around an image feature under changing perspective and lighting conditions can be seen as a class, it is possible to train a classifier, made of randomized trees to recognize feature points by feeding it samples of their possible appearances. Each non-terminal node of a tree contains a test of the type: "Is this pixel brighter than this one?" that splits the image space. Each leaf would contain an estimate of the conditional distribution over the classes given that a patch reaches that leaf. Thus, a feature from the test image is classified by simply dropping it down the tree.

This procedure is quite fast since only pixel intensity comparisons need to be done. Because of the efficiency of the randomized trees, this technique yields quite reliable classification results, even with deformable objects [111, 112]. This method assumes that a fixed number of image features have been extracted beforehand using a single or a few reference images and that their number does not change during training, i.e. addition and removal of features is not possible. However, this Randomized Tree-based approach gives very good matching results as long as the set of trained features is small (a few hundred)

and is therefore not very scalable, e.g. for outdoor environments. On the other side key point recognition relies solely on 2D image intensity values within small windows around these key points without considering the underlying 3D structure.

3.3 3D tracking-by-detection

Recently wide baseline feature matching techniques have been used to perform 3D tracking. Özuysal et al [110] propose an approach that relies on the Randomized Tree-based approach described in the previous section. Thereby, an *a priori* 3D model of the object of interest is not required. During an automated training phase, the system learns both geometry and appearance of a set of reliable object feature points, i.e. the system collects or harvests a list of features that can be reliably recognized under large motions and aspect changes which can cause complex appearance variations. In the training process, an ellipsoid is defined that roughly projects at the object's location in the first frame. The object is then required to move slowly in the consecutive frames, where feature points are extracted within this projection and the surrounding image patches are used to train a first classifier. This classifier is used in the following frames to match the initial features. Then, as the set of new features increases, features that cannot be reliably detected are discarded and at the same time new features are added to account for aspect changes. New views of the features are used to refine the classifier and each time features are removed or added, the classifier is updated accordingly. Once all the training images have been processed, the model's geometry is refined using a bundle-adjustment algorithm. Furthermore, the same tracking and statistical classification techniques are used for both training in order to select the most stable features, and for detection and pose computation at run-time. Therefore, the features collected during training are those that can be effectively tracked by the wide-baseline matching algorithm.

A SIFT-based approach was introduced by Skrypnyk and Lowe [133]. In this approach first a database of SIFT features is built from a set of training images during an offline training stage. For this purpose, the 3D locations of the features is computed using a structure-from-motion algorithm. Multi-view correspondences are established using the SIFT descriptors and the 3D coordinates of the features are estimated by a global optimization after over all camera parameters as well as image point coordinates. Thereby, the optimization procedure is initialized as proposed by Szeliski et al [137]. At runtime, SIFT features are extracted from the current test image in the same manner as done during training. The features are matched against the features in the database by comparing their descriptors. This results in a set of 2D/3D correspondences from which camera pose can be estimated using RANSAC in conjunction with the three point algorithm (see Figure 6.1). In the test image the best candidate for a SIFT feature in the database is the nearest neighbor, in the sense of the Euclidean distance of the descriptor vectors. Due to the high dimensionality of the search space, a k-d tree is used to speed up the matching process. Correspondences in the search space are explored in the order of their nearest distance. The search is stopped after a number of candidates has been considered [13]. To decrease the number of mismatches that may result in cluttered scenes the query feature is matched only in case the nearest neighbor is close enough in the image.

Camera pose estimation from each frame independently and from noisy data typically results in jitter [146]. Therefore, the authors use a regularization term to smooth the

camera motion across consecutive frames. To avoid drifting the weight of this term is estimated iteratively. This method runs at four FPS and is therefore not fast enough for real-time tracking as the authors confirm.

3.4 Feature-based tracking systems

A number of approaches have been proposed addressing the problem of tracking based on natural features [43, 146, 68, 29, 119, 147, 100, 132, 23, 108]. One of the earlier markerless model-based real-time 3D visual tracking systems was the RAPiD (Real-time Attitude and Position Determination) system described by Harris [46]. RAPiD demonstrated real-time tracking of models such as fighter aircraft and the terrain near a landing strip. Drummond and Cipolla [31] employed an M-estimator to improve the robustness of a RAPiD-style edge tracking system. Further, RAPiD's view-sphere approach to determining control point visibility is replaced by a real-time hidden edge removal based on graphics acceleration hardware and a BSP-tree1 representation of the model to be tracked. This allows the tracking of more complex structures than possible with RAPiD.

Genc et al. [43] propose a general learning-based framework for feature-based tracking using a single camera. In a two stage process first a set of natural 3D features is learned using an external tracking system (e.g. marker-based). In the second stage the system uses these learned features for tracking as soon as enough stable features are acquired in the first stage. This marker-less tracking system needs an initialization that provides a rough estimate of the camera's position and orientation. It makes use of an external marker-based tracker for initialization. Once the system looses track it needs to be re-initialized in the same way. In this case the initialization process does not need to be very accurate and perform in real-time. The system is able to converge even with partial or imprecise tracking information for initializing system. However, the authors confirm that using markers for initialization is not an acceptable solution in most applications.

Vacchetti et al. [146, 68] propose an automatic initialization method that relies on a learning stage, where a data base of key features is constructed based on a set of key frames taken during an offline procedure. The key features consist of a 3D point on the object model and a viewpoint invariant local descriptor based on its appearance in the images. The initialization is done by robustly matching feature points in the initial image with the points present in the database based on a similarity measure. A disadvantage of this method is that these local descriptors are sensitive to scale and zooming. Therefore, the working space is limited in tracking area that is covered by sufficiently enough key frames. Other efforts have been undertaken for the integration of point features into edge-based trackers [147, 119]. Vacchetti et al [147] combine edge-based tracking with earlier work on tracking Harris feature points. It corresponds to multiple iterations of standard RAPiD with the addition of an M-estimator, using the nearest hypothesis at each iteration. Rosten et al [119] apply the combination of edge and point tracking to an indoor environment. Thereby, a novel FAST algorithm is introduced for matching.

Davison [29] presented a real-time monocular vision-based SLAM (Simultaneous Localization and Mapping) implementation. This system tracks feature points using small 2D image patches and cross-correlation. Apart from four known features initialized at start-up, the 3D positions and appearances of all other features tracked are learned on-line.

Klein et al [60] presented a real-time, full-3D edge tracker based on a particle filter. This system is capable of tracking quite complex self-occluding three-dimensional structures. The system exploits graphics hardware in a novel manner, allowing it not only to perform hidden line removal for each particle but also to evaluate pose likelihoods directly on the graphics card.

Satoh et al. [122] use a bird's-eye view camera additional to the mobile camera that constrains the pose estimation problem. Nevertheless, they don't make use of it in the initialization phase. The initial registration is done each time manually by moving the mobile camera closely to a predefined initial pose.

Other approaches in computer vision literature for pose or motion estimation make use of the ICP principle [153, 15, 151]. The problem with the ICP based algorithms is that they converge to the closest local minimum, and are thus not appropriate for solving large motion problems. Formulating pose estimation as a nonlinear least squares problem, and solving it by nonlinear optimization algorithms is the classical approach used in photogrametry [45]. Typically Gauss-Newton or Levenberg Marquardt methods are used for this purpose [81, 122, 66]. Lowe [81] used the Gauss-Newton method for the pose estimation problem by projecting the model into the image plane and minimizing the registration error in image. A limitation of this method is that it is not robust to occlusions and requires relatively small pose differences. Hager et al. [82] proposed a fast and globally convergent pose estimation method. They formulated the pose estimation problem as that of minimizing the collinearity error in object space rather than in the image space. The algorithm is computationally efficient and accurate but only allows a small number of outliers ($< 20\%$).

3.5 Discussion

This chapter has introduced a number of different approaches to 3D object detection and tracking. Furthermore, we gave a brief review of the feature based tracking systems used in the field of AR. Even though the tracking-by-detection methods described in the previous section are still too slow for real-time applications, the question arises whether the conventional tracking methods will ever be obsolescent sometime. However, we believe that this is unlikely to happen since treating each frame independently has its problems as described in the case of the SIFT-based tracker in the previous section. Using temporal continuity constraints for tracking can increase the robustness and most important the accuracy of the results. Wide-base line matching techniques all tend to be not only less accurate but by far much more computationally expensive. Therefore, our intention is to combine a robust and fast detection system with a real-time tracking system. Such a combined system has both the robustness from a detection system and the high accuracy of a tracking system.

Chapter 4

Sensor Fusion for Pose Estimation

This chapter introduces a sensor fusion framework for automated initialization of markerless tracking systems [101]. It is designed to be used for indoor as well as outdoor environments. The initial positional data can be provided by stationary cameras in closed buildings and for instance by GPS for outdoors while providing no or only rudimentary estimates with respect to the user orientation. The complete initialization is then achieved by fusing the data from multiple sensors, e.g. mobile and stationary cameras or GPS. In cases where tracking is lost for instance because of large occlusions, the initialization procedure is started automatically. This approach can be easily extended to arbitrary additional sensors.

Our intention is to use stationary cameras for non-precise tracking of a user's head and combine the tracking data with those acquired by mobile cameras. Thereby special attention is given to the statistical analysis of the errors in sensors. Specially suitable for this purpose are networked smart cameras. These cameras are equipped with integrated processors and signal processor chips that can immediately process images formed on the sensor chips. The cameras are thus autonomous, i.e. independent from a computer, and provide their tracking results via wireless network to mobile or stationary PCs where the AR applications are running.

This is designed to be integrated in an ubiquitous tracking environment [150], where various tracking sensors with different modalities are used to build dynamically extensible

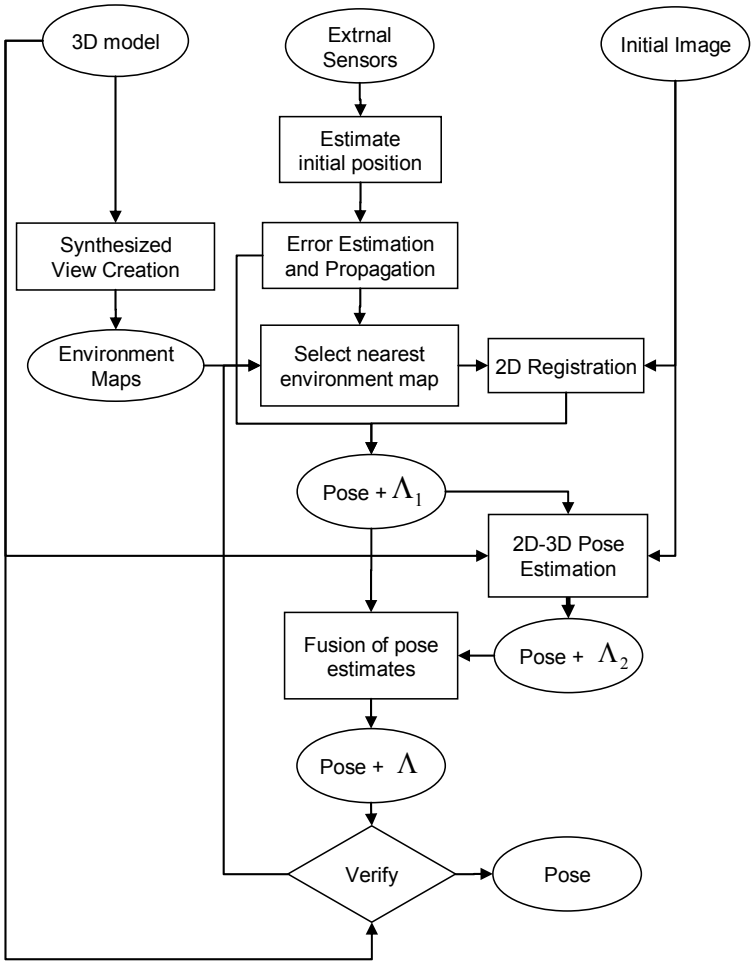


Figure 4.1: Overview of the Algorithm.

networks of trackers with high-precision, low-latency requirements. This approach can be extended to any kind of trackers, since the uncertainties of these individual trackers are taken into account.

Our system requires a rough 3D model of the target object or the scene for automatic initialization and tracking. This is not a problem in practice, since in many applications a 3D model already exists (e.g. in automotive industry) or can be created automatically or interactively by commercially available software (e.g. ImageModeler from RealViz [117], Canoma from Metacreations [88] or Boujou from 2d3 [17]). Furthermore, technological advances in three-dimensional scanning provide accurate devices for automatic model building.

This chapter is organized as follows: Section 4.1 gives a general overview of our method. In section 4.2 we describe how the error estimation and propagation is done which is used for fusion of pose estimations in section 4.3. Sections 4.4 and 4.5 explain the details of the two main steps of the iterative algorithm consisting of a coarse registration followed by a refined registration, respectively. In section 4.6 the experimental results are presented.

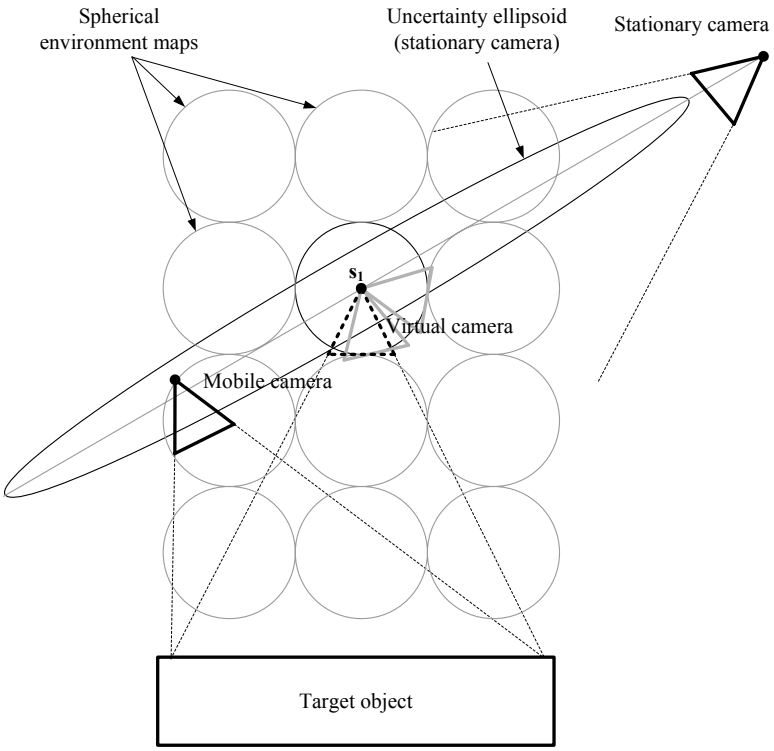


Figure 4.2: The iterative initial registration approach: coarse registration step with a stationary camera (Outside-In).

Discussions and conclusions are provided in section 4.7.

4.1 Overview of the approach

The initialization of the tracking system is performed as follows. First we estimate the position of the user's viewpoint with stationary cameras using a head tracking system and use the image taken by the mobile camera, referred to as the *initial* image, to estimate the initial orientation parameters (see section 4.4). The estimation is then refined by applying the optimization procedure described in section 4.5. During this estimation and refinement process particular attention is given to error propagation and statistical analysis.

To refine the pose data, we introduce a novel method for robust ICP based pose estimation based on definition of two collinearity constraints. They are used as a quality measurement for outliers detection and removal, therefore increasing the efficiency of the algorithm while providing the same accuracy as the classical method introduced by Hager et al. [82].

Using a camera for tracking, the greatest uncertainty of the pose estimate is along the line of sight of the camera and the smallest error is perpendicular to this line (see Fig. 4.4). The reason is that a small translation of the camera parallel to image plane would result in an easily measurable change in the image where a small translation perpendicular to image plane generates only a small displacement and/or change of scale in the image.

In our approach the data from mobile and stationary cameras are combined in order

to estimate the pose of the user's view accurately. Since the pose of the user's view is constrained in space, using the information provided by stationary cameras, the rotational and translational parameters are correlated.

Fig. 4.1 gives an overview of the automated initial registration approach. Fig. 4.2 and 4.3 show a respective graphical overview using one stationary camera as an external sensor. The uncertainty ellipsoid determined by the stationary cameras is a function of the characteristic of the external outside-in tracking system and the position of the user's viewpoint. Using more than one camera or even other tracking devices, only the shape of the uncertainty ellipsoid will change and requires no further handling in our system.

The main steps of our iterative initial registration algorithm are:

1. Estimating the initial position $(t_{x_1}, t_{y_1}, t_{z_1})^t$ of the user's viewpoint using stationary cameras and its uncertainty as described in section 4.2 (Outside-In).

2. Estimating the initial orientation $(\theta_1, \phi_1, \psi_1)^t$ by extracting edges from the initial image and finding the best match of the extracted edges with the edges on the nearest environment map using a similarity measure (see section 4.4) (Inside-Out).

3. Establishing correspondences between 2D edges in the initial image and 3D edges on the 3D model and determining the relative pose $\mathbf{s}_2 = (t_{x_2}, t_{y_2}, t_{z_2}, \theta_2, \phi_2, \psi_2)^t$ and its associated error distribution (see section 4.5).

4. Statistical fusion of the pose estimates $\mathbf{s}_1$ and $\mathbf{s}_2$ from outside-in and inside-out cameras to an overall estimate $\mathbf{s}$ using their uncertainties [1] (see section 4.3).

5. Go back to step 2 unless one of the following termination criteria is reached:

   (a) The displacement between image and model data is smaller than ϵ.

   (b) The change in motion parameters estimated in two consecutive iterations is smaller than $\Delta\epsilon$.

   (c) The maximal number of iterations is reached.

The thresholds ϵ and $\Delta\epsilon$ defined in the termination criteria depend on two requirements: First, the required precision of the initial pose for a successful feature based tracking system which depends on the tracking solution used, and second, the time required for initialization which depends on the application the tracker is used for.

The system provides the uncertainty of the estimated pose in form of covariance matrix. The next section describes how the error estimation and propagation through different coordinate systems is done.

4.2 Error estimation and propagation

We assume that the noise in the image points is independent and its distribution is known by a covariance matrix $\Lambda_{\mathbf{pp}}$. Depending on the edge detection algorithm used the covariance matrix $\Sigma_{\mathbf{pp}}$ could be estimated based on the entries of the Hessian [131]. The uncertainty of the pose is represented by a 6×6 covariance matrix $\Lambda_{\mathbf{ss}}$. It is defined as

[1] Note, that in our case the orientational uncertainties for $\mathbf{s}_1$ are very high compared to its positional uncertainties

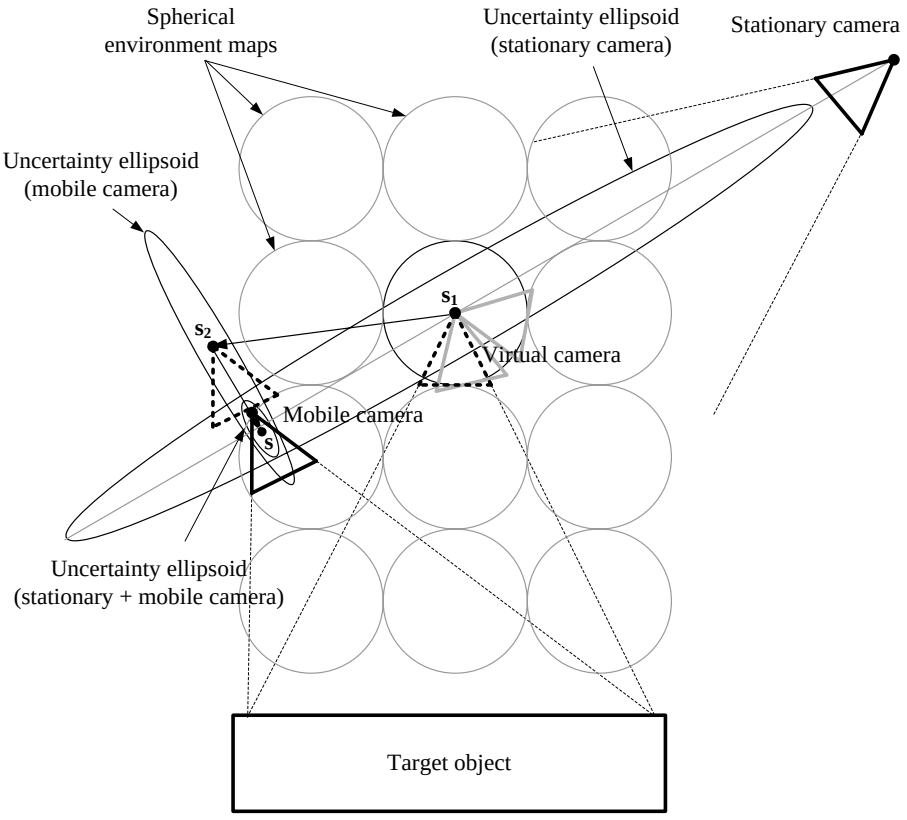


Figure 4.3: The iterative initial registration approach: refined registration step using a mobile camera (Inside-Out) and fusion of pose estimations.

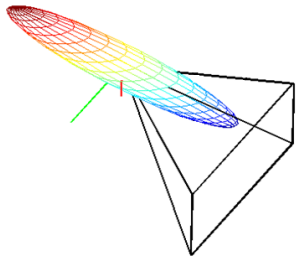


Figure 4.4: Uncertainty of the pose estimate of a camera.

$\Lambda_{\mathbf{ss}} = E(\Delta\mathbf{s}\Delta\mathbf{s}^t)$, the expectation of the square of the difference between the estimated $\tilde{\mathbf{s}}$ and the true values $\mathbf{s} = \tilde{\mathbf{s}} + \Delta\mathbf{s}$.

To compute the covariance matrix $\Lambda_{\mathbf{ss}}$ the nonlinear function f is linearized at the estimated pose $\tilde{\mathbf{s}}$ [49]. The Taylor series expansion gives after neglecting terms of the second and higher order:

$$\mathbf{p}_i + \Delta\mathbf{p}_i = f(\mathbf{P}_i, \tilde{\mathbf{s}} + \Delta\mathbf{s}) \approx f(\mathbf{P}_i, \tilde{\mathbf{s}}) + J_i\Delta\mathbf{s},$$

where $J_i = \left|\frac{\partial f}{\partial \mathbf{s}}\right|^t_{\mathbf{P}_i,\tilde{\mathbf{s}}}$ is the Jacobian of f evaluated at $(\mathbf{P}_i, \tilde{\mathbf{s}})$. Since $\mathbf{p}_i \approx f(\mathbf{P}_i, \tilde{\mathbf{s}})$, we get

$$\Delta\mathbf{p}_i = J_i\Delta\mathbf{x}.$$

Stacking all the equations for n points yields $\Delta P = J\Delta\mathbf{s}$. This equation can now be solved for $\Delta\mathbf{s}$ in a least squares manner as $\Delta\mathbf{s} = (J^tJ)^{-1}J^t\Delta P$. The covariance matrix is calculated by substituting $\Delta\mathbf{s}$:

$$\Lambda_{\mathbf{ss}} = (J^tJ)^{-1}J^t\Sigma_{\mathbf{PP}}((J^tJ)^{-1}J^t)^t. \tag{4.1}$$

Since we assume that the errors in image points are not correlated we have $E(\Delta\mathbf{p}_i\Delta\mathbf{p}_j^t) = 0$, for $i \neq j$. Therefore $\Sigma_{\mathbf{PP}}$ is a diagonal matrix containing $\Sigma_{\mathbf{pp}}$ as diagonal elements. Using equation (4.1) the uncertainty of the pose can be estimated as a covariance matrix in the respective camera coordinate system.

Having different coordinate systems (see Fig. 4.5), we need to transform the covariance matrix properly to the same unit coordinate system. Propagating uncertainty in general through different functions is described by the *error propagation law* [64]. Given the pose $\mathbf{x}$ and its covariance matrix $\Lambda_{\mathbf{xx}}$, let $\mathbf{y} = g(\mathbf{x})$ be the function which transforms $\mathbf{x}$ to $\mathbf{y}$ in another coordinate system. The covariance matrix of $\mathbf{y}$ can then be calculated by

$$\Lambda_{\mathbf{yy}} = J\Lambda_{\mathbf{xx}}J^t, \quad \text{with} \quad J = \frac{\partial g}{\partial \mathbf{x}}. \tag{4.2}$$

A useful representation of covariance matrices in 3D are the error ellipsoids, assuming that the errors are jointly Gaussian. The joint probability density function (pdf) for N-dimensional error vector $\mathbf{x}$ is

$$p(\Delta\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^N\,|\Lambda_{\mathbf{xx}}|}} e^{-\frac{1}{2}\Delta\mathbf{x}^t\Lambda_{\mathbf{xx}}^{-1}\Delta\mathbf{x}}.$$

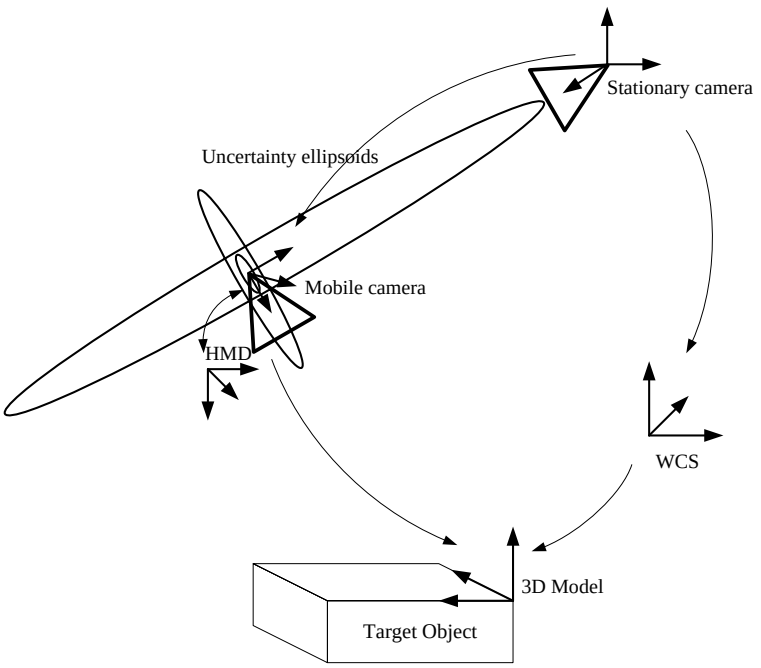


Figure 4.5: The coordinate systems.

If the argument of the exponent is constant, the surface of constant probability is an ellipsoid specified by the equation $\Delta\mathbf{x}^t\Lambda_{\mathbf{xx}}^{-1}\Delta\mathbf{x} = c^2$, for a constant c. For $c = 3$ the cumulative probability of the error vector $\mathbf{x}$ being inside the ellipsoid is approximately 97% [49].

4.3 Fusion of pose estimations

A Kalman Filter-based fusion of pose estimates from multiple tracking systems is achieved in the following manner (see [55] for the basic filtering technique). Let $\mathbf{s_1}$ and $\mathbf{s_2}$ be the two pose vectors and $\Lambda_{\mathbf{s_1s_1}}$, $\Lambda_{\mathbf{s_2s_2}}$ the respective covariance matrices. The combined estimate $\mathbf{s}$ is obtained by weighting and averaging the covariance matrices [35, 49].

$$\mathbf{s} = (\Lambda_{\mathbf{s_1s_1}}^{-1} + \Lambda_{\mathbf{s_2s_2}}^{-1})^{-1}(\Lambda_{\mathbf{s_1s_1}}^{-1}\mathbf{s_1} + \Lambda_{\mathbf{s_2s_2}}^{-1}\mathbf{s_2}) \tag{4.3}$$

Reforming this equation yields for $\mathbf{s}$ and its covariance matrix

$$\begin{aligned} \mathbf{s} &= \Lambda_{\mathbf{s_2s_2}}(\Lambda_{\mathbf{s_1s_1}} + \Lambda_{\mathbf{s_2s_2}})^{-1}\mathbf{s_1} + \\ &\quad \Lambda_{\mathbf{s_1s_1}}(\Lambda_{\mathbf{s_1s_1}} + \Lambda_{\mathbf{s_2s_2}})^{-1}\mathbf{s_2}, \\ \Lambda_{\mathbf{ss}} &= \Lambda_{\mathbf{s_2s_2}}(\Lambda_{\mathbf{s_1s_1}} + \Lambda_{\mathbf{s_2s_2}})^{-1}\Lambda_{\mathbf{s_1s_1}}. \end{aligned} \tag{4.4}$$

Figure 4.6 shows three examples for two error ellipsoids and the resulting combined ones.

4.4 Coarse registration using environment maps

In computer vision Adelson and Bergen [1, 86] assigned the name *plenoptic function* (from plenus, complete or full, and optic) to the pencil of rays visible from any point in space, at any time, and over any range of wavelengths. They use this function to develop a taxonomy for evaluating models of low-level vision. The plenoptic function is a parameterized function for describing everything that can be seen from the point of view of the user. From a given point of view we can select any of the viewable rays by choosing an azimuth and elevation angle (θ, ϕ). In computer graphics terminology, the plenoptic function describes the set of all possible environment maps for a given scene.

We first define a complete sample of the plenoptic function as a full spherical map for a given viewpoint, defined thanks to the external outside-in tracking cameras. Having a virtual model of the environment or the target objects, the viewing space can be coarsely sampled and a set of spherical environment maps $\mathbf{M} = \{\mathbf{M}_i | 0 < i \leq N\}$ is generated[2] (see Fig. 4.7). For the first stage of the initialization procedure the orientation of user's head is estimated as following. Given a set of samples $\mathbf{M}$ from the plenoptic function in form of spherical environment maps, the system selects the closest environment map $\mathbf{M}_k \in \mathbf{M}$. The initial view is then projected onto the spherical map $\mathbf{M}_k$, and the best match represented by three rotational parameters is estimated using a similarity measure.

Several methods have been proposed in the literature to align two 2D images [155]. We use a intensity based similarity measure known as the gradient correlation. For this purpose four gradient images are created by horizontal and vertical Sobel templates from the respective environment map and the initial image. The normalized cross correlation (NCC) is calculated of both horizontal and vertical gradient images, respectively. The similarity measure value is the average of the two NCCs. We use an efficient implementation for fast computation of NCC [71, 45].

Since the gradient images are used for registration we only need to save the gradient images of the environments maps. To reduce the amount of data storage needed for storing the gradient images Laplacian pyramid can be used. In order to speed up the searching for the highest correlation value, the same image pyramids can be used. This results in a hierarchical iterative registration of the initial image and the respective spherical environment map.

Due to the uncertainty in estimating user's viewpoint from stationary cameras and the limited sampling rate of the plenoptic function only a coarse pose estimation can be achieved with the method described above (see Fig. 4.2). We therefor use this estimated pose as the initial values for a second stage of the initialization, in which a 3D-2D pose estimation method is proposed to accurately estimate the relative displacement of the pose.

4.5 Refined 2D-3D registration

The coarse pose estimation brings the contour edges extracted in the initial frame close to the corresponding edges in the respective environment map. The extracted contour edges are represented as a set of discrete points. Using the 3D model of the scene, we can

[2] Due to the complexity of the scene and the sample rate this can be a quite time consuming procedure. This does not affect the computational cost of the system since this can be done offline using a rendering system.

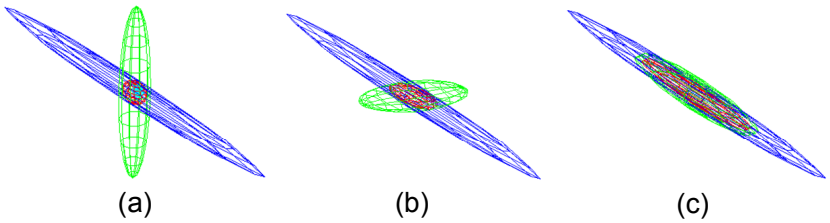


Figure 4.6: Two error ellipsoids (blue and green) and the resulting fusion of them (red).

retrieve the 3D position of the edge points on the virtual model. This section describes how to determine the relative pose that coincides the position of those 3D edge points on the model onto the 2D edge points in the initial image plane, i.e. registration of 3D and 2D points. For more robustness small edges are removed by thresholding and only dominant edges are used.

Since we have no a-priori knowledge of correspondences we use a 3D-2D registration algorithm based on the iterative closest point (ICP) principle [153, 15]. It is composed of three iterated steps, the first of which determines correspondence candidates between 2D and 3D edge points. In the second step a robust technique is used to discard the outliers by analyzing the statistics of the distances. And finally the third step estimates the 3D rigid transformation that minimizes the displacement of matched points.

The next three sections describe each step of the algorithm.

4.5.1 Establishing correspondence candidates

Since there is no distance metric relating the 2D edge points in the initial frame to 3D edge points on the model, there is no obvious way to directly applying the ICP principle to the registration of 3D model edge points to 2D image edge points.

Let $\{\mathbf{p}'_j | 0 < j \leq m\}$ denote the set of extracted 2D image edge points. The correspondence candidates are chosen in a way that both the 2D error distance between the back projected model edge point and observed image points $\mathbf{p}'_j$ as well as the 3D distance of the model points to the projection rays of $\mathbf{p}'_j$ in object space is minimized (see Fig. 2.1).

Given the estimated pose parameters $\tilde{R}$ and $\tilde{\mathbf{t}}$, the distance between a 3D model point $\tilde{R}\mathbf{P}_i + \tilde{\mathbf{t}}$ to the projection ray of the 2D edge point $\mathbf{p}'_j$ is due to object space collinearity equation (2.2)

$$d_{obj}(\mathbf{P}_i, \mathbf{p}'_j) = \|\mathbf{Q}_i - F_j \mathbf{Q}_i\| = \left\|(I - F'_j)(\tilde{R}\mathbf{P}_i + \tilde{\mathbf{t}})\right\|, \tag{4.5}$$

where F'_j is the projection operator (see section 2.1.1) defined for the image points $\mathbf{p}'_j$

$$F'_j = \frac{\mathbf{p}'_j \mathbf{p}'^t_j}{\mathbf{p}'^t_j \mathbf{p}'_j}.$$

Analogue due to the image space collinearity equation (2.1) the distance between the 2D edge point and the back projected 3D model point in the image plane is

$$d_{img}(\mathbf{P}_i, \mathbf{p}'_j) = \left\|\mathbf{p}'_j - \frac{\tilde{R}\mathbf{P}_i + \tilde{\mathbf{t}}}{\tilde{\mathbf{r}}_3 \mathbf{P}_i + \tilde{t_z}}\right\|. \tag{4.6}$$

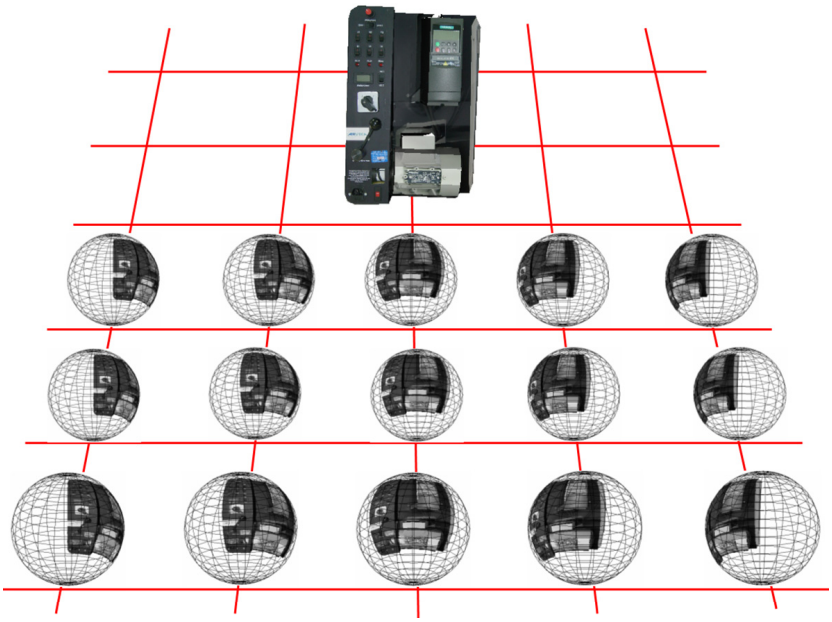


Figure 4.7: A subset of the environment maps surrounding the object of interest.

The smaller d_{obj} and d_{img}, the more likely $(\mathbf{P}_i, \mathbf{p}'_j)$ represent a correct correspondence. To establish correspondence candidates for every image point we take the model edge points with the smallest distance d defined as the sum of the weighted distances in image and object space.

$$d(\mathbf{P}_i, \mathbf{p}'_j) = \alpha_{img} d_{img}(\mathbf{P}_i, \mathbf{p}'_j) + \alpha_{obj} d_{obj}(\mathbf{P}_i, \mathbf{p}'_j).$$

Since the distances in image and object space are of different order they are weighted properly using the factors α_{img} and α_{obj}. For $\alpha_{img} = 1$ and $\alpha_{obj} = 0$ only the nearest points in the image are considered as correspondence as used in [66] whereas for $\alpha_{img} = 0$ and $\alpha_{obj} = 1$ only the nearest point in the object space are chosen as candidates [151]. We use both distances to select the best matches for this purpose. I.e. for any 2D edge point $\mathbf{p}'_j$ $(0 < j \leq m)$, its correspondence candidate $\mathbf{P}_{c_j}$ is determined as

$$\mathbf{P}_{c_j} = argmin_{c_j \in \{1,\ldots,n\}} \, d(\mathbf{P}_{c_j}, \mathbf{p}'_j), \tag{4.7}$$

where c_j are the corresponding indices of the 3D model points. The search space is determined by the size of the model edge point set n. In order to speed up the search, optimized K-D tree data structure can be used to accelerate the closest point search [115, 153].

4.5.2 Estimating motion

This section describes briefly how to re-estimate the pose parameters that minimize the displacement between the corresponding edge points.

According to the ICP principle we minimize the object space collinearity error (2.2) by moving the model data $\mathbf{P}_{c_j}$ such that at each step the displacement between $\mathbf{p}_j$ and $\mathbf{P}_{c_j}$ is minimized.

The basic idea is to reduce at each iteration the 3D-2D registration problem to the 3D-3D registration of points by using the 3D projection points on the respective image rays instead of 2D image points [82, 151]. For each iteration the projection points have to be determined again due the new pose estimates.

Formally, we seek the rigid pose parameters R and $\mathbf{t}$ that minimizes the following mean-squares objective function

$$E(R,\mathbf{t}) = \sum_{j=1}^{m} w_j \left\| (I - F'_j)(R\mathbf{P}_{c_j} + \mathbf{t}) \right\|^2, \tag{4.8}$$

subject to the orthogonality constraint $RR^t = I$. The w_j are positive weighting factors associated with each correspondence candidate. See next section for how to choose these weights.

For a fixed rotation R the optimal translation t can be computed from (4.8) as

$$\mathbf{t}(R) = (I - \frac{1}{m}\sum_{j=1}^{m} w_j F'_j)^{-1} \sum_{j=1}^{m} w_j (F'_j - I) R\mathbf{P}_{c_j} \tag{4.9}$$

The estimated rotation matrix $\tilde{R}$ can be used as the starting point and is re-estimated iteratively as follows. Let R^k be the kth estimate of R, $\mathbf{t}^{(k)} = \mathbf{t}(R^{(k)})$, and $\mathbf{Q}_{c_j}^{(k)} = R^{(k)}\mathbf{P}_{c_j} + \mathbf{t}^{(k)}$. The next estimate $R^{(k+1)}$ is determined by

$$R^{k+1} = argmin_R \sum_{j=1}^{m} w_j \left\| R\mathbf{P}_{c_j} + t^{(k)} - F'_j \mathbf{Q}_{c_j}^{(k)} \right\|^2, \tag{4.10}$$

subject to $R^t R = I$. Such a constrained least squares problem can be solved for $R^{(k+1)}$ in closed form using quaternions [50] or singular value decomposition (SVD) [44]. For the SVD solution, first a sample cross-covariance matrix M between $\mathbf{P}_{c_j}$ and $\mathbf{L}_{c_j}^{(k)} = F'_j \mathbf{Q}_{c_j}^{(k)}$ is calculated.

$$M = \sum_{j=1}^{m} w_j (\mathbf{P}_{c_j} - \bar{\mathbf{P}})(\mathbf{L}_{c_j}^{(k)} - \bar{\mathbf{L}}^{(k)}), \tag{4.11}$$

where $\bar{\mathbf{P}}$ and $\bar{\mathbf{L}}^{(k)}$ are the centroids, respectively. Let UDV^t be a SVD of M, where U and V are orthogonal matrices, and D is diagonal. Then the optimal solution to (4.10) is

$$R^{(k+1)} = VU^t. \tag{4.12}$$

The next estimate of translation is then computed by $\mathbf{t}^{(k+1)} = \mathbf{t}(R^{(k+1)})$ from (4.9). Note, that the computational complexity at each iteration k is linear in the number of points considered. This iterative algorithm directly computes orthogonal rotation matrices, it is fast and convergent due to the convergence theorem of the original ICP algorithm [15].

4.5.3 Robust pose estimation

Due to both occlusion and inaccuracy in pose estimation, correspondence candidates $(\mathbf{p}'_j, \mathbf{P}_{c_j})$ established in the section 4.5.1 are not guaranteed to be correct. There are

two types of outliers (wrong matches). The first is where $\mathbf{P}_{c_j}$ is not the corresponding 3D model point, while the correct correspondence exists. The second is a match where the corresponding point is not included in the set of 3D model points.

A large number of algorithms described above have been developed to quantitatively evaluate the 2D-2D or 3D-3D point matches [76, 77]. In this section we describe a novel approach to evaluate the 3D-2D point matches based on a statistical model.

The equation (2.2) and (2.1) essentially describe an object space and an image space collinearity constraint. The former means that the image point $\mathbf{p}'_j$, the projection of $\tilde{R}\mathbf{P}_{c_j} + \tilde{\mathbf{t}}$ on $\mathbf{p}'_j$ and the optical center $\mathbf{O}$ of the camera are as collinear as possible. The latter constraint means that the the image point $\mathbf{p}'_j$, model point $\mathbf{P}_{c_j}$ and projection center are collinear. These constraints represent necessary conditions for a pair of 2D-3D edge points to be correct. If a candidate does not satisfy any of these constraints, it can not be a correct one. Thus, we use these constraints as a quality measurement of correspondence candidates from which relatively good matches can be selected and used for pose re-estimation.

Based on the point matches the means $\mu_{d_{obj}}$, $\mu_{d_{img}}$ and standard deviations $\sigma_{d_{obj}}$, $\sigma_{d_{img}}$ are computed. Depending on these values we reject outliers:

$$\begin{aligned} \text{if } (\Big|d_{obj}(\mathbf{p}'_j, \mathbf{P}_{c_j}) - \mu_{d_{obj}}\Big| > \kappa_{obj}\sigma_{d_{obj}} \text{ or} \\ \Big|d_{img}(\mathbf{p}'_j, \mathbf{P}_{c_j}) - \mu_{d_{img}}\Big| > \kappa_{img}\sigma_{d_{img}}) \\ \text{then } w_j = 0. \end{aligned} \tag{4.13}$$

where w_j is a weighting factor introduced in equation (4.8). The maximum tolerance parameters κ_{obj} and κ_{img} represent how many percent of matches are considered as outliers and rejected. For a parameter value of 1.0 approximately 68% of matches lying in the interval $[\mu - \sigma, \mu + \sigma]$ are taken into account and the rest is rejected as outliers. For a value of 3.0 nearly all the matches are used for motion estimation. The maximum tolerable distances in image and object space can be determined based on the maximal motion expected. They can be chosen properly based on the scene depth and sampling rate of the plenoptic function (see section 4.4). This values have an impact on the convergence of the algorithm. If they are too small, more iterations are required for the algorithm because many good correspondence candidates will be discarded. As a result of this procedure, a set of refined correspondences will be obtained.

Another problem that arises with the solution described in the previous section is that if the image points are perturbed by homogeneous Gaussian noise, the pose solution will implicitly more heavily weight model points that are farther away from the camera, since d_{obj} increases with distance of the model point to the camera. Supposing that the residual error, i.e. the distance d_{obj} is approximately proportional to the depth and equal for all points, the weights w_j can be chosen as

$$w_j = \frac{1}{(Z_{c_j}^{(k)})^2}, \tag{4.14}$$

where $Z_{c_j}^{(k)}$ is the depth of each model point $Q_{c_j}^{(k)}$ in the camera coordinate system [82].

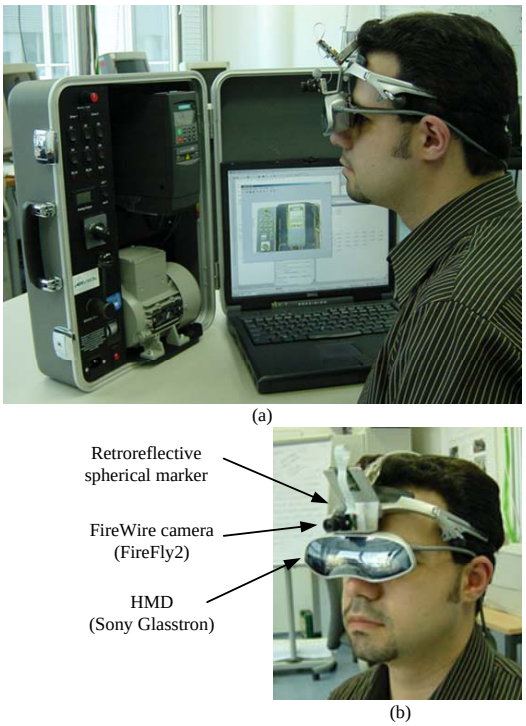


Figure 4.8: The experimental setup.

4.6 Results

The registration algorithm has been tested in a real scene and the registration accuracy was analyzed. This section presents the experimental results.

The registration algorithm has been implemented in Matlab on a 2GHz Pentium 4 CPU, with 1.0 GB RAM. The code is not optimized yet because we are in testing and evaluation phase. The algorithms are however designed with particular attention to real-time requirement. In our experimental setup (see Figure 4.8(a)) a FireWire (IEEE-1394) digital camera (FireFly2 from Point Grey) was mounted on the optical see-through HMD (Sony Glasstron). We used a 4 mm wide angle lens with a field-of-view of 68 degrees for the experiments. The camera was internally calibrated and lens distortion was corrected using the camera calibration toolbox [16].

For evaluation purposes we use the A.R.T. [6] outside-in infrared tracking system in our AR lab to track the position of the user's viewpoint. Thereby, a small retro reflective spherical marker is attached to the mobile camera (see Fig. 4.8(b)), which was then tracked by three ART cameras hanging in the corners of our lab. The tracking data were sent via wireless LAN to the mobile computer with the application running. In the future this marker-based system can be replaced by a marker-less head tracking system using stationary smart cameras (see Fig. 4.9).

As an AR scenario, we used a control unit box provided by Siemens Automation & Drive as the target object. We created an accurate 3D model of the box using the software ImageModeler from RealViz [117] (see Fig. 4.10(a)). ImageModeler uses photographic

Figure 4.9: Real time face tracking system based on the Continuously Adaptive Mean Shift (CAMSHIFT) algorithm [18]. The two perpendicular white lines are the two main axis of the error ellipse in the image.

(a) Control Box

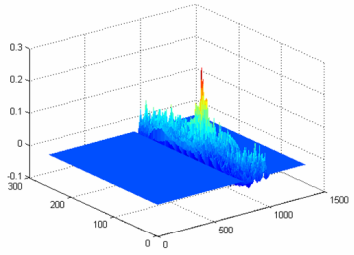
(b) 3D model

(c) Cylinderical environment map

(d) Best match of the initial image on the environment map

(e) Initial image

(f) Virtual camera view



(g) Gradient correlation

Figure 4.10: The coarse registration.

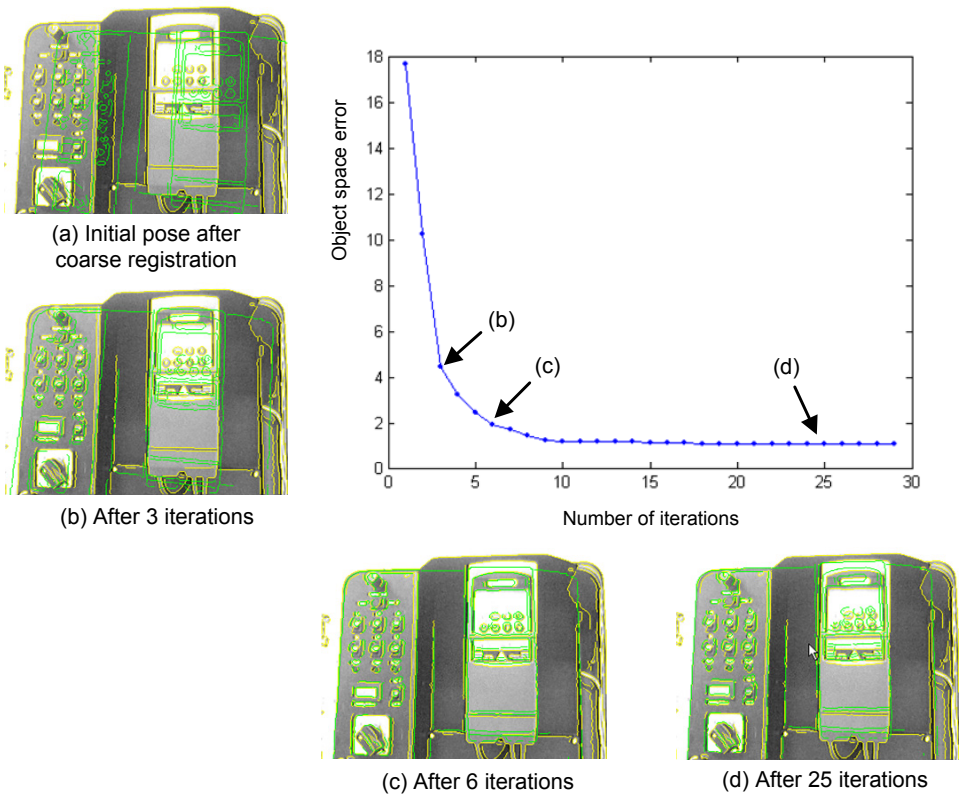


Figure 4.11: The refined 2D-3D registration. Aligned model edges with the edges of the initial image. After 25 iterations the registration accuracy is largely above the requirements of the current feature-based tracking algorithms.

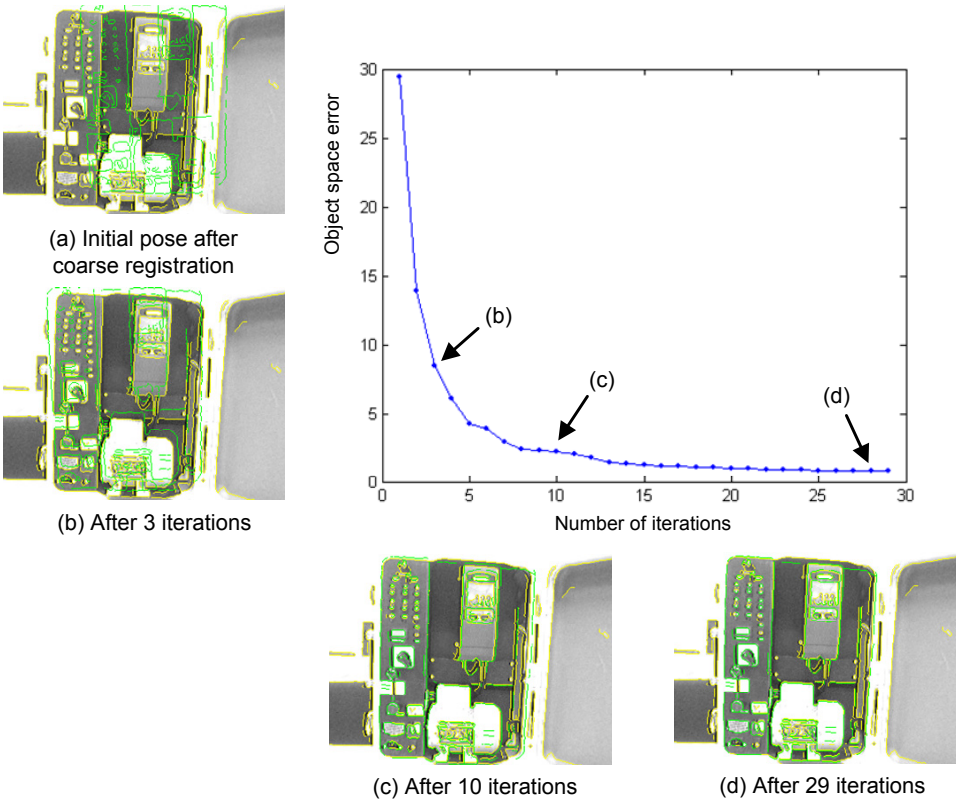


Figure 4.12: The refined 2D-3D registration. Aligned model edges with the edges of the initial image. After less than 30 iterations the results are accurate enough for the existing feature-based tracking algorithms.

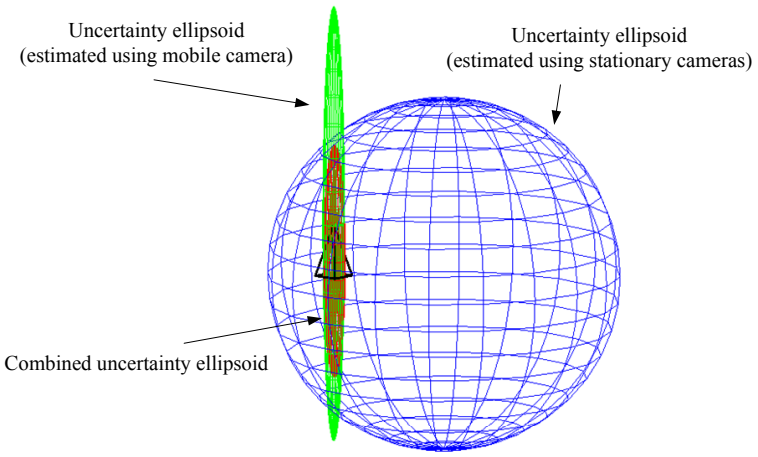


Figure 4.13: Positional uncertainty and sensor fusion using the A.R.T. tracking system. The diameter of the spherical error ellipsoid is 4cm.

images taken from the object to recover the 3D geometry and maps automatically the original images onto the model's surface as texture maps, resulting in a highly realistic model.

The pose of the box in the room was determined by placing a ART-target at a fix position on the box. The transformation between the ART marker coordinate frame and the model frame was calculated by measuring the 3D coordinates of some points on the box in model and marker coordinate frame respectively. From those correspondences the transformation parameters were calculated. This step needs to be done only once when the box is moved relative to the ART tracking cameras.

For projecting a complete plenoptic sample the most natural surface would be a unit sphere centered about the viewing position (see section 4.4). However, the difficulty of spherical projections is the lack of a representation that is suitable for data storage, particularly for a uniform discrete sampling [86]. We have therefore chosen to use a cylindrical projection as the plenoptic sample representation. The advantage of a cylinder is that it can be easily unrolled into a simple planar map.

Figure 4.10(b) shows a cylindrical environment map of the box as a sequence of images taken from panning the virtual camera 360 degrees around the optical axis. Thereby the internal parameters of the virtual camera are identical to the real camera used.

The viewing space in front of the virtual box was sampled automatically every 5 cm in each space direction, resulting in a total set of 1080 environment maps, where only the respective gradient images were stored. Figure 4.10(d) shows the initial image of size 320×240 taken by the mobile camera after the lens distortion correction.

Using the A.R.T. outside-in tracker the position of the marker attached to the camera was determined and the nearest environment map was selected automatically. At a range of 1.5 m, the stated RMS (root mean square) accuracy by the manufacturer of locating a single ART marker is 0.5 mm. Since the marker is about 3.5 cm away from the camera, we assume the maximal positional error is about 4 cm.

In our experiments with three tracking cameras, the positional uncertainty could be modeled with an error ellipsoid of the form of a sphere (see Fig. 4.13) since the variations

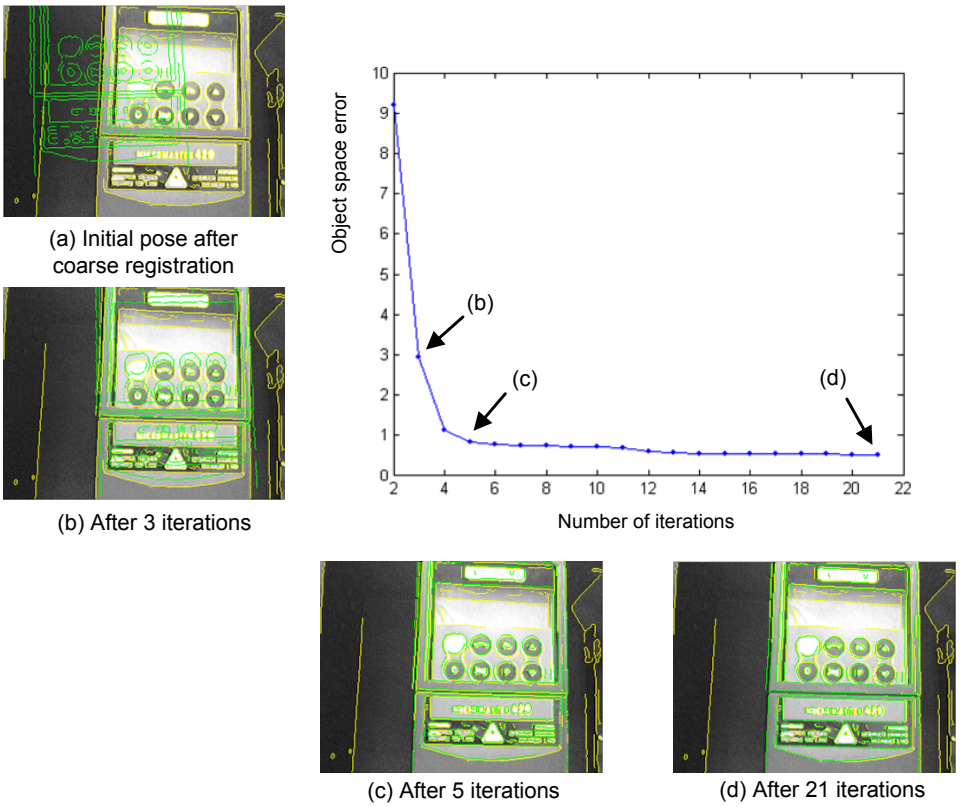


Figure 4.14: The refined 2D-3D registration. Aligned model edges with the edges of the initial image. After 21 iterations the registration accuracy is largely above the requirements of the current feature-based tracking algorithms.

are not statistically significant in this study.

A coarse estimation of the orientation was then done by finding the best match of the initial image in the respective environment map (see Figure 4.10(d)). From the position of the best match the azimuth and elevation angle were derived. In our experiments elevation and the rotation angle (around the optical axis of the camera) are considered to be small. However, up to a maximum variation of ± 30 degrees the correct match could be found properly using the NCC measurement (see section 4.4).

The view of the virtual camera with the coarse orientation is shown in Fig. 4.10(f). The final displacement is then estimated using the method described in section 4.5. For this purpose the motion between the 2D edges in the initial image and 3D edges on the model is estimated.

Figure 4.11 shows the procedure of the iterative registration of the 3D edges with 2D edges and a plot of the the object space error d_{obj} defined in (4.5), during the pose estimation process. We observe a fast convergence of the algorithm during the first iterations that slows down as it approaches its minimum. After less than 25 iterations the initial registration result is accurate enough for the existing feature-based tracking algorithms.

The uncertainty of the final pose derived from the mobile camera was then estimated as a covariance matrix as described in section 4.2. The maximum translational error along

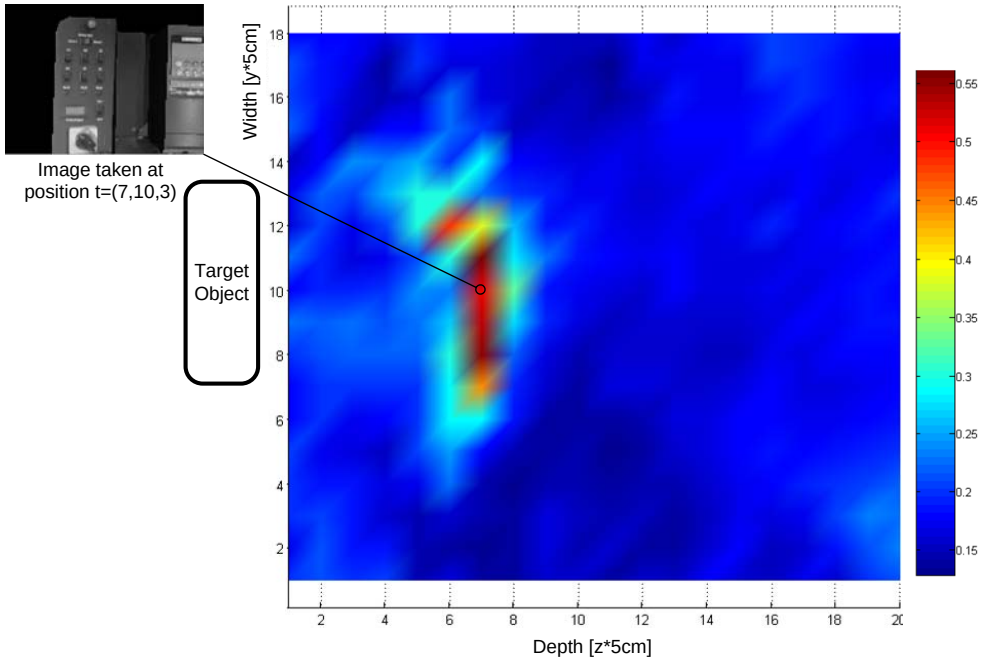


Figure 4.15: Global similarity map. Graphical visualization of the similarity values (NCC) of a test image (taken at position (7,10,3) from the virtual box) with the set of surrounding environment maps.

the line of sight is about 21 mm. The error ellipsoid of the overall estimation after fusion has a major axis of 1.6 mm. Figures 4.12 and 4.14 shows the registration process of two different initial views of the target object. This figure shows two real examples, in both cases in less than 30 iterations initialization has been successful with an accuracy which is largely above the requirements of the current feature-based tracking algorithms.

Figure 4.15 shows a visualization of the similarity values (NCC) of a test image taken in front of the virtual box, with the surrounding environment maps. As expected the location of the image is within the the area with highest similarity values. This map is called the *global similarity map*. For a more robust solution capable of handling partial occlusions and clutter background we will investigate a feature based detection framework in chapter 6. Within that framework for each local feature a *local* similarity map is created and used for classification.

4.7 Discussion

This chapter presented a sensor fusion approach for automated initialization of marker-less tracking systems. This was achieved by analyzing and estimating the error of tracking sensors. The uncertainty of the tracking sensors is represented by covariance matrices and can be visualized as 3D ellipsoids. The initial pose was then estimated iteratively with a coarse-to-fine strategy by taking the uncertainties into account. We applied the method to an augmented reality system using mobile and stationary cameras.

The pose parameters are estimated in two estimation and refinement steps. Thereby the second step is independent of the first one and can require several iteration steps to

converge. This can be improved in a future work by considering the positional uncertainty derived from the stationary cameras during the refined pose estimation step.

The pose refinement is based on minimizing the object space collinearity errors. A more efficient and faster solution could be the minimization of both object and image space collinearity errors simultaneously for pose estimation.

One of the contributions of this approach is its adaption, appropriate modification and integration of advanced methods such as plenoptic viewing, NCC and ICP. The initialization method is therefor fully automated. Experiments on synthetic and real data have proven successful.

Chapter 5

Image Correspondence and Covariant Features

Any sufficiently advanced bug is indistinguishable from a feature.

– Rich Kulawiec

The subject of this chapter is the correspondence problem and the relevant recent techniques dedicated to it. This chapter gives a snapshot of the state of the art in image feature detectors as well as robust and computational efficient matching techniques. These methods have been implemented and optimized with particular attention to real-time performance for the proposed object detection system in the next chapter. Many of the technical terms used later will be introduced and defined in this chapter.

5.1 The correspondence problem

Given two images of the same scene or object taken from two different viewpoints as in figures 5.1 and 5.2, a natural question is: *which features do correspond between the images?* Here the word 'correspond' means 'are on the same scene spot or the same part on the object'[1]. Figures 5.1 and 5.2 show some corresponding points as 'features'. Depending on the viewpoint and the structure of the object, not every feature needs to have

[1]Since the problems for scene or object are equivalent we will only use 'object'.

Figure 5.1: Two images from different views of the same scene (Taj Mahal Hotel, Mumbai (Bombay), India). Feature points with the same number correspond.

a corresponding feature in the other view. They might be out of the other field of view, or occluded by other features. Furthermore, the appearance of the visible features might be different between the two views. Sources of appearance variations beside viewpoint changes, could be zoom, i.e. changes in focal length, illumination variations and noise due to the imaging process. Beside points, features could be edges, line segments, small patches or *regions*. This problem of finding correspondences between two views is called the *two-view correspondence problem* [38].

By relaxing the assumption that both images show the same object, we can have the question: *do the images show the same object?* This formulates the basic *object detection problem*. The correspondence and detection problem are closely related. Finding correspondences means detecting features of the object, and detecting many features of the object might imply the detection of the whole object.

The object detection problem is one of the fundamental problems in computer vision. Other common problems in computer vision related to the correspondence problem are object tracking, stereo vision, 3D reconstruction, image/video retrieval, mosaicking, robot navigation, etc. In this thesis we focus on the problem of object detection and propose two novel methods in chapters 4 and 6.

5.2 Correlation techniques

In the computer vision literature many techniques have been proposed for finding two-view correspondences. The first classic methods were based on a two step process. First, interest points were extracted automatically in each image and in the next step only correspondences of those interest points were searched for between the views. The interest points were usually corner-like structures, e.g. Harris corner detector [47] was used for this purpose. Other existing corner detectors have been proposed by Förstner [40], Smith et al [134] ($SUSAN$ detector) or recently by Rosten et al [119] ($FAST$ detector). Each corner was then characterized by a small window of pixels around it. The search for correspondences was performed by computing the correlation between the grayscale patterns of pair of windows with fixed size. Pairs of interest points with high correlation between

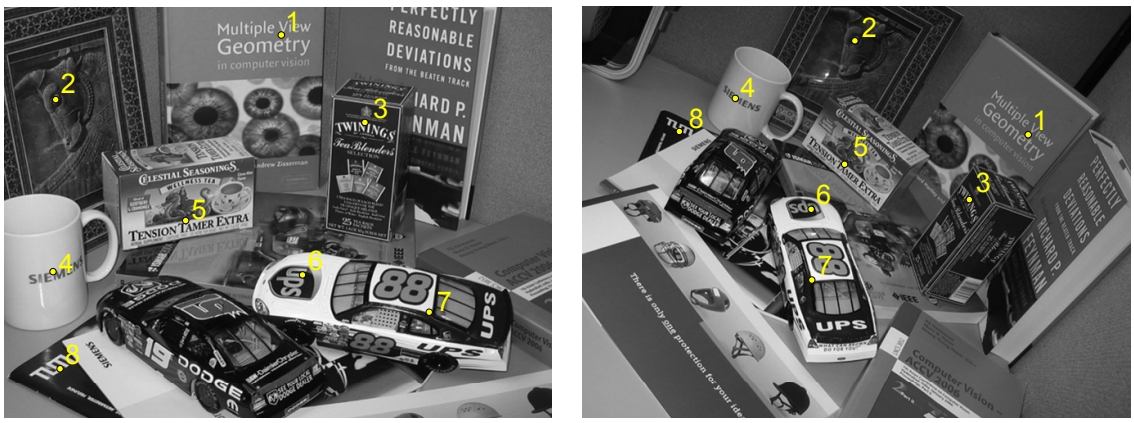


Figure 5.2: Two images from different views of the same scene. Feature points with the same number correspond.

Figure 5.3: The classic 'corner + fixed window' strategy for finding correspondences is not suited for wide-baseline conditions.

their respective windows were matched. In case of images with small-baseline (e.g. video sequence) this technique combined with additional search area constraints can give quite good results.

Feature corners allow stable and accurate detection, because their neighborhood offer two-dimensional signal variation. They are easier to localize than points on homogeneous areas, or along straight edges. However, the problem becomes more challenging with increasing viewpoint and orientation changes. The Harris corner detector is per definition invariant to translation and in-plane rotations (see Fig. 5.4). On the other side, fixed correlation windows can only be used for translation. This would only apply for features on planes parallel to the image plane with a translation parallel to the image plane. In case of rotation the correlation windows can not be compared directly anymore. Under the general viewpoint changes and rotations, we observe a larger class of image transformations. Translation along the optical axis or variations in the local length, cause the change in the size of patches, namely *scale change*. Rotations about other axis can cause the patches to

be stretched along one direction. In these cases we observe that the correlation windows do not cover the same image content, i.e. the same physical surface.

Clearly, the 'corners + fixed window' strategy can not cope with this class of transformations and fails when the images are taken in these *wide-baseline* conditions (see Figure 5.3). Next section describes how a corner or blob detector can be combined with automatic scale selection to obtain a *scale invariant detector*. The scale invariant approach will then be extended in section 5.5 to make it invariant to the general class of affine transformations.

5.3 Scale invariant detectors

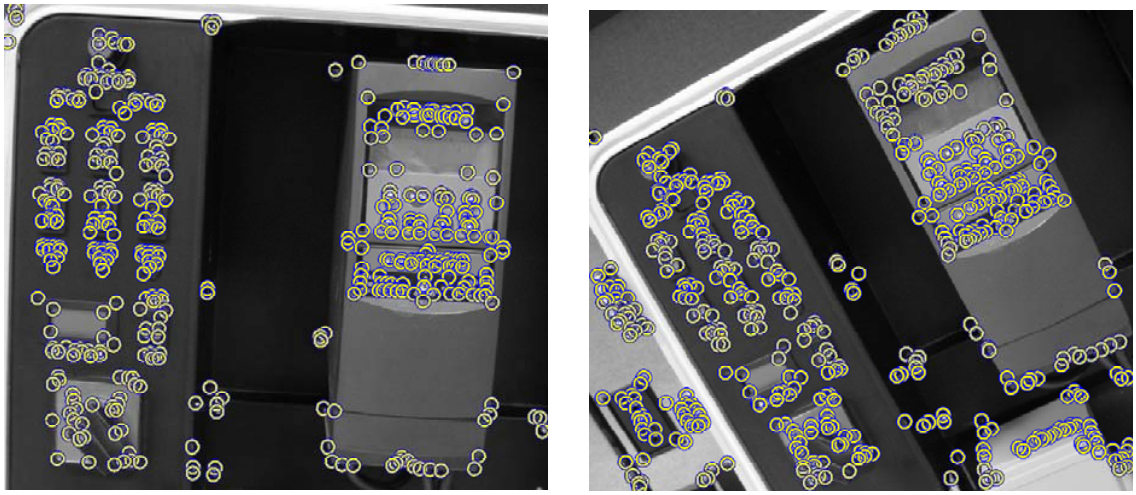
Many approaches have been proposed for extracting scale invariant regions. In the early eighties Crowley [25, 26] introduced a method that searches for local extrema in the scale space representation of an image (x,y,$scale$). In this approach the pyramid representation is computed using difference-of-Gaussian filters. Feature points are identified by local extrema in the scale space. The existing approaches mainly differ in the different ways used to build the scale space representation. Lindeberg [74] searches for local maxima of the scale normalized differential operator. He proposes to use the Laplacian-of-Gaussian (LoG) and several other derivative based operators. The LoG is circularly symmetric and detects blob-like structures. The scale space representation is built by successive smoothing of the image with Gaussian based kernels of different sizes.

David Lowe [79] approximates the LoG with difference-of-Gaussian (DoG) filters and also detects local extrema in the scale space pyramid. Using DoG instead of LoG significantly accelerates the computation process. This method will be described in more detail in the next section.

A common drawback of the DoG and LoG representation is that local maxima can be detected close to contours or straight edges, where the signal change is only in one direction. Clearly, the features are less stable, since their localization is more sensitive to noise or small changes in neighboring texture. Therefore, they need to be eliminated in a post-processing step. Another approach solving this problem, is to select the scale for which the trace and the determinant of the Hessian matrix simultaneously assume a local extremum. As a result, this would penalize points for which the second derivatives detect signal changes only in one direction. This detector was introduced by Mikolajcyzk [91] and is coined the *Hessian-Laplace detector*. Similarly, points can be detected by the scale adapted Harris function and selected in scale space by the LoG operator. In contrast to Hessian-Laplace detector that detects blob-like structures, this detector called the *Harris-Laplace detector* detects corner-like structures (see Fig. 5.5(b)-(c)).

A different approach for scale selection was proposed by Kadir and Brady [53]. It explores the idea of using local complexity as a measure of saliency. The salient scale is selected at the entropy extremum of local intensity histograms. The method searches for scale localized features with high entropy, with the constraint that the scale is isotropic. However, we will not use this detector in our object detection system later on, because of its high computation time.

(a)



(b) Harris

Figure 5.4: (a) Test images. (b) Harris corner detector.

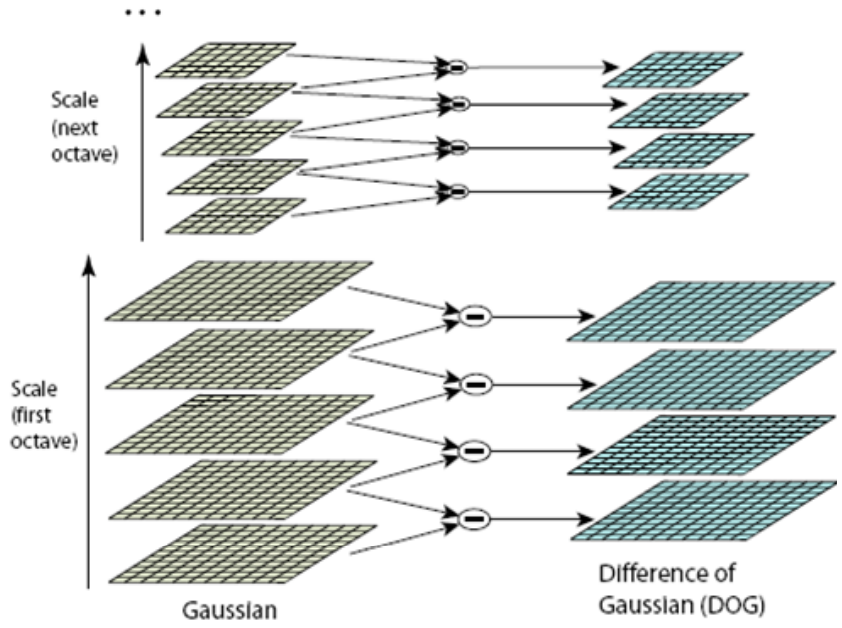


Figure 5.5: Scale space and DoG [80].

5.3.1 SIFT detector

This section describes a method invented by David Lowe [80] for detecting regions which are invariant to similarity transformations (i.e. in particular scale) and partially invariant to photometric changes. The detector is coined *Scale-Invariant Feature Transform (SIFT)*. The feature detection procedure consists of two major stages. The first stage searches over all possible scales and image locations to identify stable feature points that are invariant to scale and orientation. This is achieved by using a continuous function of scale known as *scale space*. The only possible scale-space kernel is the Gaussian function [65, 73]. The scale space of an image $I(x, y)$ is a function $L(x, y, \sigma)$ defined as

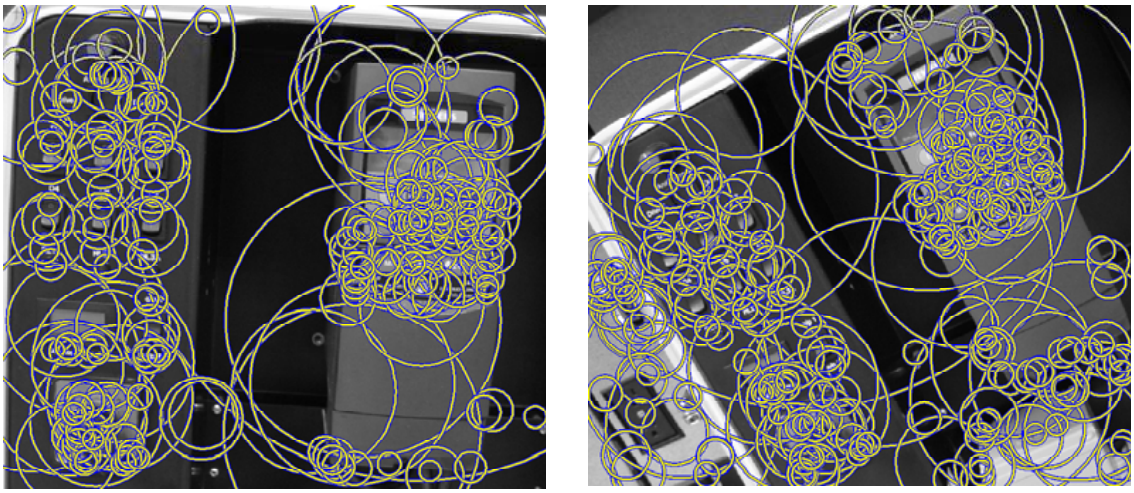
$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \text{ with } G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}, \tag{5.1}$$

where $G(x, y, \sigma)$ is a variable-scale Gaussian and $*$ is the convolution operation in x and y. Lowe has proposed an efficient algorithm to detect stable interest locations in scale space using scale-space extrema in the *difference-of-Gaussian (DoG)* function $D(x, y, \sigma)$ convolved with the input image:

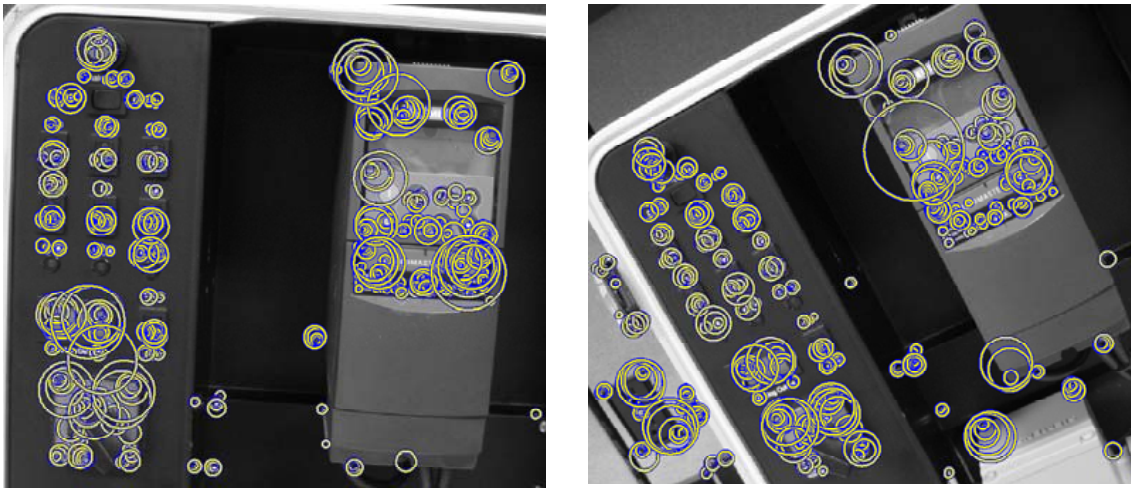
$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma), \tag{5.2}$$

where k is a constant factor.

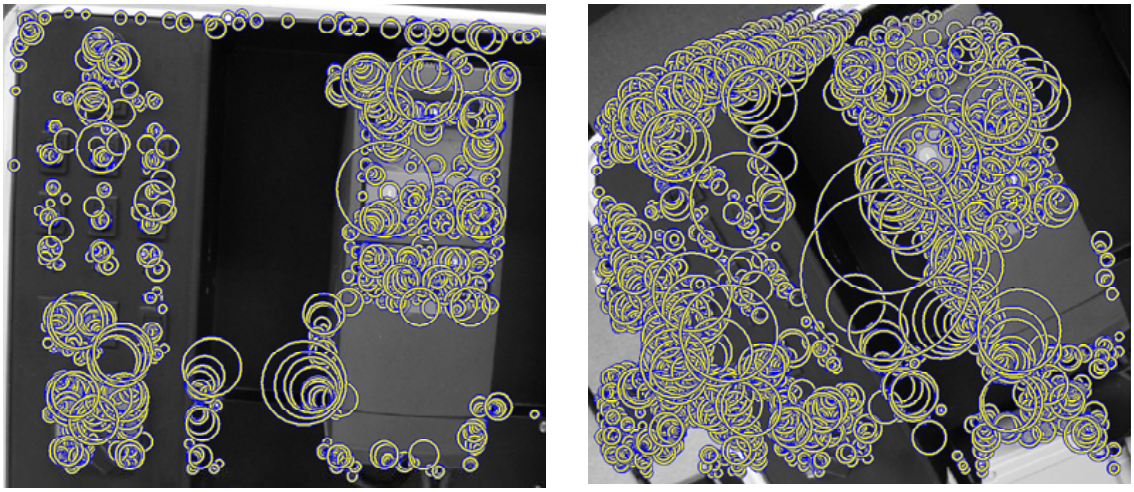
Lindeberg [73] proved that the DoG function provides a close approximation to the scale-normalized Laplcaian of Gaussian. In experimental comparisons, Mikolajczyk et al [90] showed that using the maximia and minima of $\sigma^2\nabla^2 G$ for scale selection yields the most stable features compared to other functions (see section 5.5.1). The DoG scale space $D(x, y, \sigma)$ can be constructed very efficiently as shown in Fig. 5.5. The initial image is incrementally convolved with Gaussians to produce a set of scale space images separated



(a) SIFT



(b) Harris-Laplace



(c) Hessian-Laplace

Figure 5.6: Scale invariant feature detectors.

by a constant factor k. Each octave of the scale space is divided by s, so that $k = 2^{1/s}$. Adjacent Gaussian images are subtracted to produce the set of DoG images (see Fig. 5.5). After a complete octave is processed, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

Feature locations are identified by local maxima and minima of the DoG scale space $D(x, y, \sigma)$. They are found by comparing each sample point to its eight neighbors in the current image and nine neighbors in the scale above and below. Lowe showed in experimental comparisons that the scale space DoG function has a large number of extrema and it would be very expensive to compute them all. But fortunately, the most stable and useful subset can be detected with a coarse sampling of scale as low as three per octave [80].

Once a feature point candidate has been found by comparing a pixel to its neighbors, the next stage consists of a detailed fit to the nearby data for location, scale and ratio of principal curvatures. Based on this information edge points and points with low contrast which are sensitive to noise, can be rejected. Brown et al [19] developed a method for fitting a 3D quadratic function (Taylor expansion of the DoG scale space function) to the local sample points to determine the interpolated location of the maximum. This proved to significantly improve the matching and stability.

Furthermore, feature locations along edges (where the DoG function has a strong response) are eliminated in a post-processing step. This can be achieved by computing the ratio of the principal curvatures from the Hessian matrix at the location and scale of the feature point. The eigenvalues of the Hessian matrix are proportional to the principal curvatures (see also section 5.5.1). Since we are only interested in the ratio, there is no need to compute the eigenvalues explicitly. Their sum and product can be computed from the trace and the determinant of the Hessian matrix H, respectively. Let r be the ratio between the largest magnitude eigenvalue and the smaller one. It turns out, that $\mathbf{trace}(H)^2/\mathbf{Det}(H) = (r+1)^2/r$. Therefore, to check whether the ratio of principal curvatures is below some threshold r we only need to check

$$\frac{\mathbf{trace}(H)^2}{det(H)} < \frac{(r+1)^2}{r}. \tag{5.3}$$

This can be computed very efficiently, and experiments show good results using a a value of r=10 (see Fig. 5.5(a)).

One major advantage of SIFT features is their extreme computational efficiency. The cost of extracting these features can be minimized by taking a cascade filtering approach, in which the more expensive operations are applied only at locations that pass an initial test. The original implementation of the SIFT detector was kindly provided by David Lowe. To achieve an optimized version for our object detection system, we have re-implemented the detector for the most part in C++ using Intel's IPL Library and could obtain up to 300% speed up in time. Typical extraction times are around 300 ms for VGA images on a Pentium 4, 2.8GHz.

Furthermore, for an automated feature evaluation system a SIFT feature tracking system (*SIFT Tracker*) has been implemented. The basic idea is to start with an initial set of SIFT features that have been extracted from the scale space in an arbitrary frame, and track those features in the consecutive frames. Correspondences are found based on the respective SIFT descriptors (see section 5.7.2). Moreover, the search space is

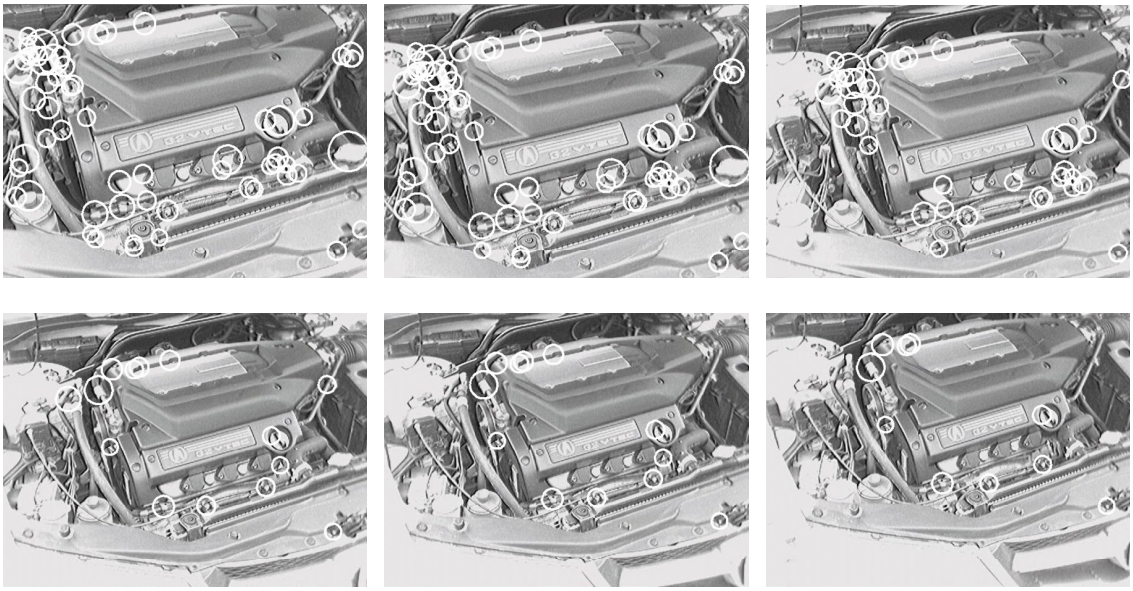


Figure 5.7: The SIFT Tracker.

constrained both in the image as well as the scale space, i.e. the features are searched for in the corresponding neighboring DoG images only. Figure 5.7 shows a set of frames from a long sequence taken from an engine. The first frame shows the initial features which are tracked through the sequence. At the end of the sequence only the most stable and reliable features survive as shown in the last frame. This procedure is performed in multiple keyframes within a sequence in order to cover the object of interest from different sides. This process has been integrated into an automated evaluation system which will be described in the next chapter.

5.4 Affine covariant regions

SIFT features described in the previous section are invariant to image scaling and rotation, but only partially invariant to 3D camera viewpoint changes. This is illustrated in Fig. 5.8(a)-(b). As can be seen, the circle can not cover the same image content, i.e. the same physical surface. Clearly, a circular region can not cope with the class of geometric deformations by the change in 3D viewpoint. The shape of the region has to be adaptive to viewpoint changes. A crucial observation is that most of the small feature regions cover an approximately *planar* surface of the object. Since a region is small with respect to the distance from the camera, perspective effects can be neglected. In this condition, two images of a small, planar region are geometrically related by an *affine transformation*. This reveals the nature of the desired feature regions: the regions that can be extracted in an affine invariant manner. Hence, they automatically adapt their shape so as to keep on covering the same physical surface patch in any image. These regions are called *affine invariant regions*, or more precisely *affine covariant regions*, since the shape of the regions changes with the image transformation and does not stay constant. In contrast to the scale invariant regions (see previous section) in the case of affine covariant regions the

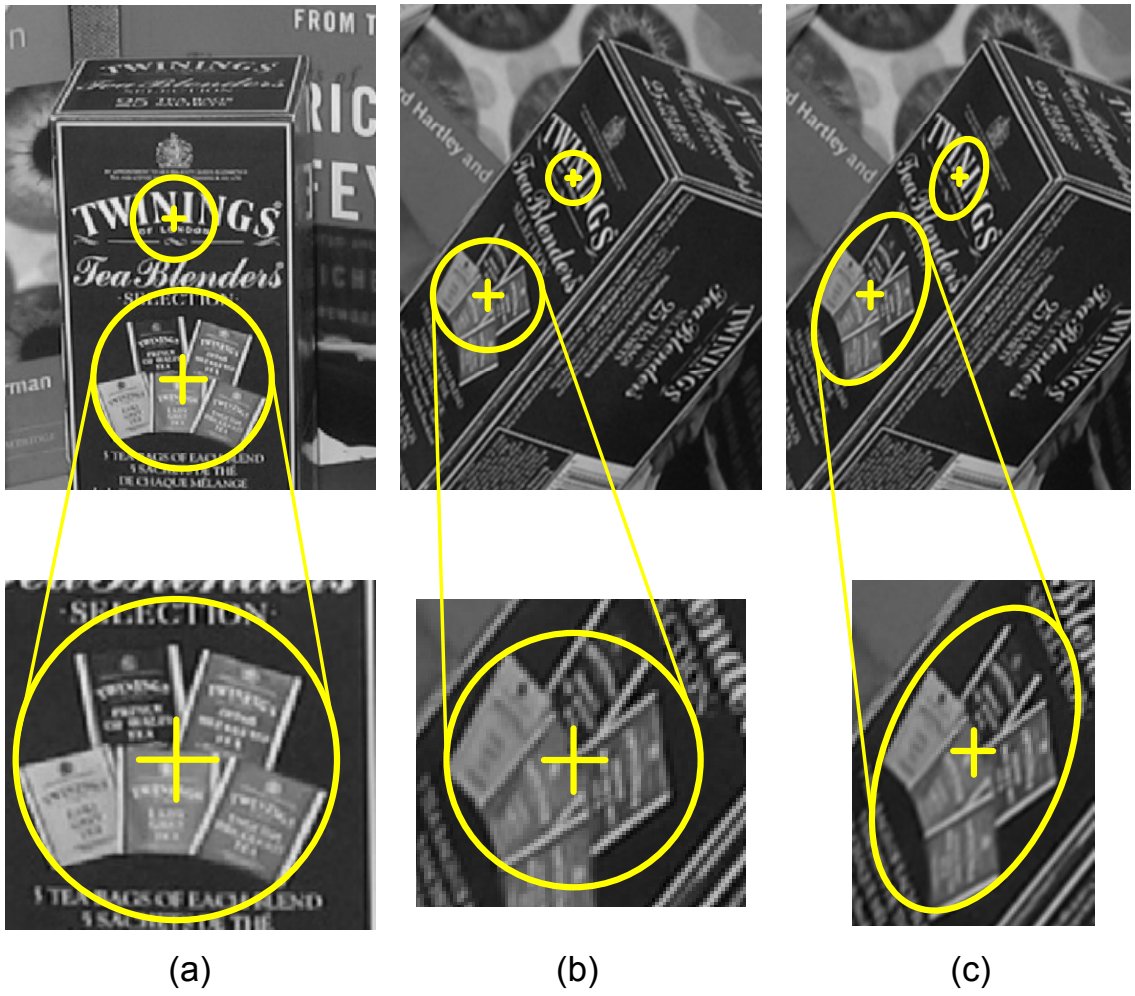


Figure 5.8: Scale invariant regions vs. affine invariant regions. (a) First viewpoint and the close-up of the image of a feature region. (b) Second viewpoint. Scale invariant circular regions clearly do not suffice to deal with general viewpoint changes. The class of transformations needed is affinity. (c) Second viewpoint with an anisotropic rescaling of the original region, i.e. affinity.

Detector	# Features	Time[s]
Harris Laplace	519	0.580
Hessian Laplace	1081	0.217
SIFT	801	0.195

Table 5.1: Computation time of scale invariant region detectors.

scaling can be different in each direction (see Fig. 5.8(c)).

Several region detectors have been developed in the last few years, differing in the property they look for, and therefore in which image regions they detect. The next section describes the main ideas behind the best known affine covariant detectors.

5.5 Affine covariant region detectors

This section gives a brief overview of the state of the art in affine covariant region detection. These detectors can be seen as a generalization of the scale invariant detectors described section 5.3. The object detection algorithm described in the next chapter is based on local features and can work with any of these detectors.

One of the first methods for finding blob-like affine regions has been developed by Lindeberg and Garding [75] in the context of shape from texture. The method explores the properties of the second moment matrix and estimates in an iterative procedure the affine transformation of local patterns. The features are extracted using the maxima of a uniform scale-space representation and to iteratively modify the scale and shape of features. However, the feature location is detected only at the initial step of the algorithm by not affine covariant Laplacian measure. Therefore, if the pattern undergoes a significant affine deformation, i.e. 3D viewpoint change, the spatial location of the maximum can be slightly different. Baumberg [11] used the affine shape estimation for matching and recognition. He extracts interest points at several scales using the Harris detector and then adapts the shape of the point neighborhood to the local image structure using the iterative procedure proposed by Lindeberg[75]. The affine shape is estimated for a fixed scale and fixed location, i.e. the location and scale are not extracted in an affine covariant way. Therefore the feature regions are not covariant to large affine transformations. Kadir et al [54] measure the entropy of pixel intensity histograms computed for elliptical regions to find local maxima in affine transformation space.

Schaffalitzky and Zisserman [124] extended the Harris-Lapace detector (see section 5.3) by affine normalization proposed by Baumberg [11]. A similar idea was explored by Mikolajczyk and Schmid [90]. Section 5.5.1 describes the related methods *Harris-Affine* and *Hessian-Affine* detectors [91, 93]. Section 5.5.2 and 5.5.3 describe two types of affine covariant regions introduced by Tuytelaars and van Gool [143, 144], one based on a combination of interest points and edges coined the *edge-based region* detector (*EBR*), and the other one based on image intensities, referred to as *intensity extrema-based region* detector (*IBR*). Finally, section 5.5.4 describes the *maximally stable extremal region* detector (*MSER*) developed by Matas et al. [85]. The detectors proposed by Lindeberg [75] and Baumberg [11] have not been included since they are very similar to the Harris-/Hessian-Affine detectors.

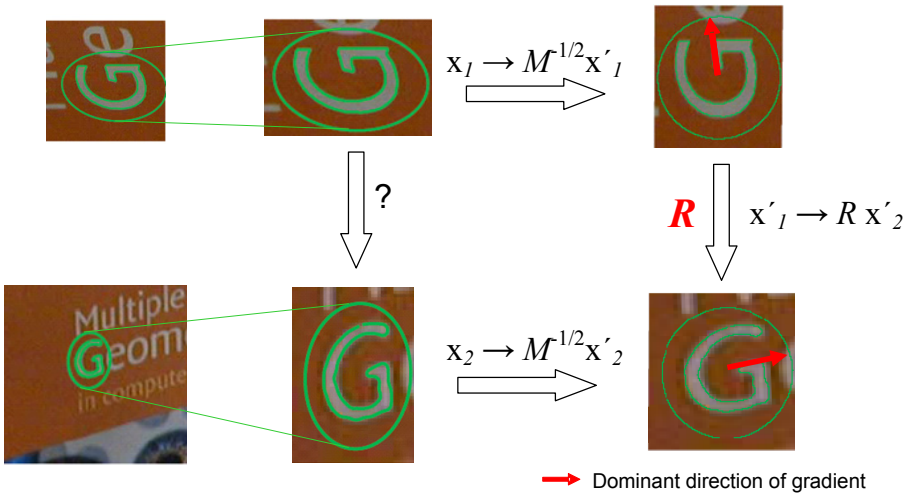


Figure 5.9: Affine Normalization using the second moment matrices Miko04, Baumberg00.

5.5.1 Harris-/Hessian-Affine detector

Baumberg and Lindeberg [11, 75] introduced a region detector which follows two steps. First, the location and scale of the regions is determined by a multi-scale Harris detector or a detector based on the Hessian matrix. In the second step, an adaptive procedure is applied to determine the shape of an elliptical region. Thereby, scale selection is based on the Laplacian and the elliptical shape is determined with second moment gradient matrix. The second moment gradient matrix describes the gradient distribution in a local neighborhood of a point and is used both in the Harris detector and the elliptical shape estimation:

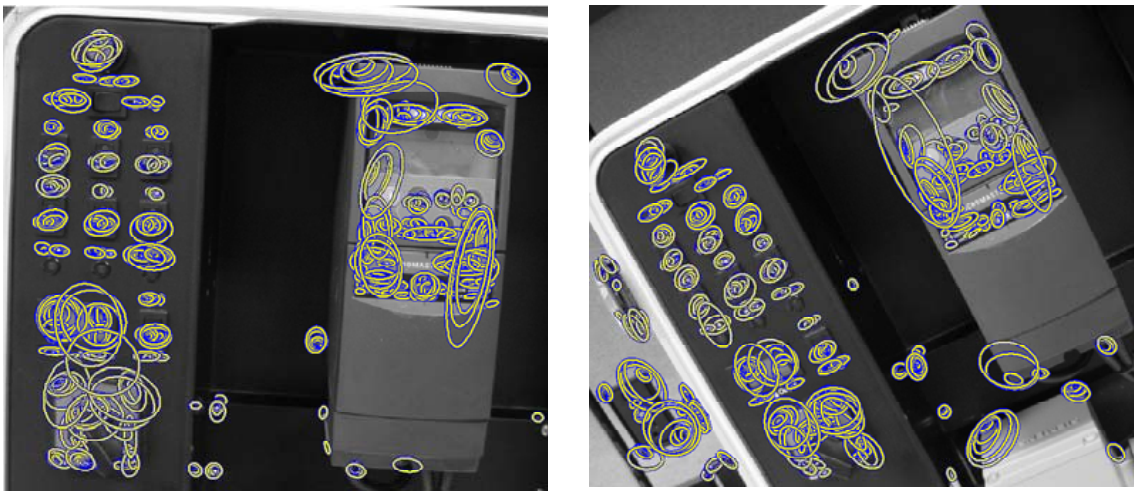
$$M = \mu(x, \sigma_I, \sigma_D) = \sigma_D^2 g(\sigma_I) * \begin{bmatrix} I_x^2(x, \sigma_D) & I_x I_y(x, \sigma_D) \\ I_x I_y(x, \sigma_D) & I_y^2(x, \sigma_D) \end{bmatrix} \tag{5.4}$$

I_a is the image derivative computed in a direction. The local derivatives are computed with Gaussian kernel of size determined by scale σ_D (*differentiation scale*). Then, the derivatives in the neighborhood of the point are averaged using smoothing with a Gaussian window of scale σ_I (*integration scale*). Stable corner points can be characterized by points for which the signal change is significant in orthogonal directions. The two principal signal changes in a neighborhood of a point are represented by the two eigenvalues λ_1 and λ_2 of the matrix M. Therefore, image points with respective large eigenvalues can be classified as corners. Similarly, flat regions can be characterized by small eigenvalues. Many interest point detectors, e.g. the Harris detector [47] rely on this principle. The Harris *measure of corner response* R combines the trace and the determinant of the second moment matrix M:

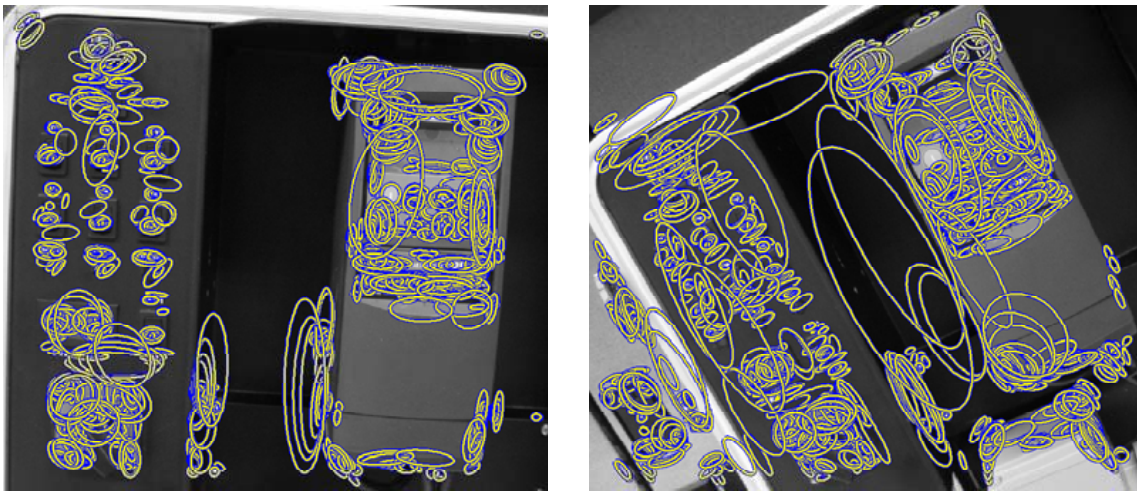
$$R = \mathbf{det}(M) - k\mathbf{trace}^2(M), \quad \text{with} \tag{5.5}$$

$$\mathbf{det}(M) = \lambda_1 * \lambda_2 \quad \text{and} \quad \mathbf{trace}(M) = \lambda_1 + \lambda_2, \tag{5.6}$$

where k is an empirical constant. Local maximas of the corner response determine the location of interest points.



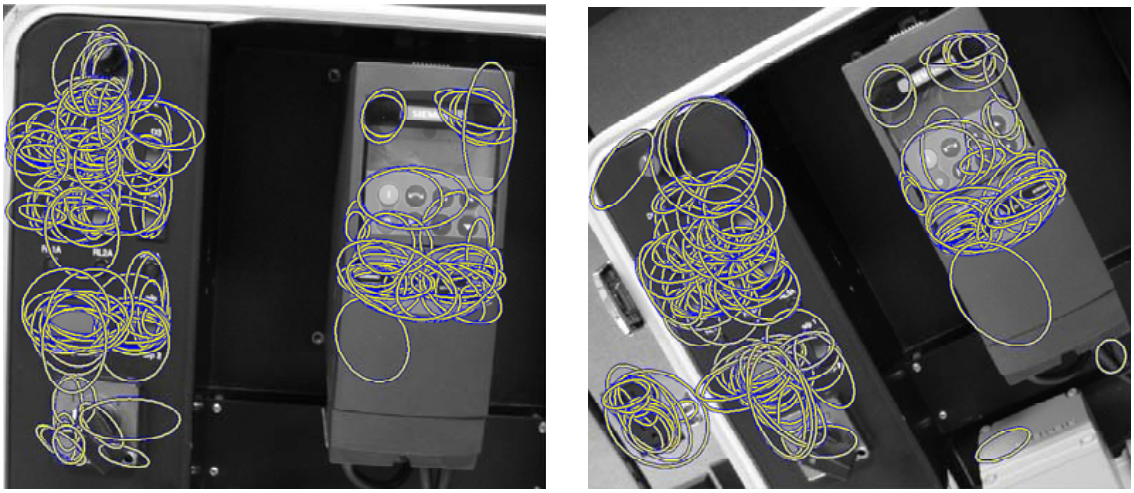
(a) Harris-Affine



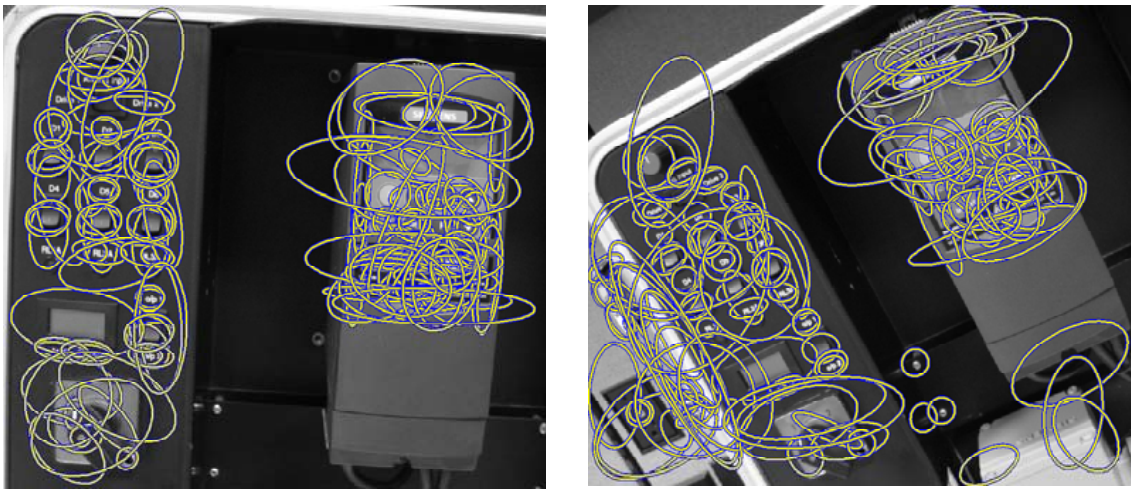
(b) Hessian-Affine

(c) MSER

Figure 5.10: Affine covariant feature detectors.



(a) EBR



(b) IBR

Figure 5.11: Affine covariant feature detectors.

A similar detector is the detector based on the Hessian matrix H:

$$H = H(x, \sigma_I, \sigma_D) = \sigma_D^2 \begin{bmatrix} I_{xx}(x, \sigma_D) & I_{xy}(x, \sigma_D) \\ I_{xy}, \sigma_D) & I_{yy}(x, \sigma_D) \end{bmatrix} \tag{5.7}$$

The second derivatives in the Hessian matrix H give strong responses on blobs and ridges. Since for long structures like for edge points the second derivative in one direction is very small, local maximas of the determinant would indicate the presence of a blob structures. In order to deal with scale changes we would need an automatic scale selection method. Lindeberg introduced the following approach for this purpose. The basic idea is to select the *characteristic* scale of a local structure, for which a given function attains an extremum over scales. The selected scale is characteristic in the quantitative sense, since it measures the scale with maximum similarity between the feature detection operator and the local image structure [74]. Therefore, the size of the regions is selected independently of image resolution. It is related to the structure and not to the resolution at which the structure is presented.

The experimental comparison in [90] showed, that for scale selection the Laplacian operator gives both in Harris and Hessian detector the best results, i.e. gives the highest percentage of correct characteristic scales to be found. When the size of the Laplacian kernel matches with size of a blob-like structure the response attains an extremum. It also provides good estimation of the characteristic scale for other local structures like corners, edges, ridges and multi-junctions.

Once we have the set of initial point locations and their characteristic scales, an iterative procedure can be applied to recover the shape of the elliptical affine region. The eigenvalues of the second moment matrix M are used to estimate the affine shape of the point neighborhood. To do so, the transformation that projects the affine pattern to the one with equal eigenvalues is computed, which is given by square root matrix $M^{1/2}$. Given two corresponding points $\mathbf{x}_1$ and $\mathbf{x}_2$ in two views, the neighborhood of these points are normalized by the following transformations: $\mathbf{x}'_1 = M_1^{1/2}\mathbf{x}_1$ and $\mathbf{x}'_2 = M_2^{1/2}\mathbf{x}_2$, respectively. Now, the normalized regions are related by a rotation R only: $\mathbf{x}'_2 = R\mathbf{x}'_1$ [11, 75]. The matrices M'_1 and M'_2 computed in the normalized frames are then equal to a rotation matrix. The affine deformation can be determined up to a rotation factor, since the rotation preserves the eigenvalue ratio for an image patch. However, this factor can be determined by other methods, e.g. normalization based on the dominant gradient orientation (see section 5.7.1).

This technique can be used to estimate the shape of the initial regions provided by the Harris or Hessian based detector. Mikolajczyk and Schmid [91, 93] proposed an iterative region estimation procedure consisting of the following steps:

1. *Detection:* detect initial region with Harris or Hessian detector and select the characteristic scale
2. *Shape estimation:* use the second moment matrix M to estimate the shape of the region
3. *Normalization:* normalize the region to a circular one
4. *Verification:* if for the new point the eigenvalues of M are not equal, goto step 2.

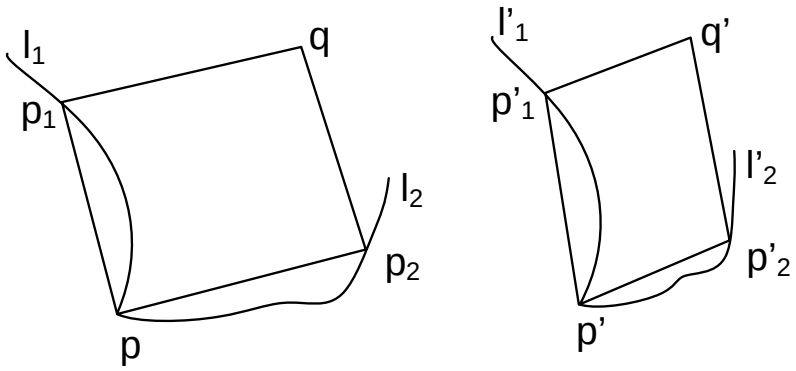


Figure 5.12: Edge based region detector.

5.5.2 Edge-based region detector (EBR)

Tuytelaars and Van Gool [142, 144] proposed a geometry based method to detect affine covariant regions by exploiting curved and straight edges in the image. The extraction process starts from Harris corner points and the edges that can be found close to such a point (using Canny edge detector [20]). By exploiting the edge geometry, the dimensionality of the problem can be reduced: once the anchor points are found (Harris corner points) the 4D search problem over all possible affinities can be reduced to a one-dimensional problem by exploiting the nearby edges geometry. Let $\mathbf{p}$ be a Harris corner point on an edge (see Fig. 5.12). Two points $\mathbf{p}_1$ and $\mathbf{p}_2$ move away from the corner in both directions along the edge. Their relative speed is coupled through the equality of relative affine invariant parameters l_1 and l_2:

$$l_i = \int |\mathbf{det}(\mathbf{p}_i^{(1)}(s_i)\mathbf{p} - \mathbf{p}_i(s_i))| ds_i, \ \mathbf{for}\ i = \{1, 2\} \tag{5.8}$$

where s_i is an arbitrary curve parameter, $\mathbf{p}_i^1(s_i)$ is the first derivative of $\mathbf{p}_i(s_i)$ with respect to s_i. This condition is used as an affine covariant criterion, which prescribes that the area between $\mathbf{pp}_1$ and the edge, and between $\mathbf{pp}_2$ and the edge remain identical. At each position and for each value l (l is used when referring to $l_1 = l_2$), the two points $\mathbf{p}_1(l)$ and $\mathbf{p}_2(l)$ with the corner $\mathbf{p}$ define a region $\Omega(l)$: the parallelogram spanned by the vectors $\mathbf{p}_1(l) - \mathbf{p}$ and $\mathbf{p}_2(l) - \mathbf{p}$. This gives a one dimensional family of parallelogram-shaped regions as a function of l. From this family one or a few parallelograms are selected for which some photometric quantities of the texture covered by the parallelogram go through an extremum. See [142, 144] for more detail. The parallelograms can then be replaced by the enclosed ellipses (see Fig. 5.12).

One of the drawbacks of this method is that the edges it relies on are often a source of errors. Edge detected in one image, may not be detected in the other image. They might be interrupted or connected in a different way in the other image. Tuytelaars and Van Gool also presented an alternative method for extracting affine covariant regions, that is directly based on the analysis of the image intensity. This method is briefly described in the next section.

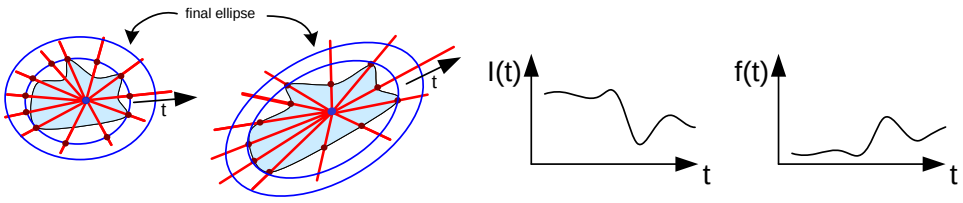


Figure 5.13: Intensity based region detector [144].

5.5.3 Intensity-based region detector (IBR)

Tuytelaars and Van Gool [143, 144] proposed furthermore a method to detect affine covariant regions that start from intensity extrema (used as *anchor points*), and explores the image around them in radial way, delineating regions of arbitrary shape, which are then replaced by ellipses. For this purpose the intensity function along rays emanating from the extremum is studied (see Fig. 5.13). Along each ray the following function f_I is evaluated:

$$f_I(t) = \frac{|I(t) - I_0|}{max\left(\frac{\int_0^t |I(t)-I_0|dt}{t}, d\right)}, \tag{5.9}$$

where t is the Euclidean arc length along the ray, $I(t)$ is the intensity value at position t, I_0 is the intensity value at the extremum and d is a small number to prevent the division by zero. The point for which the function f reaches an extremum is invariant under the affine geometric and linear photometric transformations. This is usually the case when the intensity suddenly increases or decreases dramatically, e.g at the border of a homogeneous region. $f_I(t)$ is in itself invariant. However, for more robustness points where the function reaches an extremum can be selected.

Next, all points corresponding to maxima of $f_I(t)$ along rays from the same local extremum are linked to enclose an affine covariant region (see Fig. 5.13). This region irregularly-shaped region is replaced by an ellipse with the same shape moments up to the second order. To have more a distinctive texture for reliable matching, the size of the ellipse is doubled. This ellipse fitting operation is also an affine covariant construction.

Since the elliptical regions are not centered around the original anchor points, this procedure is robust to inaccurate localization of this point.

We implemented this approach in C++ using IPL and added the shape estimation procedure to the feature detector library. To achieve more robustness to large scale changes we extended this approach to a Gaussian multi-scale method with a scale space pyramid, similar to the SIFT detector (see section 5.3.1). Thereby, anchor points can be chosen to be either extremas of DoG or points detected by the Hessian detector (see section 5.5.1). The regions are extracted in each level independently and returned. Figure 5.14 shows some results of the extraction process for single anchor points.

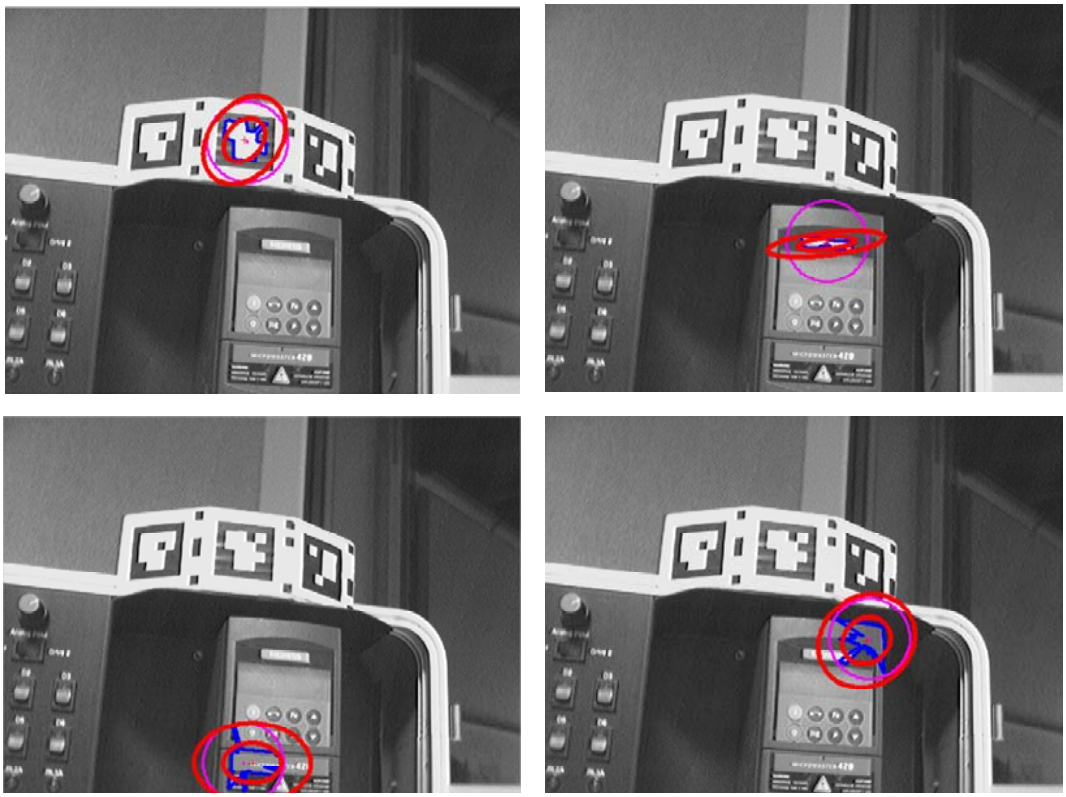


Figure 5.14: Intensity based region detector [144].

5.5.4 Maximally stable extremal region detector (MSER)

Jiri matas et al [85] designed *maximally stable extremal regions (MSER)*: connected components of pixels which are all either brighter (bright extremal regions) or all darker (dark extremal regions) than all the pixels on its regions boundary. The maximally stable describes the property that it is optimized in the threshold selection process, i.e. it considers all possible global thresholding of the greylevel image, and retains regions which change only little across a wide range of thresholds. The detected connected component of pixels are then replaced by an ellipse in the same manner as done by Tuytelaars and Van Gool (see section 5.5.3). The MSER detector achieves scale invariance in a natural way without searching or even building a scale space. The authors present a very time-efficient extraction procedure (near linear complexity) [85].

The set of all detected connected components C obtained by thresholding has the following properties. First, the set is closed under continuous (affine, homography or even projective) transformation of image coordinates, the topology is preserved pixels from a single connected component are transformed to a single connected component. Second, the set is closed under monotonic transformation of image intensities since it depends only on the ordering of pixel intensities which is preserved under monotonic transformation. This ensures that common photometric changes modeled locally as linear or affine leave C unaffected.

In an effort to maximize distinctiveness while still handling occlusions and keep the regions approximately planar, each region is produced in 3 differently sized versions (scaled 1x, 2x and 4x respectively). The matching algorithm then attempts to use the largest ones first.

5.6 Complexity and computation time

The computational complexity of the algorithm for finding initial points in all the detectors except MSER is $O(n)$, where n is the number of pixels [93].

Harris-/Hessian-Affine: For the Harris-Affine and Hessian-Affine detectors the complexity of the automatic scale selection and shape adaptation algorithm is $O((m+k)p)$, where p is the number of initial points, m is a number of scales in the automatic scale selection and k is a number of iterations in the shape adaptation algorithm.

Intensity extrema-based: For the IBR region detector, the complexity of constructing a region around the intensity extrema is $O(p)$, where p is the number of intensity extrema.

Edge-based region detector: For the EBR region detector, the complexity of constructing the actual region starting from the corners and edges is $O(pd)$, where p is the number of corners and d is the average number of edges nearby a corner.

Maximally Stable Extremal Region Detector: for the MSER detector, the computational complexity of the sorting step is $O(n)$ if the range of image values is small (e.g. < 255), since the sort can be implemented as bin sort. The complexity of the union-find algorithm is $O(nloglogn)$.

Computation times for the different detectors is presented in table 5.2. The computation times have all been measured on a Pentium 4, 2.8GHz PC , for the first test image shown in Fig. 5.4(a) with 640x570 pixels. The computation time is dependent on the number of detected regions. The number of detected regions varies depending on

the object or scene type. However, this variety is also a virtue: the different detectors are complementary. Some detectors respond well to structured scenes others to more textured scenes [93]. We'll come back to this point in chapter 6.

5.7 Feature descriptors

Once feature regions have been extracted from the images with one of the methods described in the previous sections, the next step is to find corresponding features between the images. This *matching* process follows a common scheme in most approaches. First, the textured content of each feature region is represented with a relatively small number of measurements, which concisely describe the region and are at the same time discriminative enough to be used for comparison to find correspondences. This set of measurements are referred to as the *descriptor*. Ideally, the descriptor is invariant to affine geometric transformations and photometric changes. Photometric changes can occur because of a) variation in the local surface orientation with respect to the light source, and b) brightness change of the light source itself. Furthermore, the descriptor should be ideally distinctive, i.e. it contains the informative characteristics of a feature region, which makes it distinguishable from other regions.

The region detectors described in the previous sections provide circular or elliptic regions of different size, which depend on the detection scale. In order to be able to compare these regions from different views with each other, we first need a way to align them. Therefore, first step toward computing the descriptor for a region is geometric normalization: the region which in most cases is elliptical is transformed into a unite circle, i.e. the affine transformation is removed. This process is similar to the affine normalization described in section 5.5.1. The regions might then be related by an arbitrary rotation. Many descriptors have been proposed in the literature which are already rotation invariant. Examples are combinations of color-based moments [143], linear filter banks [124], normalized Gaussian derivatives steered in the gradient direction [91], or spin images [67]. Another approach is to assign a consistent orientation to each region based on the local image properties, so that the descriptor can be represented relative to this orientation and therefore achieve invariance to image rotation. In this way, non-rotation invariant descriptors can be used or just sampling pixels. Next section describes briefly an orientation assignment that was found among a number of approaches to give the most stable results [80].

Many different descriptors have been proposed in the literature. Mikolajczyk and Schmid [89] have evaluated the performance of descriptors by applying them to different region types and different image transformations. They observed that the ranking of the descriptors is mostly independent of the region detector and that SIFT based descriptors perform best. We will use the SIFT and PCA-SIFT descriptors for our object detection system that will be introduced in the next chapter. Section 5.7.2 and 5.7.3 describe the principle ideas of these descriptors.

5.7.1 Orientation assignment

After the geometric normalization step described in the previous section, the scale of the feature region is used to select the image I with the closest scale, so that all computations are performed in a scale invariant manner. In the case of the SIFT detector we select the

Detector	# Features	Time[s]
Harris affine	515	0.691
Hessian affine	842	0.450
EBR	147	1.234
IBR	135	0.652
MSER	167	0.211

Table 5.2: Computation time of affine covariant region detectors.

closest Gaussian smoothed image (see section 5.3.1). Then, an orientation histogram is built from the gradient orientations of all sample points within the region. Thereby, each sample in the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window. The gradient magnitude $m(x, y)$ and orientation $\theta(x, y)$ for each image sample $I(x, y)$ are computed by:

$$m(x,y) = \sqrt{(I(x+1,y) - I(x-1,y))^2 + (I(x,y+1) - I(x,y-1))^2} \tag{5.10}$$

$$\theta(x,y) = tan^{-1}\left(\frac{I(x,y+1) - I(x,y-1)}{I(x+1,y) - I(x-1,y)}\right) \tag{5.11}$$

The dominant gradient orientation is given by the highest peak in the orientation histogram [80]. For a better accuracy, a parabola is fit to the closest histogram values to the highest peak to interpolate the peak position.

5.7.2 SIFT Descriptor

David Lowe [80] proposed one of the most efficient and powerful descriptors coined the *SIFT descriptor* to describe the content of a feature region. A SIFT descriptor is created by first sampling the gradient magnitude and orientation around the feature location. Figure 5.15(a) illustrates this procedure. Thereby, the coordinates of the gradient orientations are rotated relative to the dominant gradient orientation (see section 5.7.1), in order to achieve rotation invariance. For more efficiency, the magnitude and orientation values are precomputed in the orientation assignment procedure. These are weighted with the Gaussian window, which is indicated by the overlaid circle in Fig. 5.15(a). This gives less emphasis to gradients that are far from the center of descriptor, since they are most affected by the misregistration errors [80]. Figure 5.15(b) shows the respective descriptor. The region is partitioned into an array of 2x2 blocks by creating orientation histograms over 4x4 sample regions. Each block the orientation histogram has eight bins, illustrated by arrows in Fig. 5.15(b), with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction. This brings robustness against local positional shifts, since a gradient sample in the feature region can shift up to 4 sample positions ans still contribute to the same histogram. Furthermore, in order to avoid boundary effects, trilinear interpolation is used to distribute the value of each gradient sample into adjacent histogram bins.

The SIFT descriptor is formed from a vector containing the values of all the histogram entries. Lowe's experiments showed that best results are achieved with a 4x4 array of

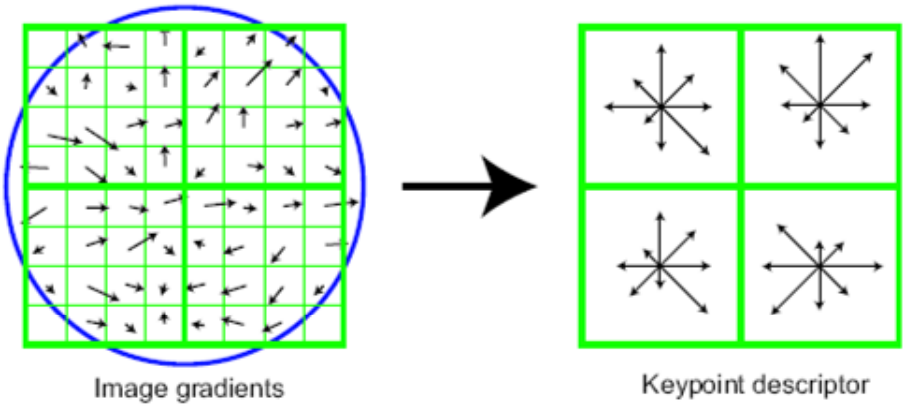


Figure 5.15: SIFT descriptor. Histogram of gradient orientations [80].

histograms each of which contains 8 orientation bins. Therefore, the SIFT descriptor vector for each feature region has 128 (4x4x8) elements. In order to reduce the effect of illumination changes, the SIFT vector is then normalized to unit length. Moreover, the effect of non-linear illumination changes e.g. caused by camera saturation, or 3D viewpoint changes, can be reduced by thresholding the value in the unit SIFT feature vector to each be no longer that $c, (0 < c < 1)$ and then renormalizing to unit length. The threshold value of 0.2 could be determined experimentally based on the matching results conducted in [80].

5.7.3 PCA-SIFT Descriptor

The distinctiveness of the SIFT descriptor described in the previous section is achieved by using a high-dimensional vector (128 dimensions) representing histograms of image gradients in the local neighborhood for each feature location. Instead of using SIFT's smoothed weighted histograms, Ke et al [57] applied *Principal Components Analysis (PCA)* to the normalized gradient regions and proposed a PCA based version of the SIFT descriptor, coined *PCA-SIFT*. Their experiments showed that the PCA-based local descriptors are more distinctive, more robust to image deformations, and more compact than the standard SIFT representation [57]. This results in increased accuracy and faster matching performance, which makes it attractive in particular for fast object detection systems. We will make use of this descriptor in the next chapter, in order to build a compact representation of multiple-view descriptors.

PCA is a standard technique for dimensionality reduction that has been applied to many computer vision problems, e.g. object recognition [107, 97] or face recognition [141]. Because of its simplicity, PCA remains popular despite some shortcomings, such as implicit assumption of Gaussian distribution and the restriction to orthogonal linear combinations.

PCA-SIFT descriptor computation can be summarized in the three major steps: First, an eigenspace is pre-computed to express the gradient values within the regions. Second, the local image gradients are computed within each region. Third, the image gradient vector is projected into the eigenspace to obtain a compact descriptor vector.

In order to compute the PCA projection matrix, we use a set of training images (see

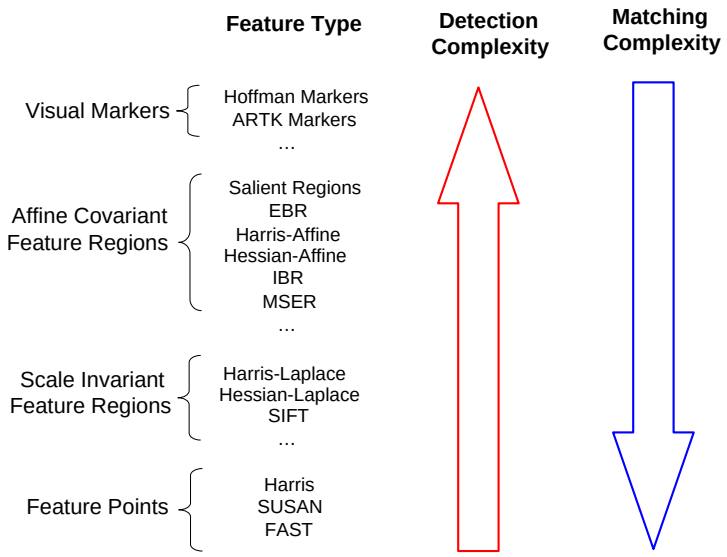


Figure 5.16: Feature detection vs. matching complexity.

next chapter for more detail). An input vector is formed by concatenating the horizontal and vertical gradient maps for a region. For instance the input vector of a normalized patch of size 41x41 pixels would have 3042 (2x39x39) elements. Similarly to the original SIFT descriptor the vector is normalized to unit magnitude, in order to minimize the impact of illumination variations. Having a set of input vectors, PCA is applied to the covariance matrix of these vectors. The matrix of the top n eigenvectors is then used as the PCA projection matrix.

To compute the PCA-SIFT descriptor for a given image region, the normalized image gradient vector is created in the same manner as before and projected into the eigen-sapce (*feature space*) using the pre-computed PCA projection matrix. A good value for the dimensionality of the feature space that has been determined empirically is $n = 20$. Compared to the standard SIFT representation with 128 elements this is a more compact representation leading to significant improvement in matching time performance. Experimental results show that variations in the input vectors, mainly due to the structure of the respective 3D model region and the distortions such as perspective effects caused by viewpoint changes, can be reasonably modeled by low-dimensional Gaussian distributions. Projecting the gradient vectors into the low-dimensional space seems to keep the structure related variations while discarding the distortions caused by other effects.

Although PCA is ill-suited for representing the general class of image patches, it seems to be very well-suited for capturing the variations in gradient image of feature regions that have been localized in space, scale and orientation.

5.8 Matching the regions

Once the region descriptors have been computed, we can now search for correspondences. Two regions are matched if their descriptors are very similar. Comparing descriptors, rather than directly regions, is not only faster, but also potentially more robust, since

descriptors often alleviate the effects of noise and misalignments. One measure of similarity can be defined as the Euclidean distance between the respective descriptor vectors. To avoid mismatches, a global threshold on distance to the closest descriptor would not perform well, as some descriptors are much more discriminative than others. We will introduce a statistical approach for this purpose in the next chapter using a set of training images. However, Lowe [80] showed that a *distance ratio test* is a more effective measure than the global thresholding method. Thereby, the distance of the closest neighbor is compared to that of the second-closest neighbor. Only matches are considered in which the nearest neighbor is less that n times the distance to the second-nearest neighbor. Matches for which the distance ratio is greater than n are eliminated. Good results are obtained already with with $n = 0.8$ [80].

Furthermore, the Mahalanobis distance can be used as a similarity measure. This distance accounts for the different variances of the descriptor's components, and their mutual covariances. The covariance matrix needed to compute the Mahalanobis distance can be estimated from a set of training images.

In order to avoid exhaustive search and computing the similarity for all pairs of descriptors, an indexing mechanism can be applied [80, 124, 91]. Indexing represents the descriptors as points in a vector space which are then organized in a data structure capable of efficiently finding nearest neighbors. Examples for different indexing data structures are the binary space partition trees [124], k-d trees [41] and variations of it [79, 125]. Lowe [13, 80] uses an approximate algorithm, called the Best-Bin-First algorithm (*BBF*).

Among all the feature extractors and matching procedures described in this chapter, the question arises which one of them is the most appropriate for an object detection system in terms of functional and computational time. To answer this question a comparison of the extraction and matching time is needed for each feature type, since both components are required for detection. Figure 5.16 shows how the complexity for extraction and matching relates among the different feature types: starting with the fastest one the FAST features [119], SUSAN [134], Harris corners [47], over scale invariant features, to the more sophisticated affine covariant features and finally the fiducial markers, such as ARToolKit makers. As shown in Fig. 5.16, with the increasing extraction complexity the matching complexity decreases for these features. As a matter of fact, the two factors are inverse-proportional. We have decided to choose affine covariant features since they provide a good trade-off between computational efficiency of the extraction and matching processes. Furthermore, the different covariant feature detectors are evaluated in the next chapter in terms of object detection performance.

5.9 Discussion

This chapter described the 2D-2D correspondence problem and its relation to the object detection problem. It presented the state of the art for covariant feature detection as well as robust and efficient matching. These techniques have been implemented and optimized to obtain near real-time performance.

In the next chapter we will introduce a novel technique for the 2D-3D correspondence problem between an image and a 3D object. However, the proposed system for object detection and pose estimation can work with any of the feature detectors that has been described in this chapter.

Chapter 6

Fusion of 3D and Appearance Models for Fast Object detection

I shall try to correct errors when shown to be errors, and I shall adopt new views so fast as they shall appear to be true views.

– Abraham Lincoln (1809 - 1865)

In the previous chapter the class of covariant feature region detectors and descriptors have been introduced. The power of using this kind of features for object detection lies in two factors. First, local features bring tolerance to partial occlusions and cluttered backgrounds. Second, since the extraction process and descriptors are invariant to scale changes or affine deformations, they bring robustness for matching under large viewpoint and illumination changes. Using covariant features this chapter presents a method for fast and robust object detection and pose estimation by fusing both the underlying 3D information and appearance of the features from multiple viewpoints [98, 99].

This chapter is organized as follows. After an introduction in the next section the proposed approach is described in more detail in section 6.2. Section 6.3 presents some experimental results followed by a performance evaluation of several feature extractors for object detection in section 6.4.

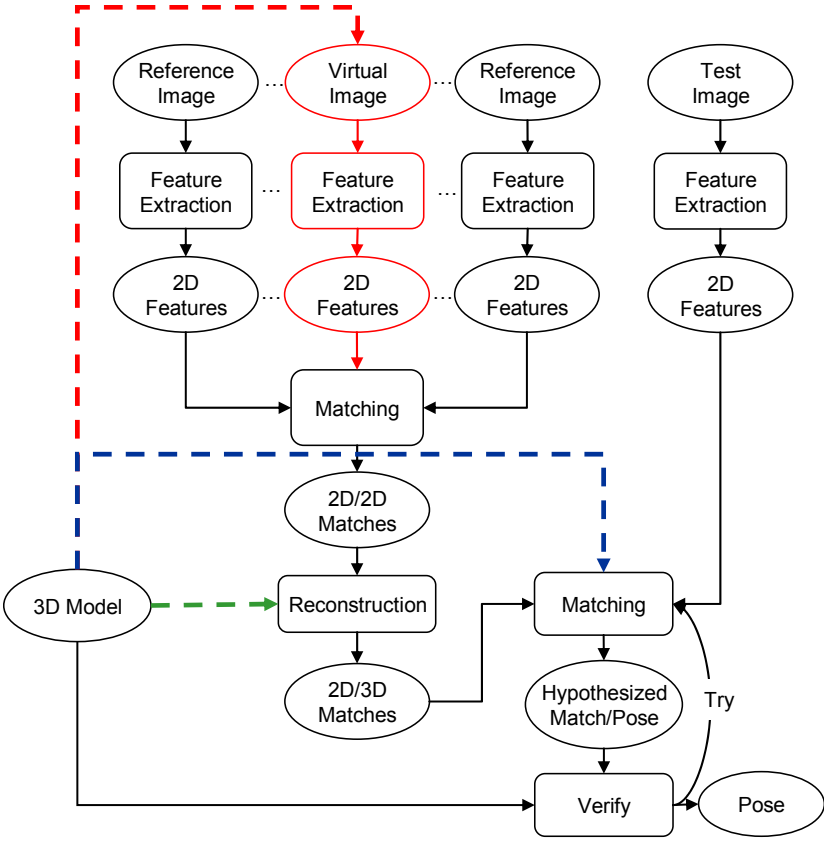


Figure 6.1: Overview of the proposed approach vs. conventional approaches.

6.1 Introduction

Computer vision literature includes many object detection approaches [37, 121, 146, 80, 87] based on representing objects of interests by a set of local features (bags of features) which are characterized by invariant descriptors for matching [127, 79, 148, 91, 124]. Chapter 3 gave an extensive literature review on feature based object detection methods. Local descriptors can be computed efficiently. See chapter 5 for implementation details and speed-up techniques. Combination of such descriptors provide robustness against partial occlusion and cluttered backgrounds. The descriptors are ideally invariant to viewpoint and illumination variations.

The use of local descriptors and the detection algorithms vary slightly between different approaches in the literature. Fig. 6.1 depicts an overview of the conventional object detection methods and the proposed approach. Most of the conventional methods make use of techniques for wide-baseline stereo (WBS) matching solely based on 2D images without considering any run-time requirements. Thereby, given a test image detection of the target object is usually done in three major steps (see Fig. 6.1). First, features are extracted independently and then matched in the second step with the features in the database. Matching is usually performed using photometric constraints combined with or without 2D geometry constraints [120]. Having enough 2D-3D correspondence candidates a pose is hypothesized. Third, geometric consistency constraints are enforced and used to

verify the hypothesized pose. If the verification does not succeed, the system goes to step two.

Limitations of these approaches are either time and/or functional performance with the increasing complexity of the objects or 3D/2D database. However, in many applications where real-time object detection is required both 3D models and several training images may be available or can be created easily during an off-line process. The key idea is to use the underlying 3D information to limit the number of hypothesis reducing the effect of the complexity of the 3D model on the run-time matching performance.

Our method differs in three aspects from the state of the art (see Fig. 6.1). First, for a sufficient statistical representation we augment appearance information from real images and 3D model with synthesized images for all those viewpoints where no real images are available. Second, we propose a statistical analysis and evaluation of the appearance and shape of features from all possible viewpoints in the scene combining real and synthetic viewpoints. Third, we make use of the known 3D geometry in both matching and pose estimation processes. We show that by fusing both appearance and geometric information rather than using them in separate procedures we can improve both time and functional performance. Especially for large environments this renders our method very powerful.

Our approach has two phases. In the training phase, a compact appearance and geometric representation of the target object is built. This is as an off-line process. The second phase is an on-line process where a test image is processed for detecting the target object using the representation built in the training phase. During training, the variations in the descriptors of each feature are learned using principal component analysis (PCA). Furthermore, for each feature a reliability measure is estimated by analyzing the computed visibility distribution from different viewpoints. The problem of finding matches between sets of features in the test image and on the object model is then formulated as a classification problem which is constrained by using the reliability measure of each feature.

Using this framework we perform an evaluation study of different feature detectors for object detection and pose estimation. Recently, Mikolajczyk et al. [93] presented a state of the art on affine region detectors and compared their performance on a set of test images under varying imaging conditions. The comparison showed that there does not exist one detector that systematically outperforms the other detectors for all scene types. The detectors are rather complementary, some are more adapted to structured scenes and others to textures.

However, given an arbitrary scene or target object it is unclear which type of features are more appropriate for matching and eventually whether their performance depends on viewing direction.

It is important to use an appropriate method to automatically select the best choice of feature detectors. In our industrial AR applications, we have the possibility of running an off-line evaluation procedure comparing the performance of different feature detectors on the learning set of real or synthesized images. The system will then automatically select the best detector for each scene and eventually for each set of viewing directions. In this chapter, we present an exhaustive evaluation of different feature detectors in conjunction with our initialization strategy. We then present detailed results of this evaluation. This needs to be considered as a clear demonstration of the system's general ability of evaluating and selecting the best feature detectors based on the defined metrics and requirements.

The evaluation of the feature detectors is performed in the context of matching and

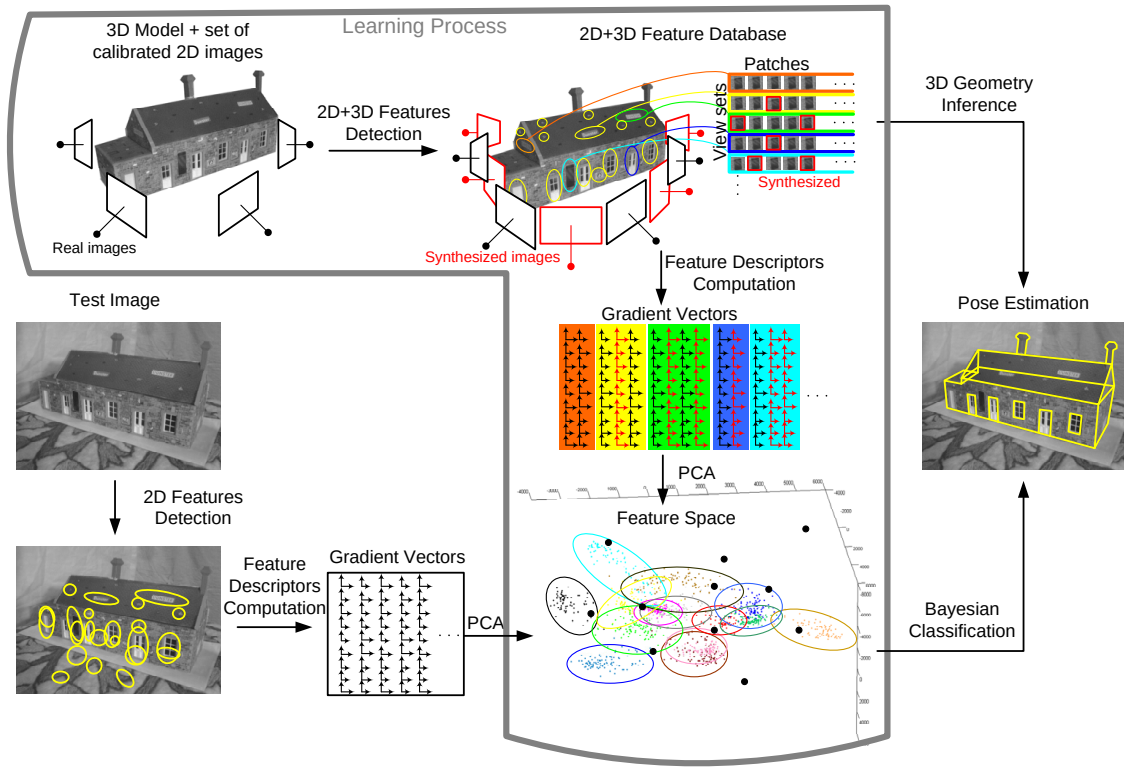


Figure 6.2: Overview of the proposed object detection process for real-time pose estimation.

detection ability of the same scene or target object observed under different viewing conditions with uncontrolled lighting. We have selected a number of state-of-the-art affine covariant feature detectors [93, 144, 85] and compare them using the same evaluation scenario and the same test data.

As an application, our method is intended to be used to provide robust initialization for a frame rate feature-based pose estimator [43] where robustness and time efficiency are very critical. In this case the initial pose recovery is sufficient to be performed under one second.

6.2 Proposed approach

Our goal is to automatically detect objects and recover their pose for arbitrary images (*test image*). The proposed object detection approach is based on two stages: A learning stage which is done off-line and the matching stage at run-time. The entire learning and matching processes are fully automated and unsupervised. Figure 6.2 gives an overview of the proposed framework. Sections 6.2.1 and 6.2.2 describe the learning step in more detail. In Sections 6.2.3 and 6.2.4 we introduce the matching and pose estimation algorithms that enforce both photometric and geometric consistency constraints.

6.2.1 Creating view sets based on similarity maps

In the first step of the learning stage a set of stable feature regions are selected from the object by analyzing their detection repeatability and accuracy as well as their visibility from different viewpoints.

Images represent a subset of the sampling of the so called *plenoptic function* [1]. The plenoptic function is a parameterized function for describing everything that can be seen from all possible viewpoints in the scene. In computer graphics terminology the plenoptic function describes the set of all possible *environment maps* for a given scene. In our case, we define a complete sample of the plenoptic function as a full spherical environment map (see Fig. 6.3(a)). Having a set of calibrated images and the virtual model of the target object, the viewing space is coarsely sampled at discrete viewpoints and a set of environment maps is created. Since not all samplings can be covered by the limited number of training images, synthesized views are created from other viewpoints using computer graphics rendering techniques.[1]

Next, affine covariant features [93] are extracted from the environment maps. In our experiments we evaluate the performance of different affine covariant detectors (see section 6.4). We also tested the scale and rotation invariant SIFT detector [80]. A hierarchical ranking of the features can be achieved using the respective 3D structure on the model (see Fig. 6.4).

We then select "good" feature regions which are characterized by their detection and descriptor performance. The evaluation of the detection and descriptor performance is done as following. The detection performance of a feature region is measured by the detection repeatability and accuracy. The performance of the descriptor is measured by the matching criterion, i.e. how well the descriptor represents a scene region. This is measured by comparing the number of corresponding regions obtained with the ground truth and the number of correctly matched regions (see Fig. 6.5(a)-(c)).

The basic measure of accuracy and repeatability is based on the relative amount of overlap between the detected regions in the environment maps and the respective reference regions projected onto that environment map using the ground truth transformation [93] (see Fig. 6.6(a)). The reference regions can be determined e.g. from the parallel views to the corresponding feature region on the object model (*model region*).

An extracted feature f_i is considered to correspond to a reference region $\tilde{f}_j$, if the *overlap error*, defined as the error in the image area covered by the respective regions , is sufficiently small:

$$1 - \frac{R_{f_i} \cap R_{H^T \tilde{f}_j H}}{R_{f_i} \cup R_{H^T \tilde{f}_j H}} < \epsilon, \tag{6.1}$$

where R_{f_i} represents the image region of f_i and H is the homography relating the two features. Some examples of the overlap errors are displayed in Fig. 6.6(b).

For each model region a *view set* is the set of its appearances in the environment maps from all possible viewpoints (see Fig. 6.3(b)). Depending on the 3D structure of the target object a model region may be clearly visible only from certain viewpoints in

[1] Due to complexity of the target object and the sampling rate this can be a time consuming procedure. However, this does not affect the computational cost of the system at run-time since this can be done off-line.

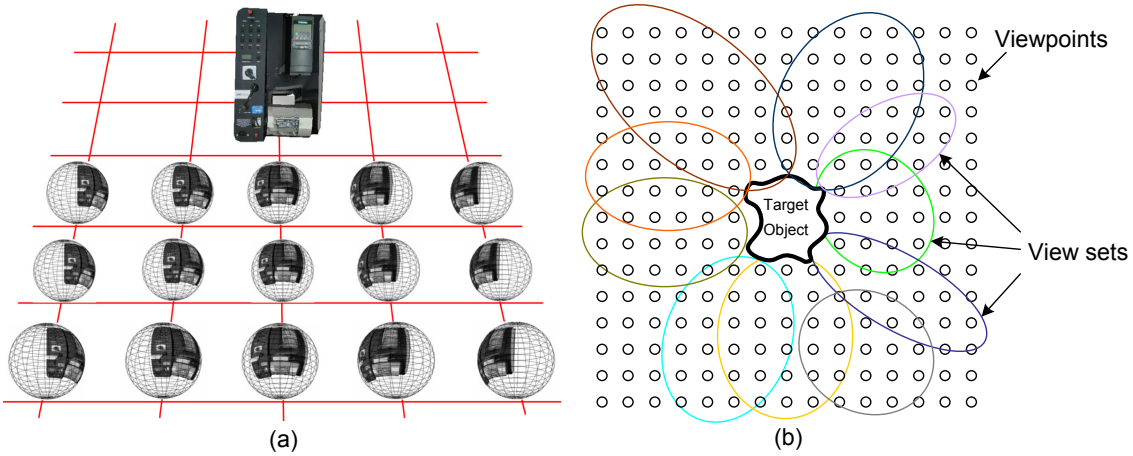


Figure 6.3: (a) A subset of the environment maps surrounding the object of interest. (b) A 2D illustration of the 3D clusters of the view sets surrounding the target object.

the scene. We create for each model feature a *similarity map* by comparing it with the corresponding extracted features. As a similarity measure we use the Mahalanobis distance between the respective SIFT descriptors. For each model region the respective similarity map represents its visibility distribution. This analysis can also be used to remove the repetitive features visible from the same viewpoints in order to keep the more distinctive features for matching.

Figure 6.7 shows some results of a simulated scene with a virtual box. The faces of the box are rendered with the texture obtained from a real tea box. Based on the similarity maps of each model region we cluster groups of viewpoints together using the mean-shift algorithm [24]. The clustered viewpoints for a model region m_j are $W(m_j) = \{v_{j,k} \in \Re^3 | 0 < k \leq N_j\}$, where $v_{j,k}$ is a viewpoint of that region.

Figure 6.8 shows some results of a simulated scene including a box and two cylinders. Over sixty patches are extracted automatically from each face of the box (see Fig. 6.8(b)-(c)). The space surrounding the box was coarsely sampled and a set of 15×15 environment maps were generated. Figure 6.9(a)-(d) show top down views of a subset of the similarity maps of four patches selected from each side of the box. Note how the presence of an occluding object (cylinders) is reflected in the similarity maps. The respective view sets determined by mean shift clustering (see B) are shown in Fig. 6.9(e)-(h).

6.2.2 Learning the statistical representation

This section describes a method to incorporate multiple view descriptors of each view set into our statistical model. We use the PCA-SIFT descriptor [57] for a more compact representation (e.g. first 32 components). To minimize the impact of variations of illumination, especially between the real and synthesized images, the descriptor vectors are normalized to unit magnitude. The image gradient vectors $g_{i,j}$ are projected into the feature space to a feature vector $e_{i,j}$.

We suppose that the distribution of the gradient vectors is Gaussian for the carefully selected features as described in the previous section. For each region we take k samples

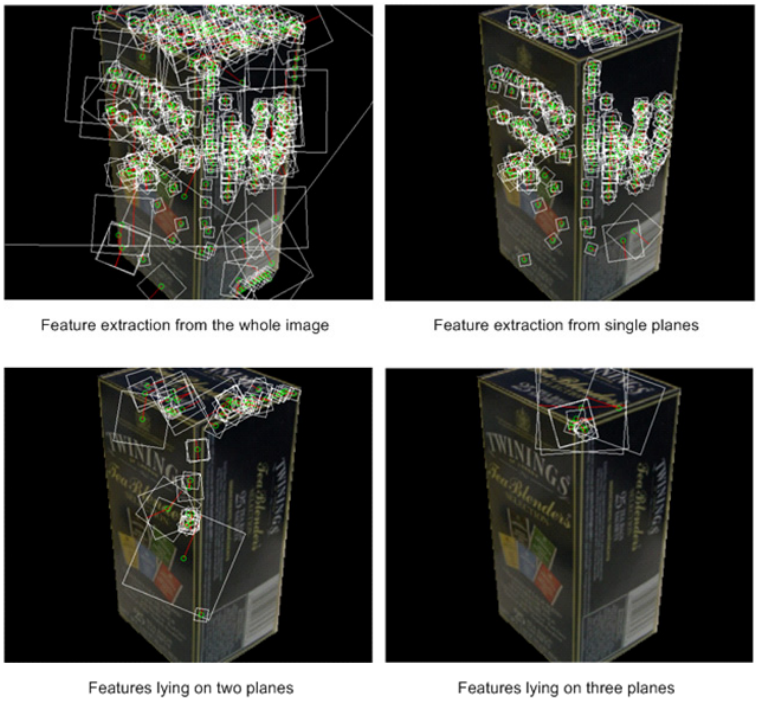


Figure 6.4: Feature selection based on the 3D structure of the target object.

from the respective environment maps so that the distribution of their feature vectors $e_{i,j}$ for $0 < j \leq K$ in the feature space is Gaussian. To ensure the Gaussian distribution of the gradient vectors for each view set we apply the χ^2 test for a maximal number of samples. If the χ^2 test fails after a certain number of samplings for a region, the region will be considered as not reliable enough and will be excluded. For each input view set V_i we then learn the covariance matrix Σ_i and the mean μ_i of the distribution.

6.2.3 Matching as a classification problem

Matching is the task to find groups of corresponding pairs between the regions extracted from the model and test image, that are consistent with both appearance and geometric constraints. The matching problem can be formulated as a classification problem [70]. Our goal is to construct a classifier so that the misclassification rate is low. From the test image, the features are extracted in the same manner as in the learning stage and their gradient image vectors are computed. The descriptors are then projected into feature space using PCA (bold dots in Fig. 6.2). We use the Bayesian classifier to decide whether a test descriptor belongs to a view set class or not. Let $C = \{C_1, ..., C_N\}$ be the set of all classes representing the view sets and let F denote the set of 2D-features $F = \{f_1, ..., f_K\}$ extracted from the test image. Using the Bayesian rule the *a posteriori probability* $P(C_i|f_j)$ for a test feature f_j that it belongs to the class C_i is calculated as

$$P(C_i|f_j) = \frac{p(f_j|C_i)P(C_i)}{\sum_{k=1}^{N} p(f_j|C_k)P(C_k)}. \tag{6.2}$$

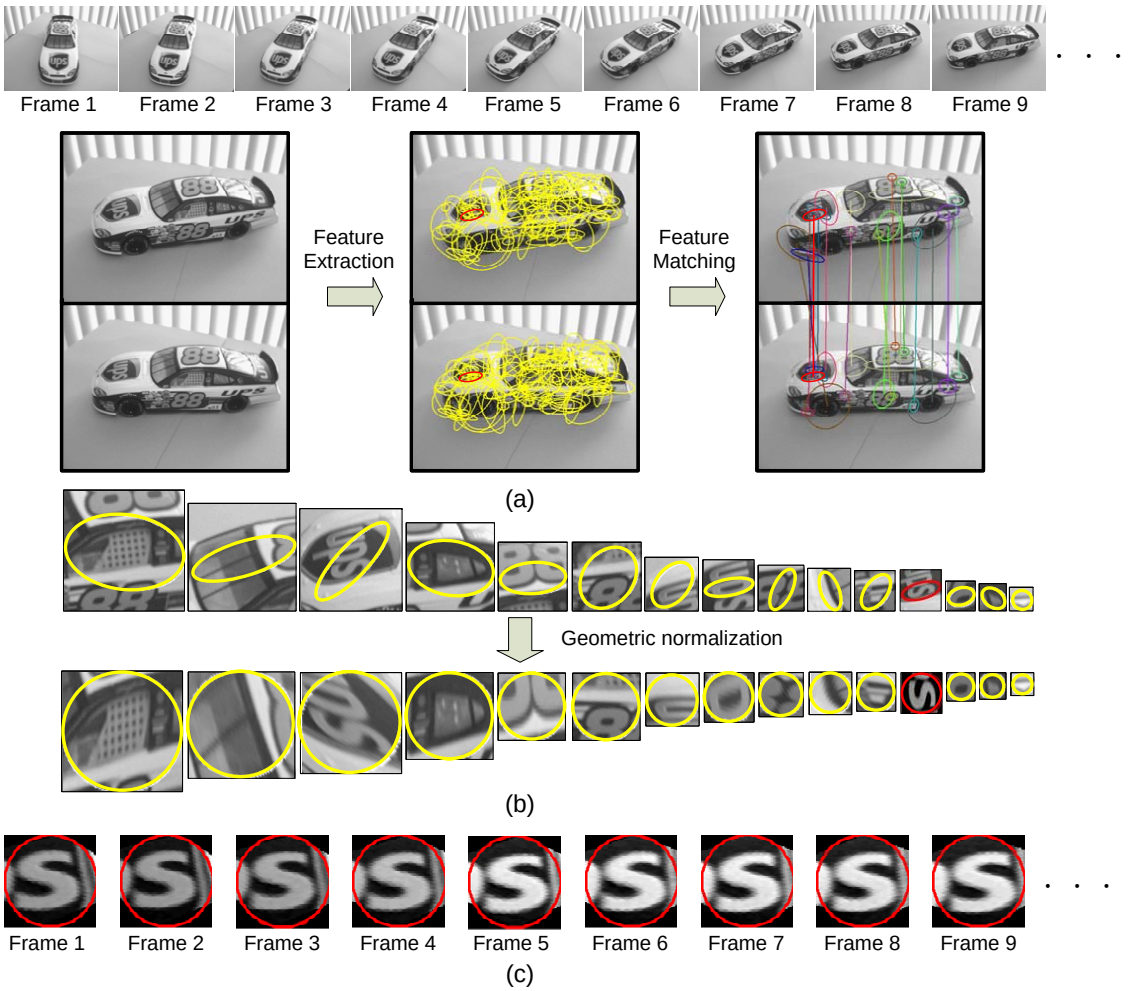


Figure 6.5: Feature descriptor evaluation: (a) The turntable sequence: the feature extraction and matching procedure. (b) Examples of geometric normalization of feature regions. (c) Normlized feature regions of a model feature in the first frames of the turntable sequence.

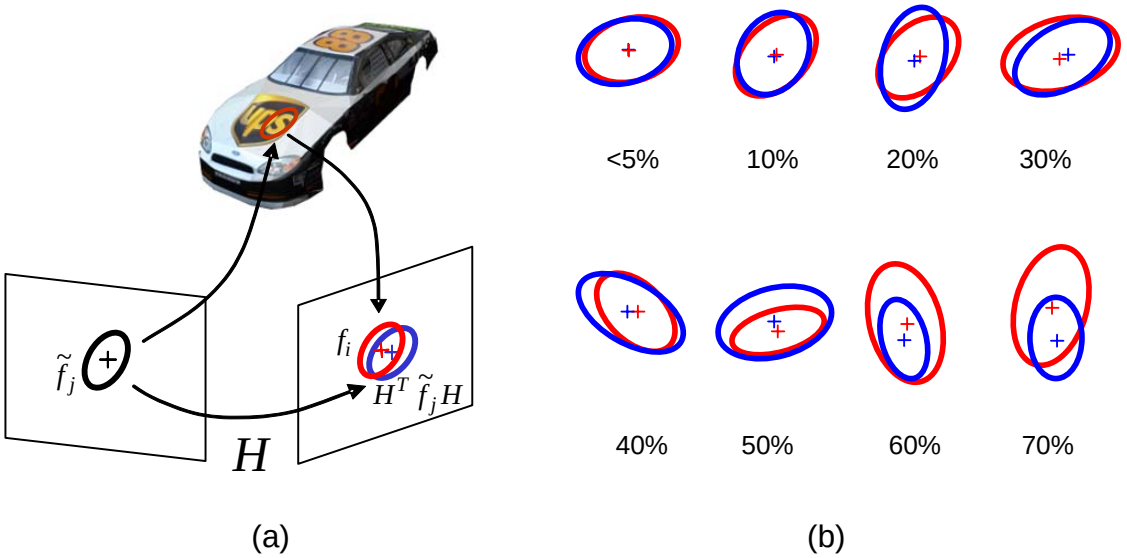


Figure 6.6: Feature repeatability evaluation: (a) The *overlap error* is defined as the relative amount of overlap in the image area between the detected regions f_i in an image and the respective reference regions $\tilde{f}_j$ projected onto that image. (b) Examples of the overlap errors.

We compute for each test descriptor the a posteriori probability of all classes and select candidate matches using thresholding. Let $m(f_j)$ be the respective set of most probable potential matches $m(f_j) = \{C_i | P(C_i|f_j) \geq T\}$. The purpose of this threshold is only to accelerate the run-time matching and not to consider matching candidates with low probability. However this threshold is not crucial for the results of pose estimation.

6.2.4 Pose estimation using geometric inference

This section describes a method using geometric consistency to constrain the search space for finding candidate matches. For the pose estimation a set of $N \geq 3$ matches are required. In an iterative manner we choose the first match $f'_1 \leftrightarrow C'_1$ as the pair of correspondences with the highest confidence:

$$\underset{\substack{f_k \in F \\ C_l \in C}}{argmax} \; P(C_l|f_k).$$

We define V_{C_l} as the set of all classes of regions which should also be visible from the viewpoints where C_l is visible

$$V_{C_l} = \{C_k \in C || W_k \cap W_l| \neq 0\},$$

where W_j is the set of 3D-coordinates of the clustered viewpoints $\{v_{j,k} | 0 < k \leq N_j\}$ for which the respective model region is visible (see building environment maps, Section 3.1).

Assuming the first candidate match is correct, the second match $f'_2 \leftrightarrow C'_2$ is chosen only from the respective set of visible regions. Therefore after each match selection the

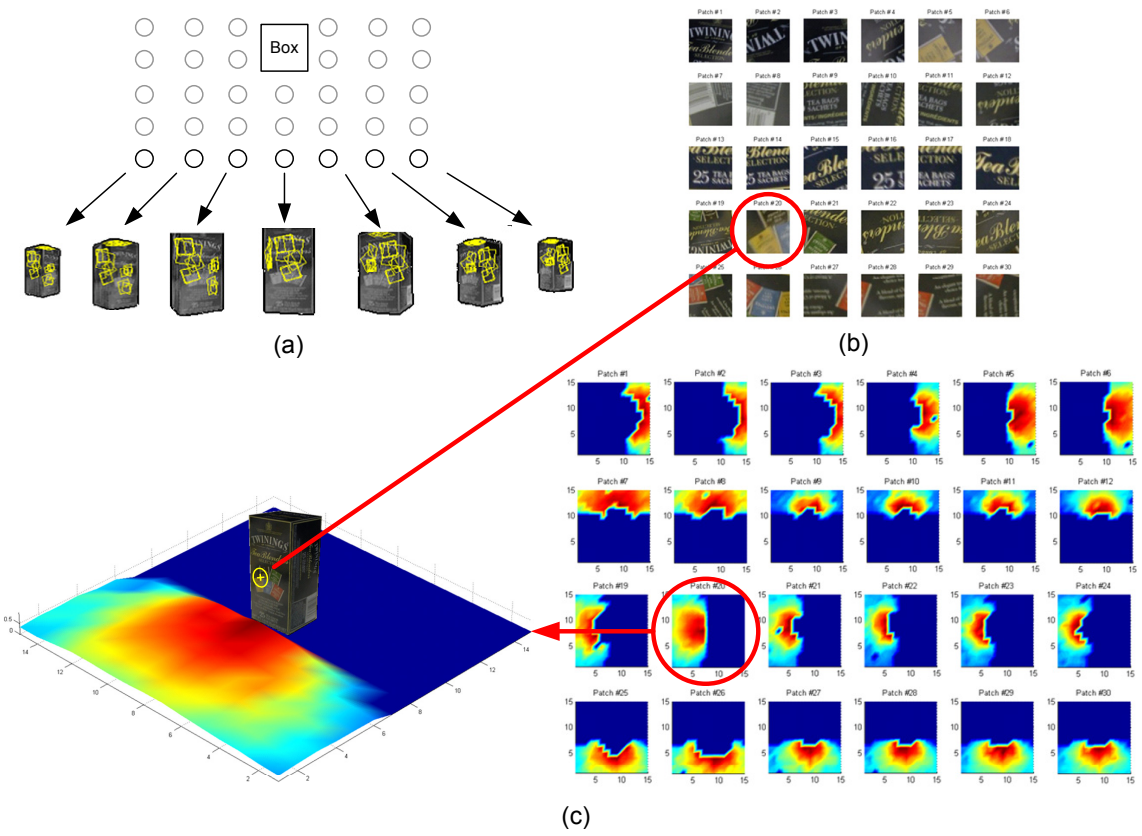


Figure 6.7: Experiments with simulated data. (a) A subset of of the environment maps surrounding the target object (box). (b) The set of extracted rectified features. (c) Top-down view of the respective similarity maps.

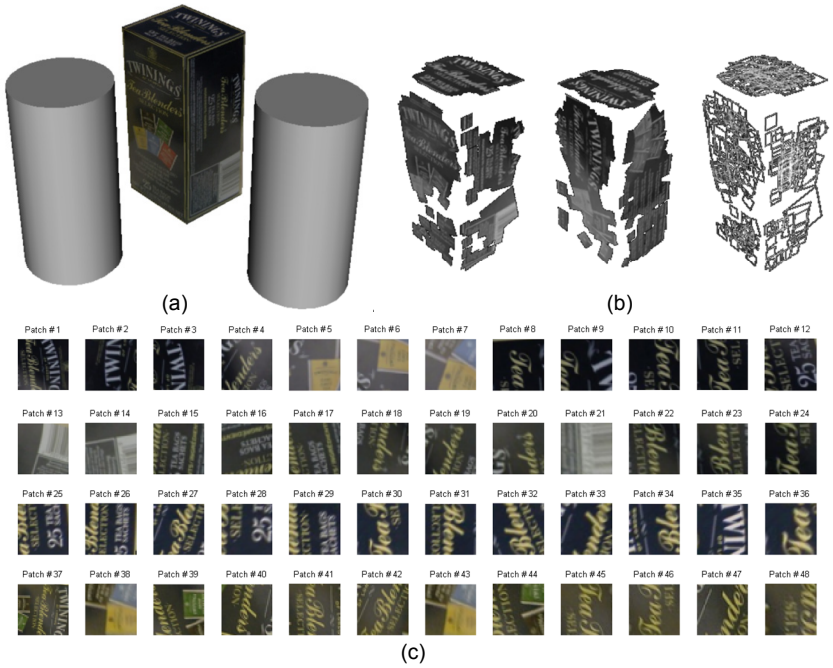


Figure 6.8: Experiments with simulated data. a) The virtual model of the object. (b) The extracted features on the model. (c) The set of rectified planar patches.

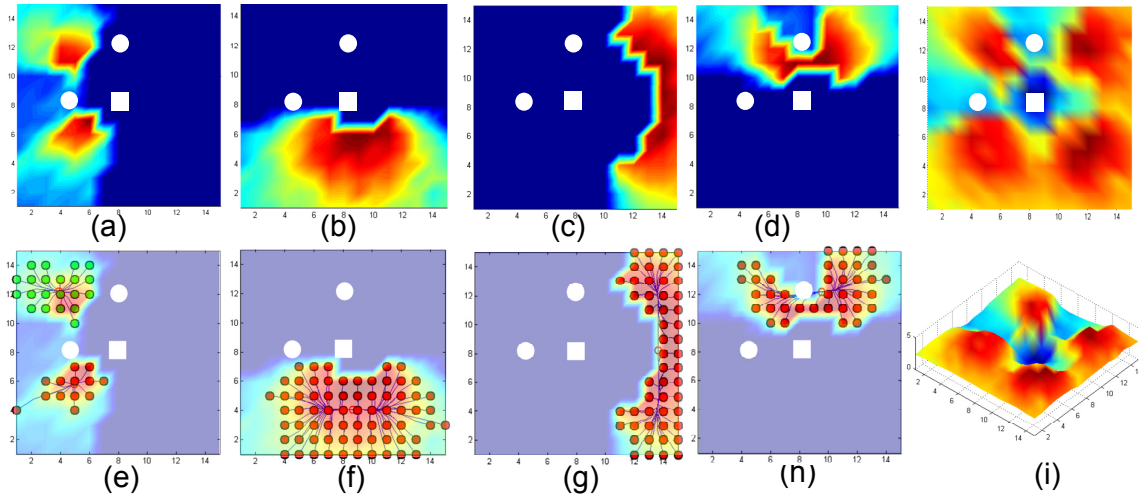


Figure 6.9: Experiments with simulated data. (a)-(d) Top-down view of a subset of the similarity maps. (e)-(h) The clustered view sets using mean-shift algorithm. (i) The average of all similarity maps.

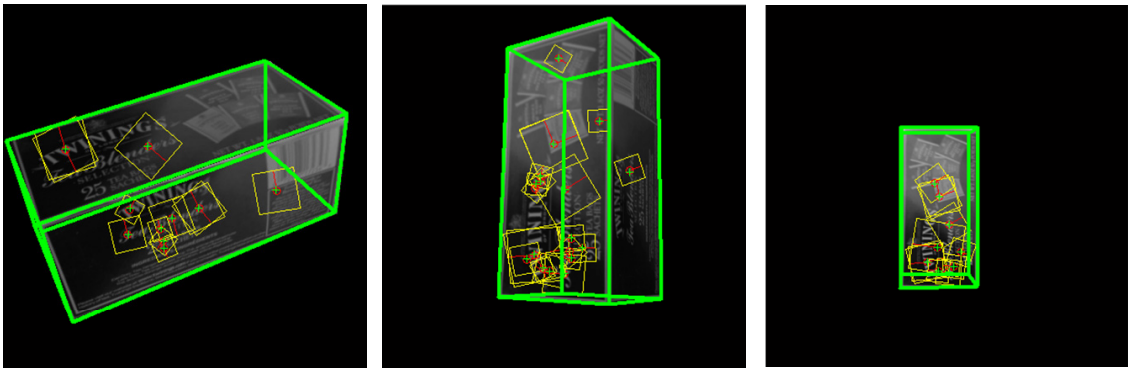


Figure 6.10: Experiments with simulated data. (a)-(c) Some virtual test images and pose estimation results.

search area is constrained to visibility of those regions based on previous patches. In general the k^{th} candidate match $f'_k \leftrightarrow C'_k, 1 < k \leq N$ is selected in a deterministic manner

$$(f'_k, C'_k) = argmax_{\substack{f_k \in F \setminus \{f_1, ..., f_{k-1}\} \\ C_k \in \bigcap_{l=1}^{k-1} V_{C'_l}}} P(C_k|f_k).$$

The termination criteria is defined based on the back-projected overlap error (see Section 3.1) in the test image. This algorithm can be implemented in different ways. One way is a recursive implementation with an interpretation tree where the nodes are visited in the depth-first manner. The depth is the number of required matches N for the pose estimation method. This algorithm has a lower complexity as the results will show, than the plain version of RANSAC or the "exhaustive" version where all pairs of candidate matches are examined.

6.3 Experimental Results

The proposed method has been tested in a series of experiments using virtual and real objects. Figure 6.10 shows some pose estimation results of the simulated scene using virtual views as test images.

The off-line learning process for real experiments uses ImageModeler from RealViz^{TM} [117] to obtain a 3D model.[2] Our experimental setup consists of a target object and a commonly available FireWire camera (Fire-I). The camera is internally calibrated and lens distortions are corrected using the Tsai's algorithm [139].

We conducted a set of experiments to analyze the functional and the timing performance of our approach. The results were compared against a conventional approach based solely on 2D key frames. Our approach requires an input consisting of a set of images (or key frames) of the target object. One target object is shown in 6.11(a). The key frames were calibrated. We used a calibration object (a known set of markers) for automatically calibrating the views. These markers were used to compute the ground truth for evaluating

[2]The accuracy requirements depend on the underlying pose estimation algorithms, the object size and the imaging device.

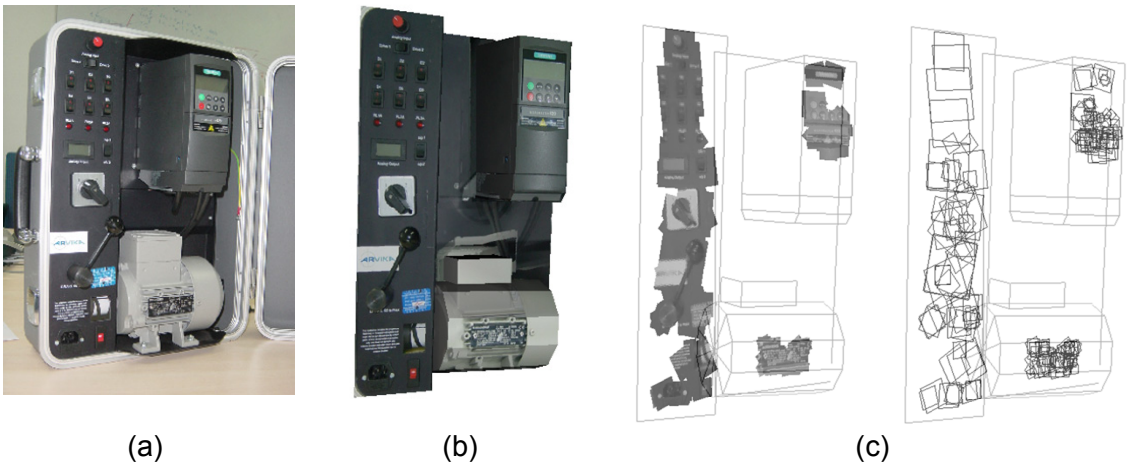


Figure 6.11: Experiments with real data. a) The target object. (b) The reconstructed 3D model using two images. (c) The set of most visible patches extracted on the model based on the statistical analysis using the similarity maps.

the matching results on test frames as well. Fig. 6.11(b) and (c) show the reconstructed 3D model and the selected features on the model.

In the first experiment, we analyzed the functional performance against view point variations for the same scene but under uncontrolled lighting. The images were taken by a moving camera around the object. For the sake of clarity of presentation, we show a subset of 19 test images from this sequence. All those images were calibrated as explained above. Fig. 6.12 shows some metrics we used to compare these results. One measure of performance is the final size of the representation (number of features in the database) used for both methods indicated by the two straight lines. With increasing number of key frames the size of the database in the conventional case would increase linearly with the number of key frames. In contrast, our method keeps fewer features in the 2D-3D database after careful implicit analysis of their planarity, visibility and detection repeatability. The database size in our method is proportional to the scene complexity not the number of available key frames. This is an important property for the scalability of the system for more complex objects.

Fig. 6.12 also shows the number of extracted features and the number of correct matches found by both methods for each of the 19 test images. It should be noted that, near the two key frames our method obtains less correct matches compared to the conventional method. This is due to the fact that our representation generalizes the extracted features whereas the conventional methods keeps them as they are. The generalization has the cost of missing some of the features in the images closer to the key frames. On the other hand, the generalization helps to correctly match more features in disparate test views.

Complexity and performance of robust pose estimation methods like RANSAC are dependent not on the number of correct matches but the ratio between correct and false matches. Fig. 6.13(a) shows the percentage of correct matches vs the viewing angle for the proposed method and the conventional approach.

Although near the key frames our method obtains fewer matches, it has a higher per-

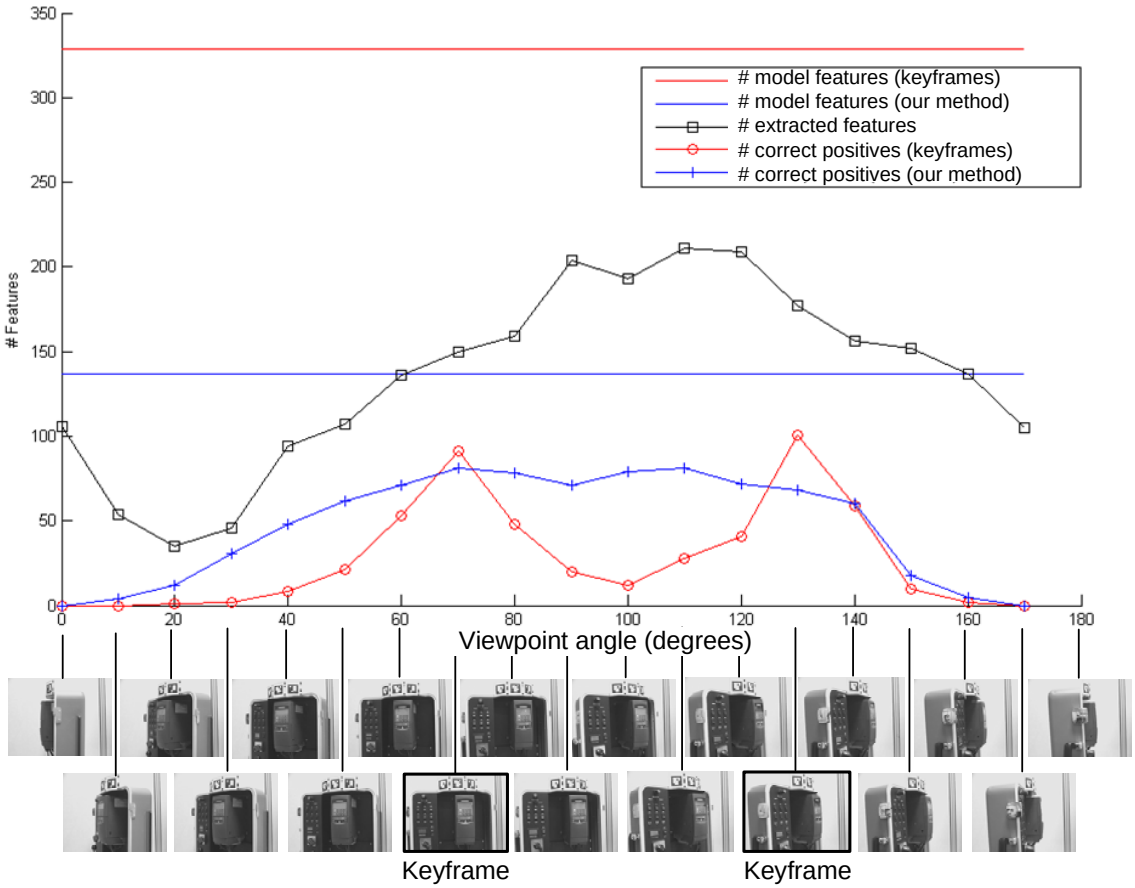


Figure 6.12: Experiments with real data. Metrics used to compare the detection results (see text).

centage of correct positives. As a result of this and the visibility constraints used our method needs only a few RANSAC iterations for pose estimation. This brings us to the timing performance of the matching methods. We use a more complex matching method than the conventional one. Therefore, each individual match costs more. However, with increasing complexity of the target object with respect to self-occlusions our representation becomes more efficient. Fig. 6.13(b) shows the respective maximal number of iterations needed (logarithmic scale) for RANSAC based pose estimation with a confidence probability of 95%. Fig. 6.14 shows a visualization of the pose estimation results. We obtain up to five folds speed-up compared to the exhaustive RANSAC method. Our non-optimized implementation needs about 0.3 to 0.6 second compared to 2.5 seconds for the conventional approach. In Figures 6.15 and 6.16 (a)-(h) more results are shown for experiments using test images with occlusions, cluttered background and illumination changes. The detection results are quite robust and the estimated pose is accurate enough to initialize our real-time 3D tracker [43]. Fig. 6.17 and 6.18 show the results of two other experiments in outdoor environments. We used each time two images to build a coarse 3D model and applied our method to several test images.

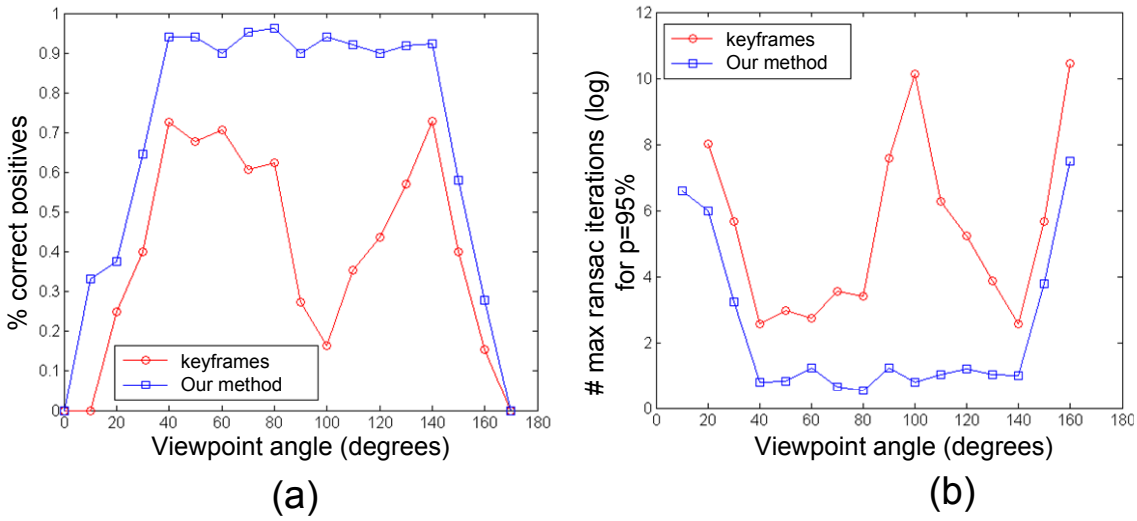


Figure 6.13: Experiments with real data. (a)-(b) Performance evaluation (see text).

The performance of the matching part of our system was evaluated by processing all pairs of object model and test images, and counting the number of established matches. Fig. 6.19 shows the ROC curve that depicts the detection rate vs false-positive rate, while varying the detection threshold T. Compared to the keyframe-based approach the proposed approach performs very well and achieves 97% detection with 5% false-positives.

6.4 Performance evaluation of covariant features

In the previous section we compared the results of the proposed method against a conventional approach based solely on 2D key frames. We also conducted a set of experiments to evaluate the functional and the timing performance of the proposed framework using different feature detectors. This section presents the results of some of the state-of-the-art feature detectors.

Five types of feature detectors are used:

- *Harris-Affine detector*: based on affine normalization around Harris points [93, 91, 124],

Figure 6.14: Experiments with real data. Visualization of the pose estimation results.

Figure 6.15: Pose estimation results on test images.

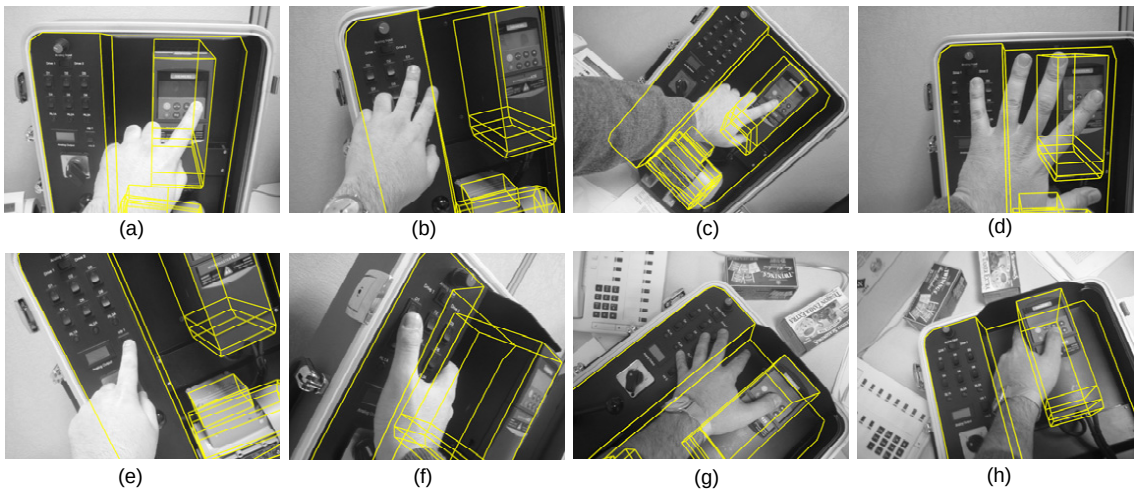


Figure 6.16: Experiment 1: Control Box. Pose estimation results on test images.

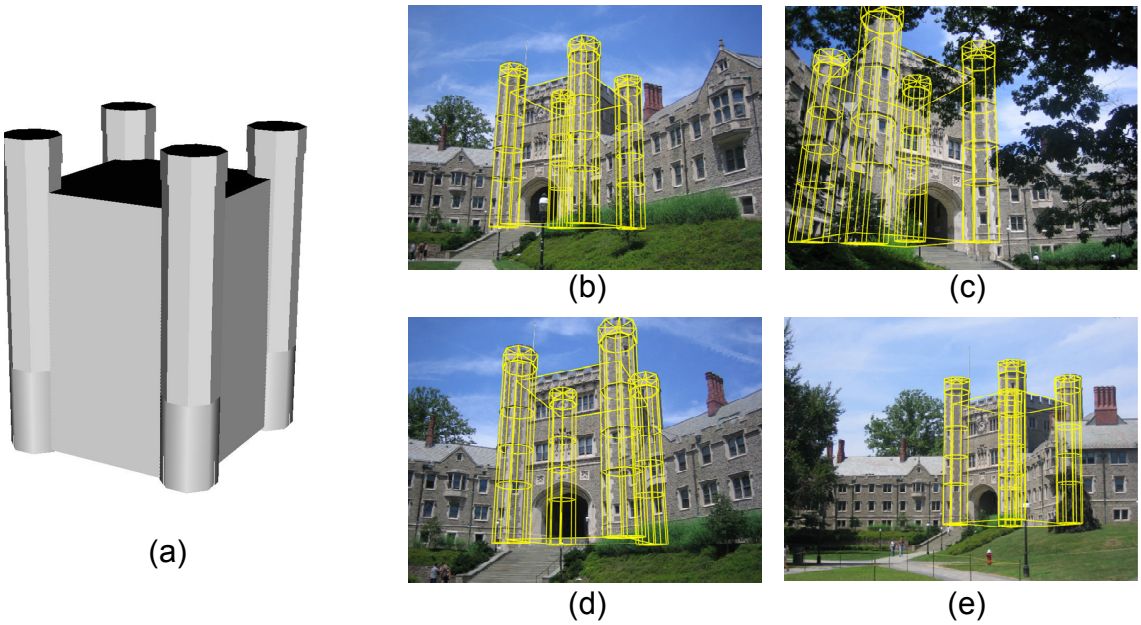


Figure 6.17: Experiment 2: Blair Tower. Pose estimation results on test images.

- *Hessian-Affine detector*: based on affine normalization around Hessian points [93, 91],
- *Intensity extrema-based region detector* (IBR) [144, 143],
- *Edge-based region detector* (EBR) [144, 143],
- *Maximally stable extremal region detector* (MSER) [85].

Figure 6.20(a)-(d) shows the feature regions for the detectors on a test image.

The performance analysis and comparison of these detectors was done under the following conditions.

- Learning for object representation: Our approach builds a representation of the object using both the 3D and appearance models as explained in the previous sections. The 3D model and the texture were then used to create synthesized images from all other possible views. The most stable features and the respective view sets were determined with different detectors and learned using the method described in Section 3.1. For this purpose we kept the values in the first 32 dimensions given by the PCA.
- Feature extraction: Affine covariant features are extracted as done in learning.
- Feature matching: Matching is formulated as a classification problem and solved using the Bayesian classifiers as described in Section 6.2.3
- Pose estimation: The pose of the target object was estimated using the geometric inference as described in Section 6.2.4.

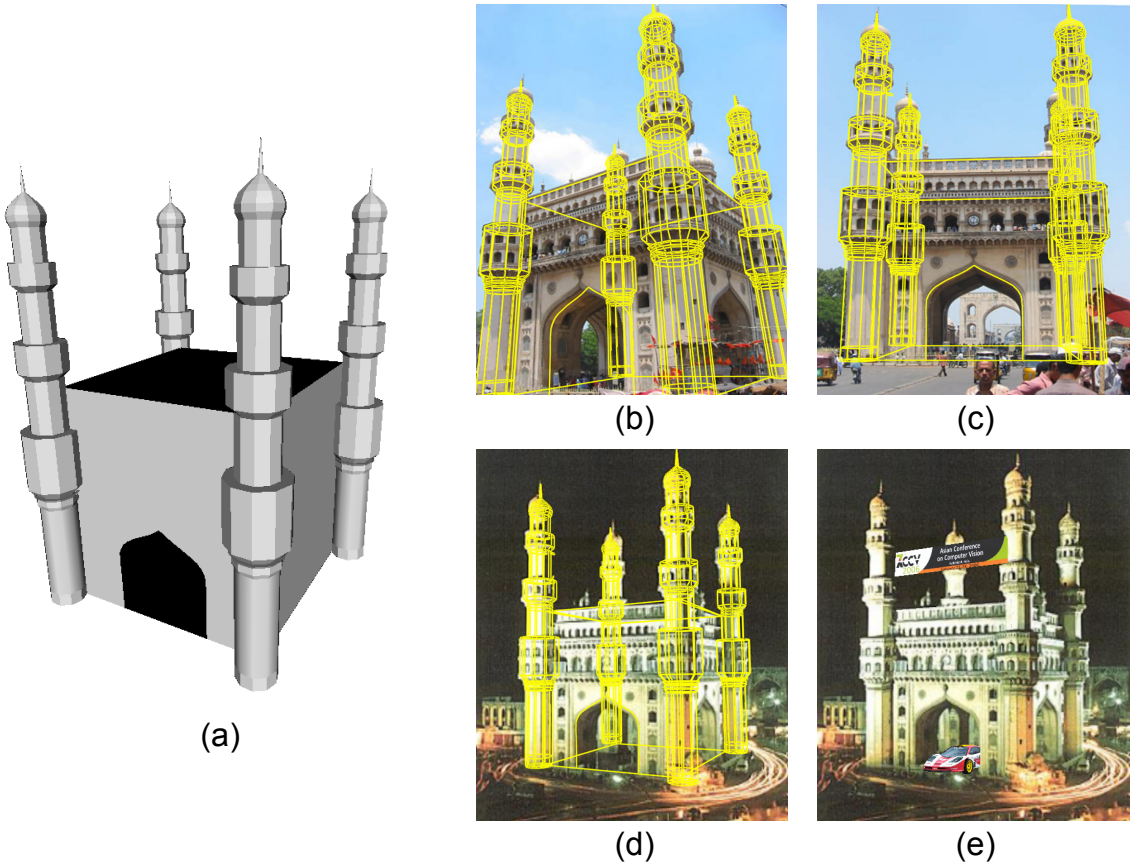


Figure 6.18: Experiment 3: Char Minar. (a) 3D model. (b)-(d) Pose estimation results on test images, and with virtual objects (e).

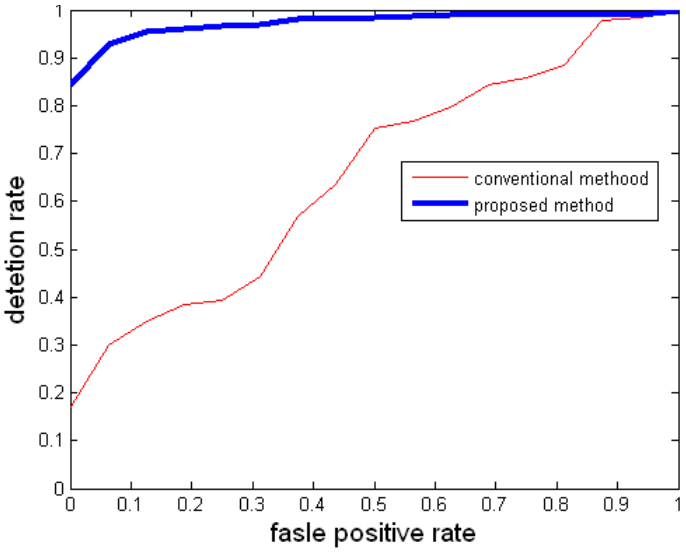


Figure 6.19: Performance evaluation: ROC plot.

Figure 6.21 shows some metrics we used to compare the results of different detectors. One measure of performance is the final size of the representation (number of model features in the database) indicated by the straight lines. The respective features are taken into the 2D-3D database based on careful analysis of their planarity, visibility and detection repeatability (see section 3). Fig. 6.21 also shows the number of extracted features from the test images (*test features*) for each of the 19 test views.

Figure 6.22(a) and 6.22(b) depict the number of correspondences and the number of correct positive respectively. The ranking of the detectors in Figure 6.22(a) based on the number of matched features do not change with respect to the number of corresponding regions on figure 6.22(b). These results are obtained by the known ground truth and the overlap error test. In this experiment Hessian-Affine and Harris-Affine provide more corresponding regions than the other detectors.

Complexity and performance of the robust pose estimation methods like RANSAC is dependent not on the number of correct matches but the ratio between correct and false matches. Fig. 6.22(c) shows the percentage of correct matches vs the viewing angle for the different detectors used. The Hessian-Affine detector almost outperforms the Harris-Affine and the same holds for MSER with respect to IBR and EBR. As a result of the visibility constraints used only a few RANSAC iterations for pose estimation are needed. This brings us to the timing performance of the matching system. Fig. 6.22(d) shows the respective maximal number of iterations needed (logarithmic scale) for RANSAC-based pose estimation with a confidence probability of 95%.

We obtain approximately up to five folds speed up compared to the exhaustive RANSAC method. Our non-optimized implementation needs about 0.3 to 0.8 second for the entire object detection and pose estimation, depending on the choice of the feature detector.

The estimated pose results for all views only differ slightly for different detectors if enough correct correspondences are available. Starting with an initial pose estimation with 10% error, our feature-based tracking system is able to converge. Therefore the proposed

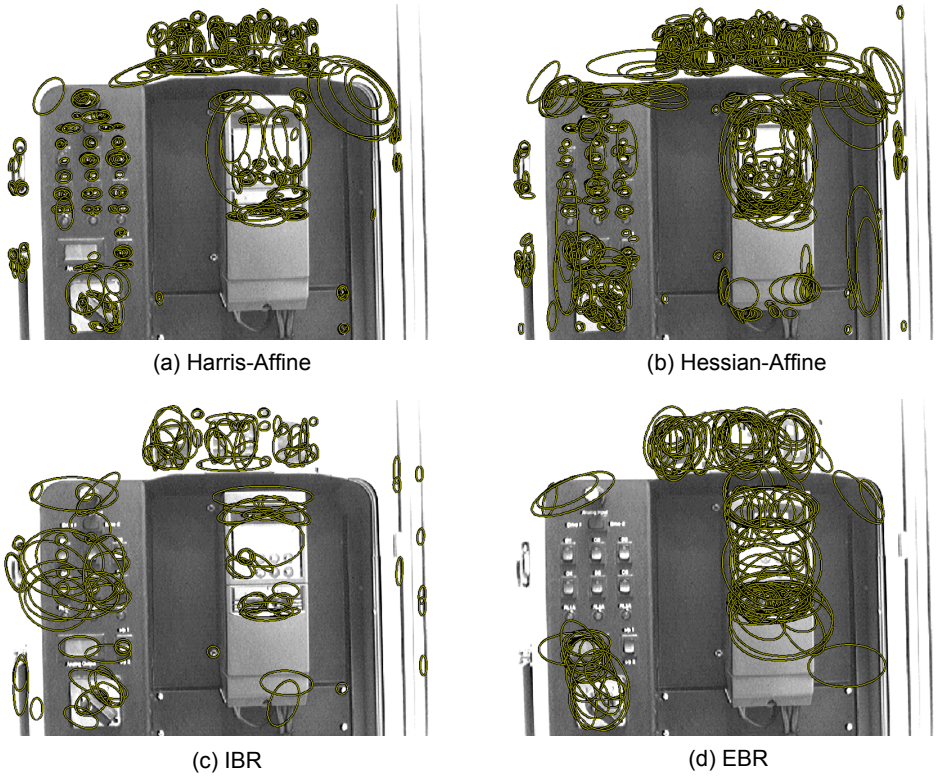


Figure 6.20: Affine covariant detectors used for experiments.

pose estimation algorithm gives higher priority to computational time requirement and robustness than accuracy requirement.

By looking at the Figures 6.21 and 6.22(a)-(d) it is obvious that for this particular object of interest the Hessian-Affine detector would be the best choice. However, our objective is that the system automatically selects the best feature detector (or a combination of detectors) for each scene and eventually for each set of viewing directions based on such an evaluation study performed as a part of the learning phase.

6.5 Discussion

This chapter addressed the problem of real-time object detection for pose estimation. The major contribution is the integration of the known 3D geometry of the target model during both matching and pose estimation steps. This is achieved by a statistical analysis of the appearances distribution of model patches in the viewing space. Instead of the local planarity assumption used in previous approaches, our proposed method is able to learn the visibility distribution of the variations in the local descriptors considering their known geometry.

Our approach tries to exploit all the information available in the scene in an off-line process allowing on-line detection and pose estimation. We performed an evaluation study comparing the performance of main covariant feature detectors proposed in the literature in conjunction with our initialization strategy. Our method allows us to determine the

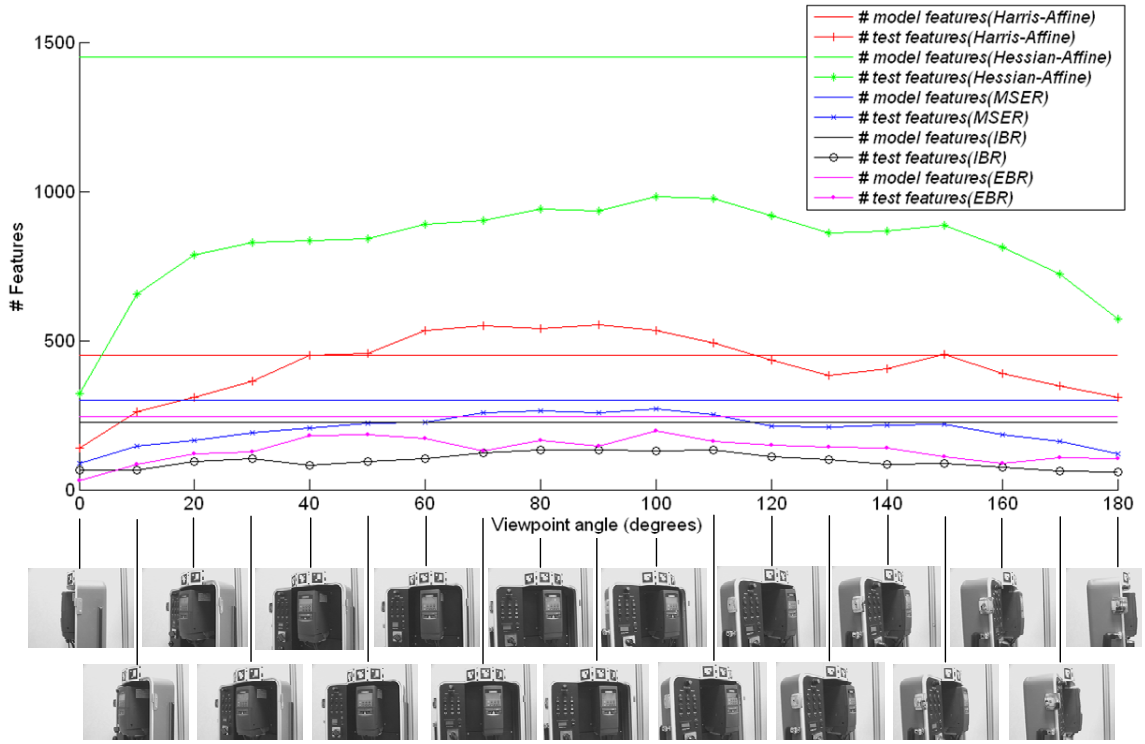


Figure 6.21: Experiments with real data. Metrics used to compare the results.

appropriate feature detector and the set of respective features with the best functional and time performance for a given type of scene or target object. This is necessary for automatic initialization of feature-based tracking solutions in industrial AR. The models of the industrial objects are usually available.

This performance evaluation is not aimed at defining the best available solution in term of feature detection, but rather at presenting an integrated part of our off-line procedure for selecting appropriate feature detector for each region of interest in the scene.

Our results show that the most visible and reliable features can be extracted from complex object in an off-line process. The 3D model of a target of interest can be furthermore exploited for faster object detection. As a first step toward this goal we used environment maps generated from all different viewpoints in the scene. The amount of data storage and the time required in the learning stage is not an issue in our industrial application. However, the number of the required environment maps can be reduced efficiently based on the 3D object structure and distance to the object.

The extraction of affine covariant features is currently the most computational cost intensive component in our algorithm. Using the 3D geometry of the target object one could reduce this complexity and increase the efficiency of the feature extraction algorithm.

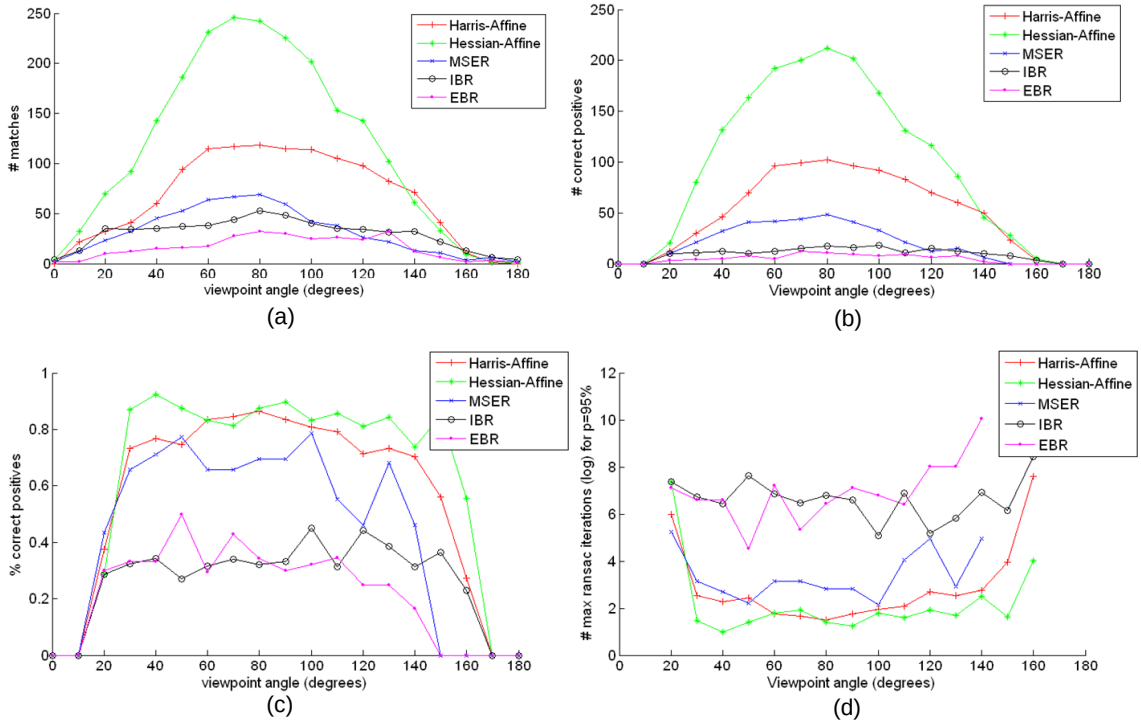


Figure 6.22: Experiments with real data. (a)-(d) See the text.

Chapter 7

Results and Applications

In this chapter we discuss our experimental results and potential AR applications of the detection system. We also describe the tools that have been developed for the offline training process. Our detection system has been integrated into an AR tracking framework for the initialization of a marker-less real-time tracking system. We introduce a tracking management framework designed for single as well as multiple object tracking. The chapter concludes with discussion of some industrial AR applications.

7.1 Test objects

The object detection system has been tested on a set of different objects. The objects have been chosen of different sizes with different textures and structures to demonstrate the scalability and reliability of the system. Figure 7.1 depicts four objects with overlaid 3D models whose pose has been estimated using our approach described in the previous chapter. These objects are used throughout this chapter to describe the different steps of the training phase and to discuss the detection and tracking results.

The 3D model of each object was built by using the RealViz Image Modeler [117] and some high resolution still images taken with a digital camera (Canon EOS20D). Figure 7.2 shows the modeling process of the toy car using eight images. To reconstruct a 3D model first correspondences between distinctive feature points were selected in an interactive

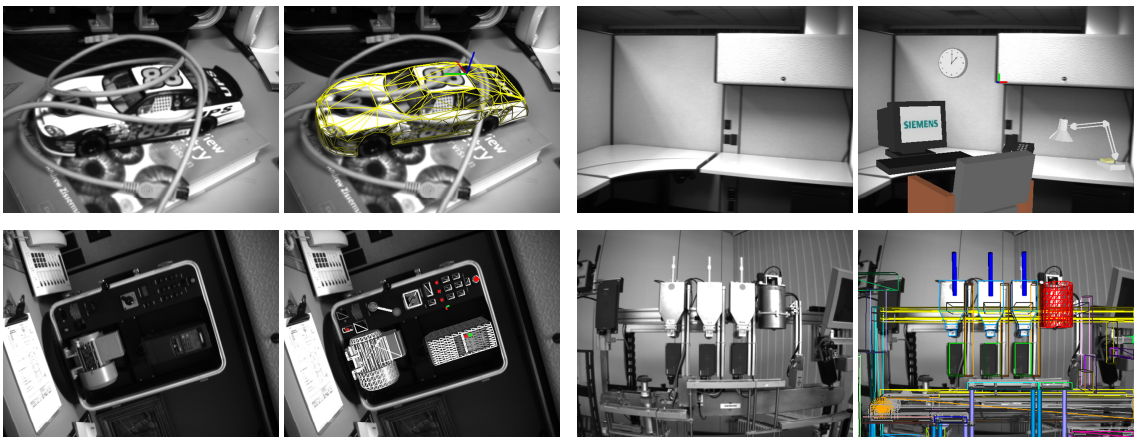


Figure 7.1: Set of objects used in our detection and pose estimation with overlaid 3D models: A toy car, a control box, an office cubicle and an industrial coffee machine.

manner and then their 3D coordinates were estimated using triangulation followed by bundle adjustment. Finally, a triangulated mesh was created and the texture from the images was applied on it (see Figure 7.2).

Figure 7.3 shows some results of the 3D object detection applied to the toy car. Despite strong viewpoint and scale changes, severe occlusions and large amounts of background clutter, the system is in most of the cases able to recover the pose correctly. Figure 7.4 more results and three images where detection fails. Very little is visible of the toy car in the first of them with large clutter where the system fails to detect it. Also in the case of occlusions by similar textures in the second image the system is unable to classify the patches correctly and fails in the pose estimation process. And since non of the affine covariant feature extractors we use are robust against image deformations the system fails in detecting distorted texture regions. These illustrate the limitations of our feature based detection system. Next section describes the offline learning phase and the implementation details in more detail.

7.2 Offline process and implementation details

The learning process requires a calibrated image sequence, both internally and externally. In this section we describe the calibration procedure of an image sequence and explain the learning stage in more detail.

7.2.1 Calibrating an image sequence

The camera is calibrated internally with an calibration grid consisting of a set of predefined markers (see section 2.2). Using the computed undistortion parameters the images of the entire sequence are then undistorted. Figure 7.5 shows the effect of undistortion applied on an image from a sequence. For the external calibration we have developed a program called *MotionTracker*. There are three different ways for calibration. First a set of markers can be placed on or around the object of interest. This requires an additional calibration of the markers. For the feature extraction and evaluation of the training process

Figure 7.2: 3D model reconstruction of the toy car using eight images.

these markers are then removed automatically since their coordinates in the images can be computed. Depending on the size and structure of the target object, markers can not always be placed anywhere or be large enough to be detected in the training image sequence without occluding important features of the object for detection later on. In this case, MoitionTacker provides two alternative ways with some interaction for calibration without using markers. First alternative is to select natural feature point correspondences from the image and the 3D model. This is demonstrated in figure 7.6(a)-(b). Thereby, a zoom window is displayed for better localization of the feature points. Having three or more 2D-3D correspondences the pose can be computed and the model is aligned on the image. Due to the small number of 2D-3D matches and the error in localizing points, the alignment is not perfect. Therefore a post-processing refinement process is applied based on using 3D-2D edges (see Fig. 7.6(c)-(d)). Second alternative way is to roughly align the model manually using the respective motion control buttons and apply the refinement procedure afterward. Section 4.5 described an ICP based refinement process that can be used for this purpose. However, we use a similar algorithm that is described in section 7.3. After the pose of the first frame is computed accurately, the target object can be tracked in the consecutive frames using the edge tracker. However, the alignment can be corrected anytime if tracking fails. The pose of each image is then stored separately and used for the training process described in the next section.

7.2.2 Training process for detection

Once a set of calibrated images is available, the learning process for the detection system can be started. Figure 7.7 shows an overview of this procedure. The set of calibrated

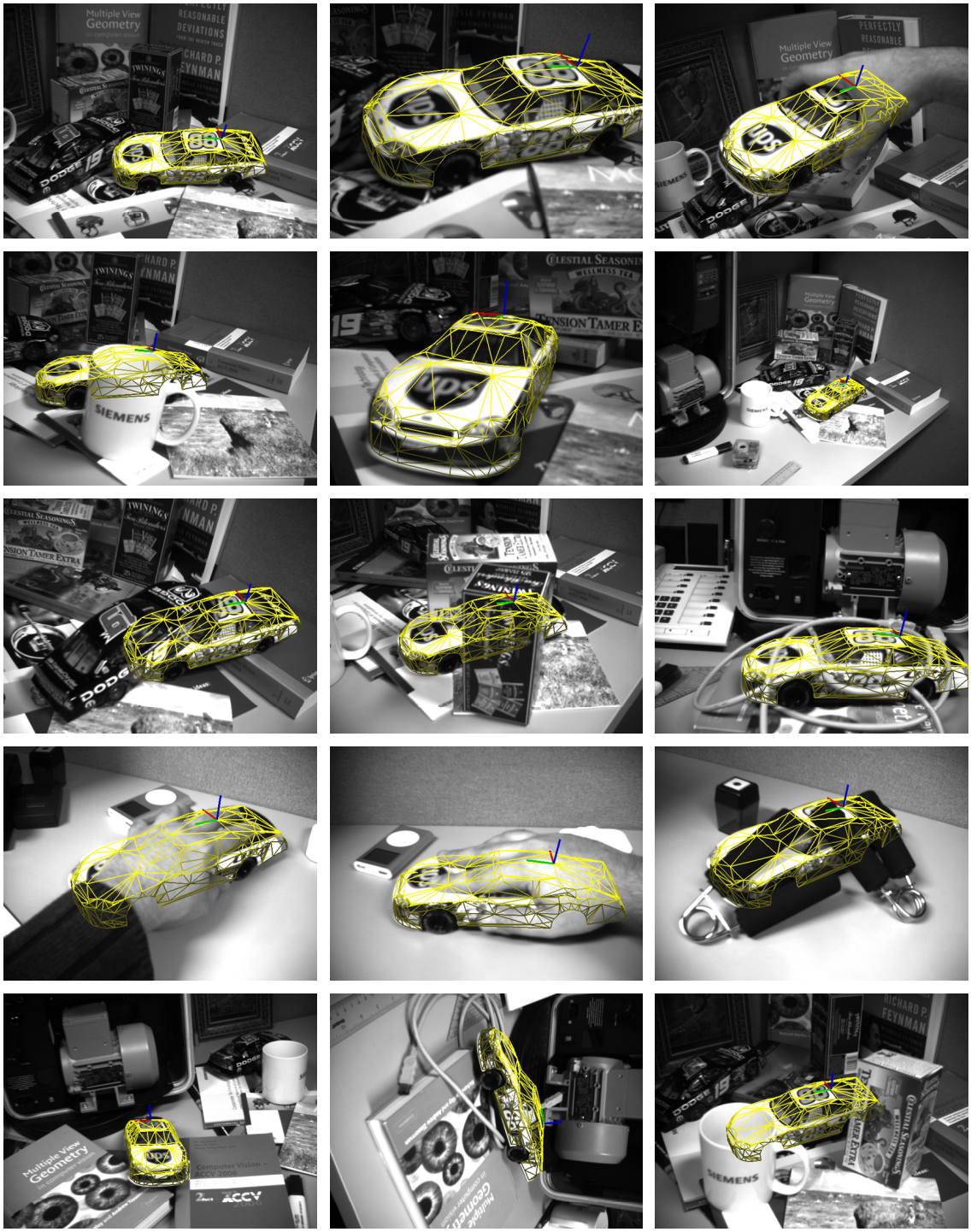


Figure 7.3: Detection results of the toy car in a sequence of images with strong viewpoint and scale changes, severe occlusions and large amounts of background clutter.

Figure 7.4: Detection results of the toy car. The last row show some cases where detection fails due to large occlusions combined with cluttered background or large distortions of the texture structure e.g. when seen from behind a convex glass.

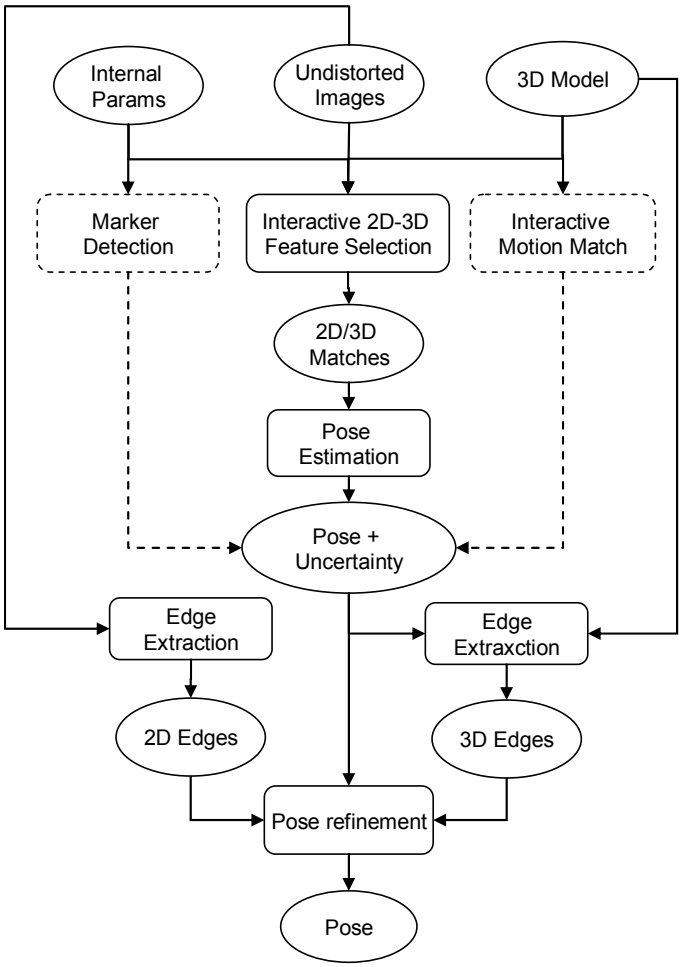
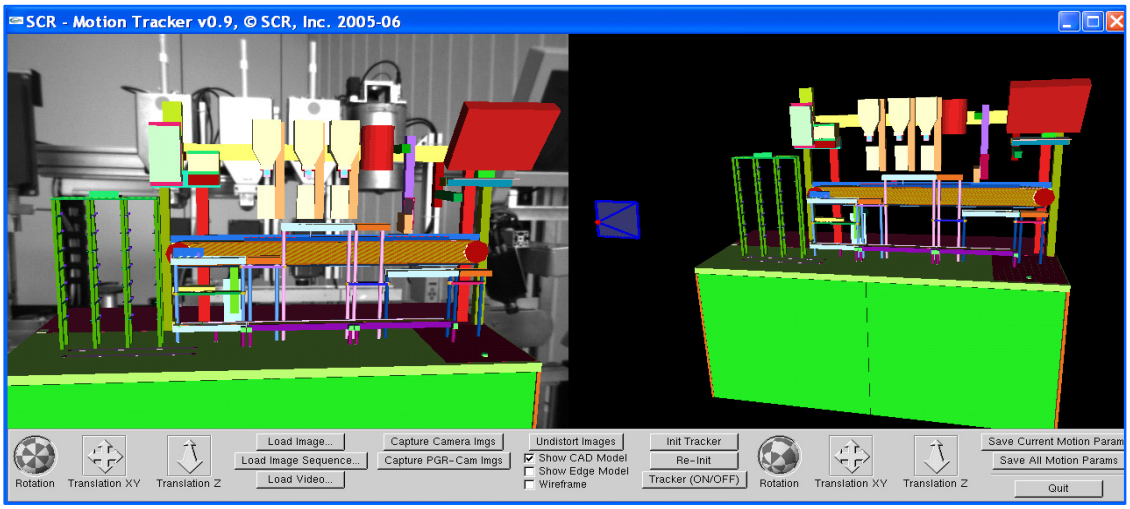
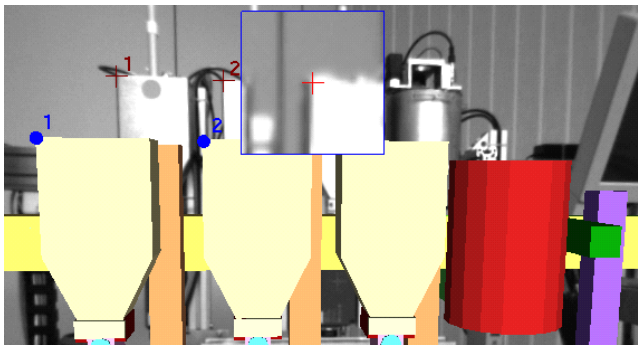


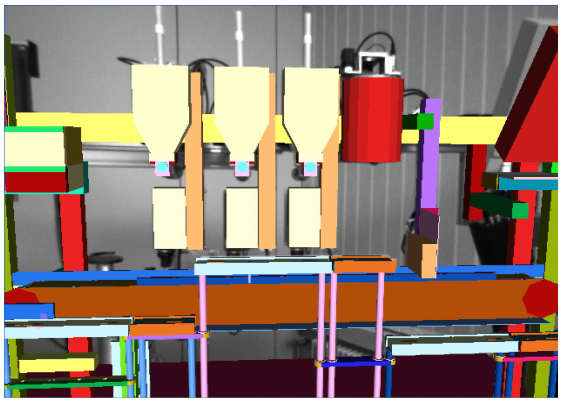
Figure 7.5: External calibration procedure for an image sequence.

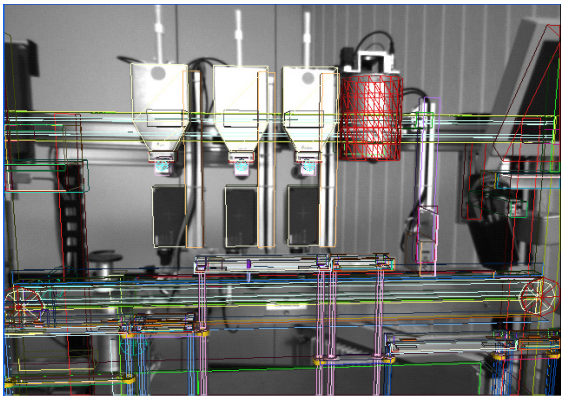


(a)



(b)





(c) (d)

Figure 7.6: Calibrating an image sequence. (a)-(d) See the text.

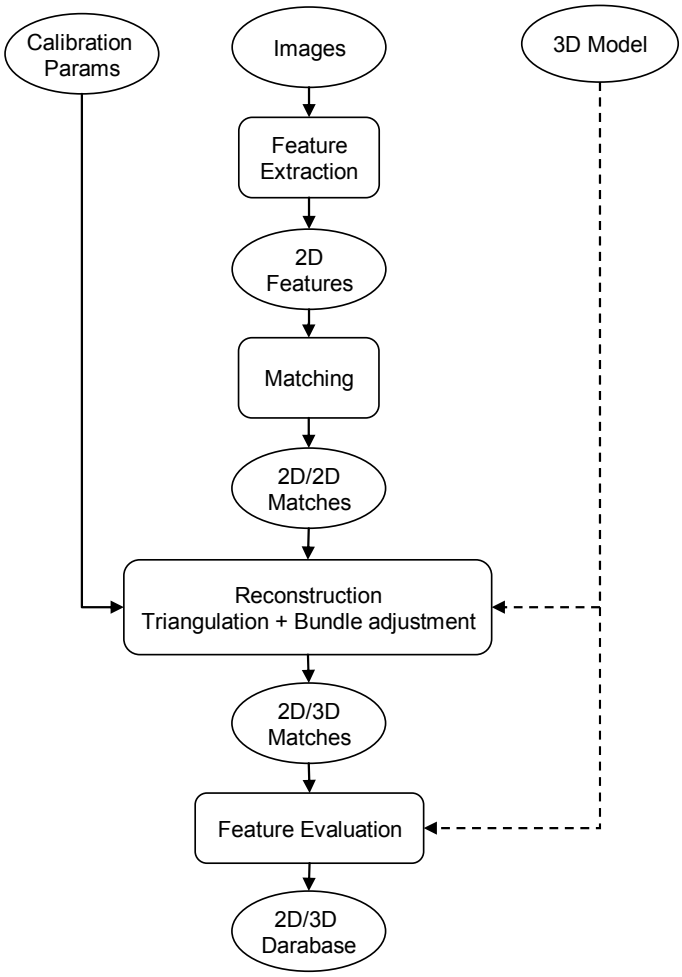


Figure 7.7: Overview of the learning stage for detection system.

images represent a subset of the samples of the environment maps (see section 6.2.1). Each feature regions is evaluated with respect to its detection repeatability and descriptor distinctiveness as described in section 6.2.1. For this purpose first the similarity maps are computed for each feature and then the respective view set created. Figure 7.8 shows four images taken from a long sequence with highlighted features during evaluation.

7.3 An edge-based tracking system

Our object detection system has been integrated into an AR tracking framework for the initialization of an marker-less real-time tracking system. This section describes an edge-based tracker developed by Tsin et al [140]. A stereo vision based tracker is described in appendix C.

In AR systems, the tracking process itself is usually not the ultimate goal. It is normally followed by other time-consuming processes such as 3D model rendering or other visualization tasks, depending on the application. Thus a good tracker is required to execute as fast as possible. Tsin et al [140] developed an edge-based tracker which is a)

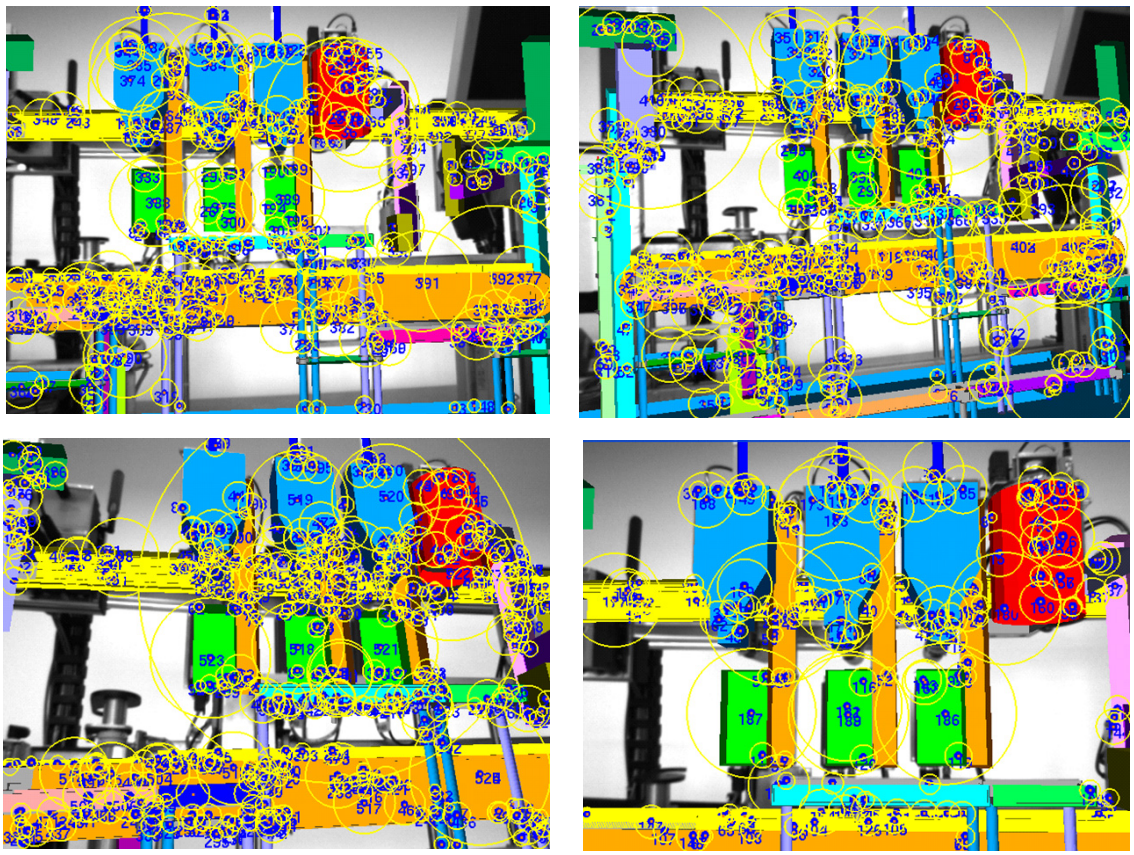


Figure 7.8: Feature evaluation. Four images taken from a long training sequence with highlighted feature regions during the evaluation procedure.

very fast (frame-rate up to 250 VGA frames per second, or as little as 4 milliseconds per frame on standard computers), b) accurate (jitter-free) and c) insensitive to heavy clutters, low contrast model edges or repetitive structures. The iterative tracking algorithm consists of the following components:

1. *Pose initialization:* An initial pose of the camera is given to the tracker, either by the detection system for the first frame or during re-initialization, or by the pose successfully estimated from the previous frame.

2. *Projection and sampling:* The edge points are projected into the image and sampled. Invisible points are ignored by conducting a visibility study for each 3D point.

3. *Feature detection:* Edge candidates are detected in the image. For each sampled point, there are a list of detected edge candidates are considered.

4. *Feature matching:* For the sampled edge points the tentative correspondences in the image are determined.

5. *Optimization:* A cost function is minimized whose global minimum corresponds to the ground truth camera pose [140].

6. Go to 4 until convergence.

For more details the reader is referred to [140]. Figure 7.10 shows some results of the edge-based tracking system.

In order to reconstruct 3D edges automatically an evaluation procedure is used based on the 3D model (see Fig. 7.9). For this purpose, the calibrated training sequence is used (see section 7.2.1). From each frame the edges are extracted using conventional edge detectors [20]. Each edge point will then vote for an existing 3D edge in the 3D model. Finally, the 3D edges with the highest vote rates are selected and used for tracking.

7.4 Detection and tracking management at run-time

This section introduces an AR tracking management framework based on combining the proposed 3D object detection and the tracking system described in the previous section. First, we describe the framework for single object detection and tracking in real-time. This is the subject of the next subsection. In the section 7.4.2 we will then extend this framework to multiple object tracking. The system is tested on real and its performance assessed. Section 7.4.3 presents some experimental results. This framework proves to be fast and reliable enough for industrial AR applications as will be shown in section 7.5.

7.4.1 A framework for single object detection and tracking

This section introduces the single object tracking manager. The behavior of the management system can be modeled with a *finite state machine* (FSM). A FSM is composed of *states*, *transitions* and *actions*. A transition indicates a state change and is described by a condition that would need to be fulfilled to enable the transition. An action is a description of an activity that is to be performed at a given moment.

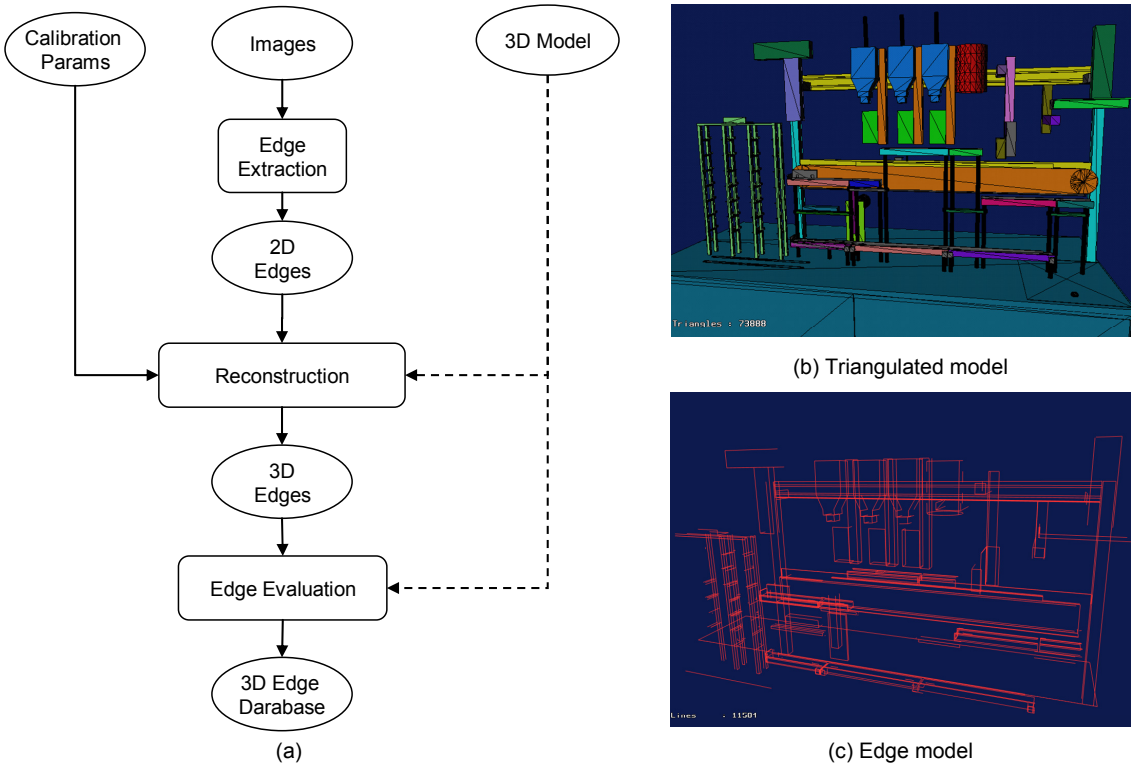


Figure 7.9: 3D edge reconstruction. (a) Overview of the 3D edge reconstruction process. (b) Triangulated model of the coffee machine and the respective 3D edge model.

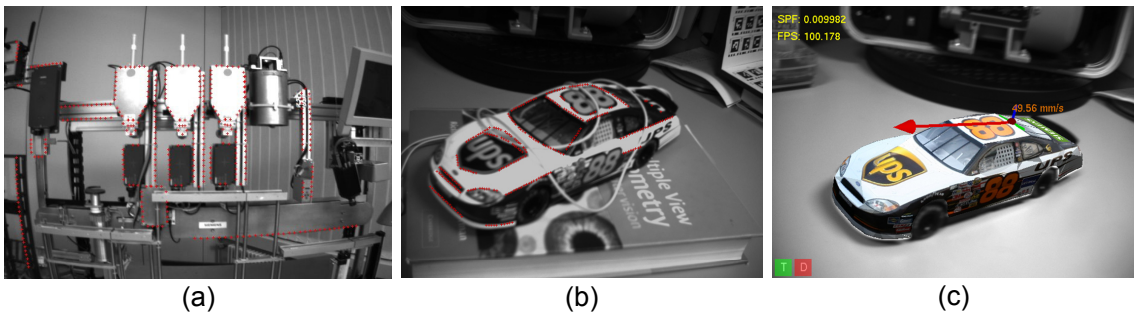


Figure 7.10: Edge Tracker. (a)-(b) Augmented views with edges being tracked. (c) Augmentation with the 3D model and the relative speed between the camera and the object.

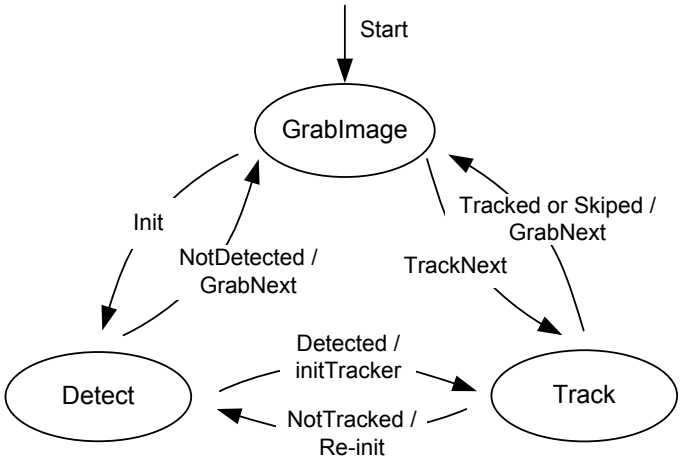


Figure 7.11: State diagram of the single object tracking management system.

There are three major states in the FSM of our tracking manager: detection, tracking or grabbing the current image. Figure 7.11 shows the state transition diagram of this FSM. The system is started with grabbing the current image. Then a state change to the detection step is performed. As long a the target object is not detected in the current image, the next camera image is grabbed and processed. Once the object is detected the computed pose is used to initialize the tracker. Is this initialization successful, next frame is grabbed and tracking is continued. Once tracking fails the system switches back to the detection process automatically. For more robustness we skip a certain number of frames when tracking fails and try to continue tracking. If it fails in all of those frames it goes back to the detector. This parameter depends on the performance of the tracker and its convergence speed. Good results are obtained with our tracker for 10 up to 15 frames. This simple FSM proves to be extremely powerful in particular for AR applications as will be shown in the following sections.

7.4.2 A framework for multiple object detection and tracking

The tracking manager for a single object described in the previous section can be extended to the case of multiple objects. Figure 7.12 shows the respective state diagram. In contrast to the case of a single object, during the first initialization process, the detector tries to detect one of the n objects in the current image. Is an object (indicated by j, $j \in \{1, ..., n\}$ in Fig 7.12) detected, the respective pose is used to initialize the tracker with that object. Once that particular object can not be tracked anymore, the systems switches to the detection procedure of that object j first, before going to the general detection process considering all the objects in the database.

7.4.3 Results

This section presents some results of the combined detection and tracking framework both for single as well as multiple objects. Figure 7.13 shows the results of some frames from a sequence in a maintenance scenario with the control box. The user standing in front of the

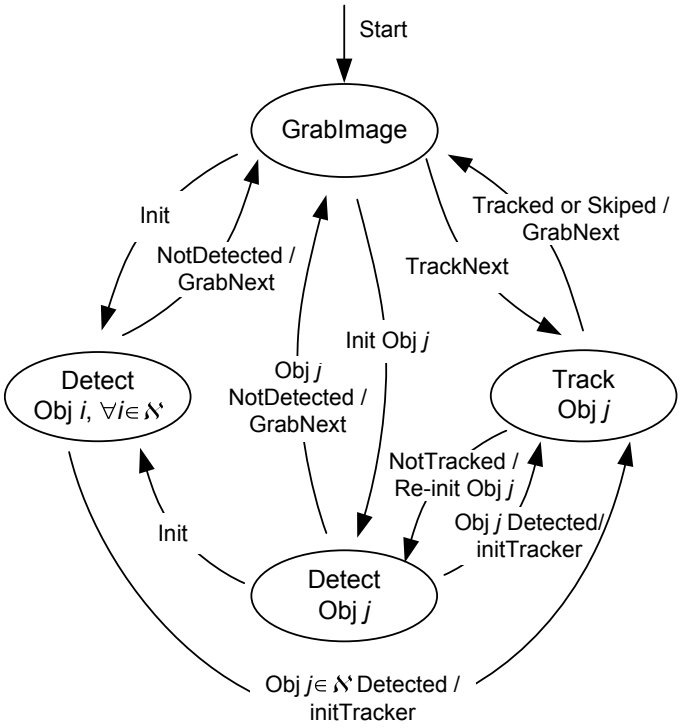


Figure 7.12: State diagram of the multiple object tracking management system.

box wears an HMD and a camera (PointGrey Firefly) with a wide angle objective attached to it. During the maintenance process visual instructions and feedback are provided to user in form of computer rendered 3D model and animations, e.g for switching or pressing buttons. Note in the first frames that a small fraction of the object is already sufficient for the system to detect the object and initialize the tracker. Due to frequent rapid head motions causing blurring effect in the camera images or when the user moves away his head and the object is not in the camera's field of view or when complete occlusions occur during interactions with the box, the tracking is lost but the system recovers quickly by detecting the object again and re-initializing the racker once a small portion of the object is clearly visible.

To demonstrate the performance of the proposed multiple object detection and tracking framework, three independent objects of different sizes have been used: an office cubicle, the control box and the toy car (see Fig. 7.14). Figure 7.15 shows the results of some frames from a long sequence taken within the cubicle. The system is able to switch between the three objects quickly and display some 3D augmentations to show the tracking results. The longest detection time when none of the objects is visible and the system tries to detect the three objects is about 800 to 900 ms.

7.5 AR for industrial applications

Augmented reality systems can be effectively used in applications for industrial processes such as manufacturing, assembly, maintenance and service, quality control inspections, etc. However, only a few cases exist where AR technology has been evaluated [27] or

Figure 7.13: Single object detection and tracking.

Figure 7.14: Experiments with multiple objects of different sizes. Office cubicle with the control box and the toy car.

used in real industrial settings. In the project *intelligent welding gun* [34] we have been able to set up an AR-assisted welding system for automotive use which is now being used continously in the early technical intergration phases for new cars at BMW. Furthermore, the author has been involved in two more projects with the goal of building an AR system for training, service and maintenance purposes. The first project coined *FixIt* [63, 33] is aimed at assisting workers in diagnosis of machine malfunctions in industrial settings and is described in appendix D. The second project coined *ARTESAS* is the subject of the next section.

7.5.1 The ARTESAS project

The project *ARTESAS* (Advanced Augmented Reality Technologies for Industrial Service Applications) aims at the exploration and evaluation of Augmented Reality base technologies for applications in industrial service environments [7]. The project is based on the results and insights of the HMI (Human Machine Interaction) project ARVIKA. ARTESAS is funded by the German Ministry for Education and Research (BMBF) and supervised by the German Aerospace Center (DLR). The major focus areas of ARTESAS are: (a) markerless tracking systems for industrial environments, (b) user friendly AR devices proved under technical and ergonomical aspects, and (c) implementation and evaluation in industrial application fields.

During the last years several vision based tracking technologies have been developed for the mobile utilization in industrial environments. However, current tracking systems still lack maturity and marker-less tracking is still a challenging research problem.

One of the recent scenarios is an industrial coffee machine with some moving parts. An accurate 3D model of the rigid parts of the machine has been created and provided to us in form of a VRML file. Figure 7.16 shows some pictures from a demonstration of the tracking system on the coffee machine at Siemens in Nuremberg, Germany. The user is wearing a wireless head mounted display with a camera attached to it. The entire electronic and the batteries are integrated into a jacket. The monitor in the background in Fig. 7.16(a) and the laptop screen in 7.16(c) display the user's view with the augmented

Figure 7.15: Multiple object detection and tracking.

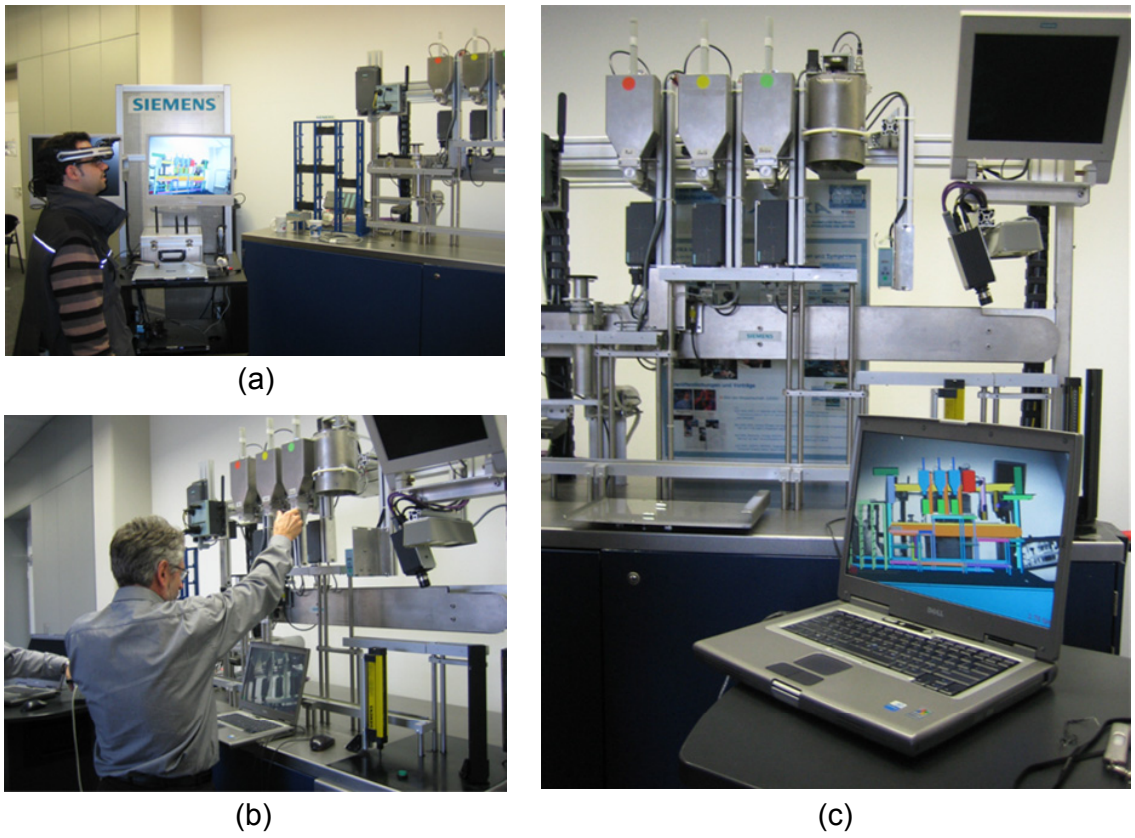


Figure 7.16: The tracking system while being tested on the industrial coffee machine at Siemens in Nuremberg, Germany. (a) The user wears a wireless head mounted display and a camera with the electronic integrated into a jacket. The user's augmented view is also displayed on the monitor (a) and the laptop screen (c).

3D model. To demonstrate the robustness of the detection and tracking system, a poster is placed behind the coffee machine and the lighting conditions changed (see Fig. 7.16(c)). Figures 7.17, 7.18, and 7.19 show some tracking results in three sequences taken with three different cameras in different locations with different background and lighting conditions. The diagram in the bottom row shows the time (in milliseconds) required for processing each frame. The green color indicates that tracking was successful while the red color indicates that the object can not be detected, e.g. when the object is not visible. Once the object can be detected again the tracker is triggered. The slight alignment error is due to the radial distorted camera images. The images will be undistored in real-time in the final system. In all cases the system has been able to detect and track the machine reliably in real-time.

Figure 7.17: Industrial machine sequence 1: tracking results of the coffee machine.

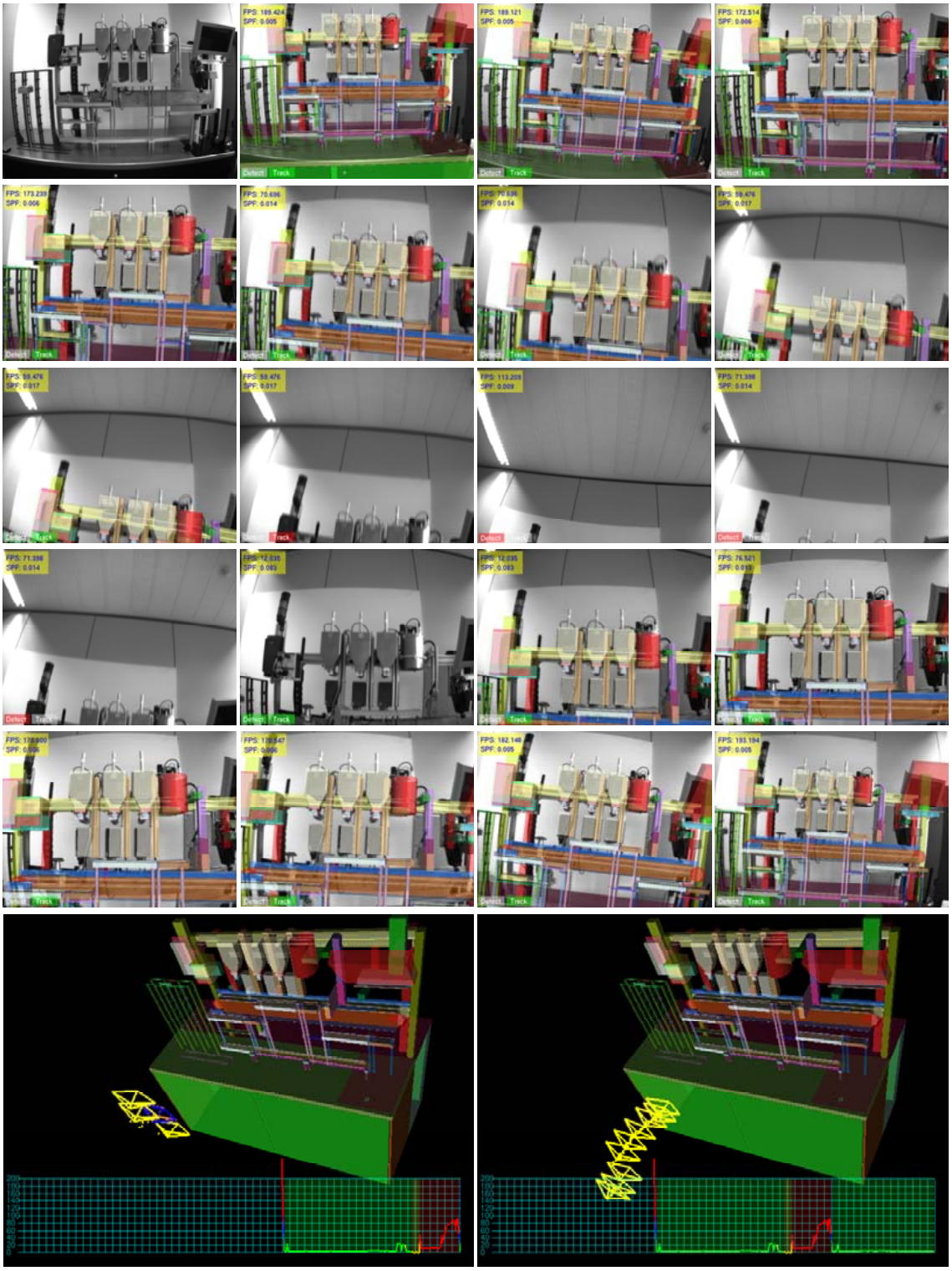


Figure 7.18: Industrial machine sequence 2: tracking results of the coffee machine.

Figure 7.19: Industrial machine sequence 3: tracking results of the coffee machine.

Chapter 8

Discussions and Conclusions

What good are computers? They can only give you answers.

– Pablo Picasso (1881 – 1973)

8.1 Summary

The main contributions of this thesis are two frameworks for fast and robust 3D object detection and pose estimation from a single image. The frameworks rely on an offline learning phase where the 3D object model is used to perform the required time consuming computations so that, at run-time both speed and reliability can be achieved. The first contribution is the formulation of a scalable sensor fusion framework capable of incorporating multiple tracking sensors. The initial pose parameters are estimated iteratively with a coarse-to-fine strategy by taking the uncertainties of the sensors into account. This approach relies on a statistical analysis, probabilistic estimation and propagation as well as fusion of the uncertainties of the sensors. This methodology has been applied to an augmented reality system using mobile and stationary cameras, and can be easily extended to the case of any additional tracking sensors. Stationary cameras are used for non-precise initial pose estimation of the user's head. This pose data is then combined with the one

acquired by the mobile camera to improve the registration accuracy. The pose refinement is based on minimizing the registration error in 3D object space rather than in the 2D image. The explicit representation of the error distributions allows the fusion of the pose data and their uncertainties.

The second contribution is a faster and more robust object detection framework for pose estimation based on local features. For this purpose the class of affine covariant feature region detectors and descriptors have been introduced in chapter 5. Local features bring tolerance to partial occlusions and cluttered backgrounds. The proposed detection framework relies on a scalable and compact representation of the object of interest. During the training process the representation is built by integrating the three-dimensional object geometry and appearance information using statistical learning techniques. For this purpose we conduct a statistical analysis and evaluation of the appearance distribution and shape of object features in the viewing space combining real and synthetic viewpoints. Instead of the local planarity assumption used in conventional approaches, the proposed method is able to learn the visibility distribution of the variations in the local multiple view feature descriptors considering their known geometry. At run-time this representation is used during the matching and pose estimation processes to limit the number of hypothesis by incorporating both photometric and 3D geometric consistency constraints. Fusion of both appearance and geometric information rather than using them in separate procedures have been shown to be very effective improving both time and functional performance. Thereby, the wide-baseline problem of finding matches between sets of feature correspondences between the actual image and object model is formulated as a classification problem. It has been shown that using a bayesian classifier yields a powerful matching method well adapted to object detection. Moreover, the compact object representation allows to reduce the effect of the complexity of the 3D model on the run-time performance and makes the method especially for large environments very powerful.

Another contribution of this thesis is a kind of exhaustive but automatic evaluation procedure has been introduced to compare the performance of different feature detectors in the learning process. Given an arbitrary object this method allows to determine the best performing feature detector(s) in terms of functional and time performance. In our industrial AR applications, this performance evaluation is aimed at presenting an integrated part of the off-line procedure for selecting the appropriate feature detector(s) for each region of interest or eventually set of viewing directions. The evaluation of the feature detectors is performed in the context of matching and detection ability of the same object observed under different viewing conditions with uncontrolled lighting.

A final contribution involves the transfer of an augmented reality tracking framework to industry. The current object detection system has been integrated into an AR tracking framework for the initialization of a marker-less real-time tracking system. The experimental results have proved the system to be fast and reliable enough for industrial AR applications.

8.2 Limitations

The feature based detection system relies on scale invariant or affine covariant feature regions. An obvious limitation of these features is reliance on texture. Some objects such as industrial machines or a car body are essentially textureless, yet easily recognizable (for

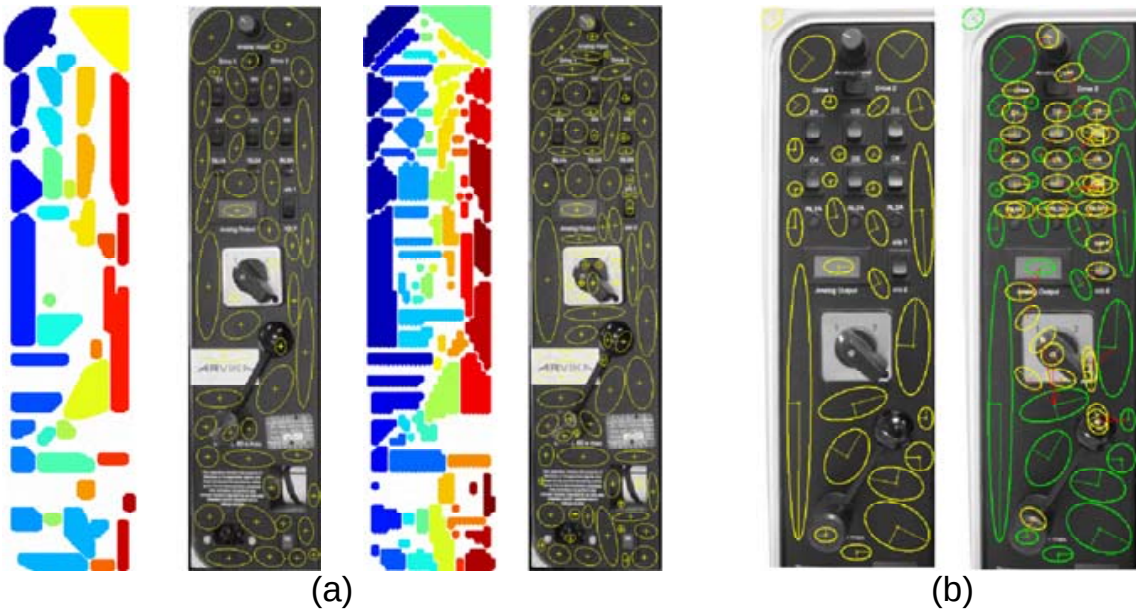


Figure 8.1: Support region. (a) Automated computation of Homogeneous regions with different sizes. (b) Hessian-affine feature regions and the respective nearby support regions.

humans). Handling such objects will require image feature detectors and descriptors that better convey shape information as opposed to appearance, yet capture an appropriate level of viewpoint invariance.

Another limitation of the system are local appearance ambiguities. The use of a single descriptor for matching has the drawback that it fails to consider global context to resolve ambiguities that can occur locally when the object has multiple similar *repetitive* features. The current system is able to identify the repetitive features automatically and exclude them from the database.

The current implementation needs about 0.3 second on a 2.8 GHz Pentium IV PC. The most computational cost intensive component in our algorithm is the extraction of covariant features. Even though some effort was spent on optimizing the methods, there is still scope for improvement.

The current system has been designed for detection of rigid parts of objects. Detection of articulated or even non-rigid objects is part of the future work.

The sensor fusion framework has been shown to be robust against viewpoint and illumination changes. However, because of the global image based matching method it is quite sensitive to severe partial occlusions and cluttered backgrounds. Combining this framework with the feature based approach by integrating additional tracking sensors could improve the results significantly.

8.3 Future work

The work undertaken for the production of this thesis has revealed numerous areas in which more research and development are required. This section identifies a number of directions of future work.

In the sensor fusion framework the pose parameters are estimated in two estimation and refinement steps. Thereby the second step is independent of the first one and can require several iteration steps to converge. This can be improved in a future work by considering the positional uncertainty estimated using the stationary cameras during the refined pose estimation step. The pose refinement is based on minimizing the object space collinearity errors. A more efficient and faster solution could be the minimization of both object and image space collinearity errors simultaneously for pose estimation.

The extraction of affine covariant features is currently the most computational cost intensive component in our algorithm. As a future work we intend to design a fast feature extractor using the known 3D geometry of the target object in order to reduce the complexity and increase the efficiency of the feature extraction algorithm.

The 3D model of a target of interest can be furthermore exploited for faster object detection. As a first step toward this goal we used environment maps generated from all different viewpoints in the scene. The amount of data storage and the time required in the learning stage is not an issue in our current industrial application. However, the number of the required environment maps can be reduced efficiently based on the 3D object structure and distance to the object.

The current approach relies on textured objects. To increase the range of target objects for our approach, we will investigate the use of additional image features such as edges, line segments and even homogeneous regions. We believe that a framework can be designed to find out the most informative features in any given situation and, thus, to allow us to mix different image features in a natural way. Ideally multiple feature detectors should be used in parallel in a hierarchical way. As an example figure 8.1 shows how neighboring homogeneous regions can be used as characteristics. The basic idea is to create and assign to each model feature a set of so called *support regions* or *entities* for unique characterization in the 3D model feature database. These homogeneous regions shown in figure 8.1(a) have been extracted during the offline process and are used at run-time as support regions, based on the geometric transformation of nearby existing features.

Starting with an initial set of matches at runtime a process would gradually explore the test image, recursively looking for more and more matches from the initial ones. This can happen in a similar way to the method in [36] but in a hierarchical way to guarantee real-time performance. For each feature the corresponding support regions are propagated to the test image constructing new matches based on their known geometric transformation. The method is repeated iteratively for each new feature correspondence so that the amount of matches grows after each iteration. Once enough reliable feature matches are found, pose estimation is applied and the results verified.

The use of a single descriptor for matching has the drawback that it fails to resolve ambiguities that can occur locally when the object has multiple similar *repeatable* features. One possible way to tackle this problem is to consider global context: The feature descriptor is augmented with a context vector that contains shape information from a larger neighborhood similar to [95]. The global context scale can be made invariant since it is a function of the feature size.

Since during feature extraction in the test image the feature region location and orientation can slightly deviate from the learned features in the database, the uncertainties of the features can be taken into account based on which a refinement process would be

conducted. Using the uncertainties of each individual feature, an error estimation and propagation can be performed for the respective support regions.

The challenge of a framework with multiple detectors and descriptors is to come up with an efficient hierarchical representation of the feature detectors and descriptors to guarantee real-time performance. This would mean that for instance the most reliable features with lower extraction time need to be processed before the more complex ones where eventually refinement is needed. In the case of descriptors a set of possible types of hierarchical descriptors can be considered: a) nearby homogeneous regions, b) different scaled versions of the model regions, c) nearby features like edges, lines, colors, d) topological arrangements of features, Barycentric neighborhood constraint, Belief propagation techniques, e) sideness constraint: the center of a region should be on the same side of the directed line going from the center of a second region to the center of a third region. f) ratio of regions or region intersections as an invariant of the affine transformation, g) ratio of lengths of line segments along rays emanating from the region's center.

During learning several sets of support descriptors are generated for each specific model region for different uncertainties. At run-time for each feature the set of support regions and descriptors is selected based on the uncertainty of the estimated affinity. Ranking inside each set of the support regions (descriptors) is based on a) their uncertainty (covariance matrix), b) complexity (computational time), c) likelihood of detection (overlap error and similarity measure) which depends on the information content, and d) the threshold of time given

These functionalities can be implemented in form of a set of distributed *agents* each of which starting with a single match tries to explore the test image and find new correspondences. Furthermore, these agents can be used to form a new data format by augmenting a CAD format where beside geometric content, visual information about reliable features for detection as well as tracking is contained.

Appendix A

Medical AR applications

Two medical AR applications in minimally invasive heart surgery were part of the collaboration project HEART (Heart surgery Enhanced by Augmented Reality Techniques) between TU Munich and the German Heart Center in Munich. One application was the visualization of the optimal placement of ports for minimally invasive surgery [138]. The other application is assistance in the placement of aortic stents [10]. The first AR application aimed at providing an efficient solution to the problems of teleoperator based heart operations, namely the optimal port placement and intra-operative navigation in robotically assisted minimally invasive cardiovascular surgery. Traub et al [138] developed a novel system incorporating both port placement planning and intra-operative navigation. The optimal port placement is planned offline on a three-dimensional virtual reconstruction of the patient's computed tomography (CT) scan. Using this planned data an accurate in-vivo port placement is performed by superimposing virtual models of the thorax and the teleoperator arms on their real world counterparts (see Figure A.1). The collision detection method ensures an intersection-free planning and proper placement of the teleoperator arms to access the volume of interest. Thus, a significant reduction of operation time can be obtained by a precise and collision-free planning and placement of teleoperator arms.

Figure A.1: Medical AR Applications. Optimal port placement using AR [138].

Appendix B

Mean shift clustering

This section described the mean shift clustering algorithm that has been used in section sec:viewSets for computing the view sets. Mean shift [22] is a simple iterative procedure that shifts each data point to the average points in its neighborhood. The mean shift algorithm can be generalized to make k-means like clustering algorithms to its special case [22] .

Let $S \subset X$ be a finite set in the n-dimensional Euclidean space, X. Let K be a *kernel* and $w : S \to (0, \infty)$ a $weightfunction$. The *sample mean* with kernel K at $x \in X$ is defined as

$$m(x) = \frac{\sum_{s \in S} K(s - x)w(s)s}{\sum_{s \in S} K(s - x)w(s)}. \tag{B.1}$$

Let $T \subset X$ be a finite set (the *cluster centers*). The difference $m(x) - x$ is called *mean shift* in Fukunaga and Hostetler [42]. The evolution of T in the form of iterations $T \leftarrow m(T)$ with $m(T) = m(t) : t \in T$ is called a *mean shift algorithm*. For each $t \in T$, there is a sequence $t, m(t), m(m(t)), ...$, that is called the *trajectory* of t. The weight $w(s)$ can either be fixed throughout the process or re-evaluated after each iteration. It may also be a function of the current T. The algorithm halts when it reaches a fixed point

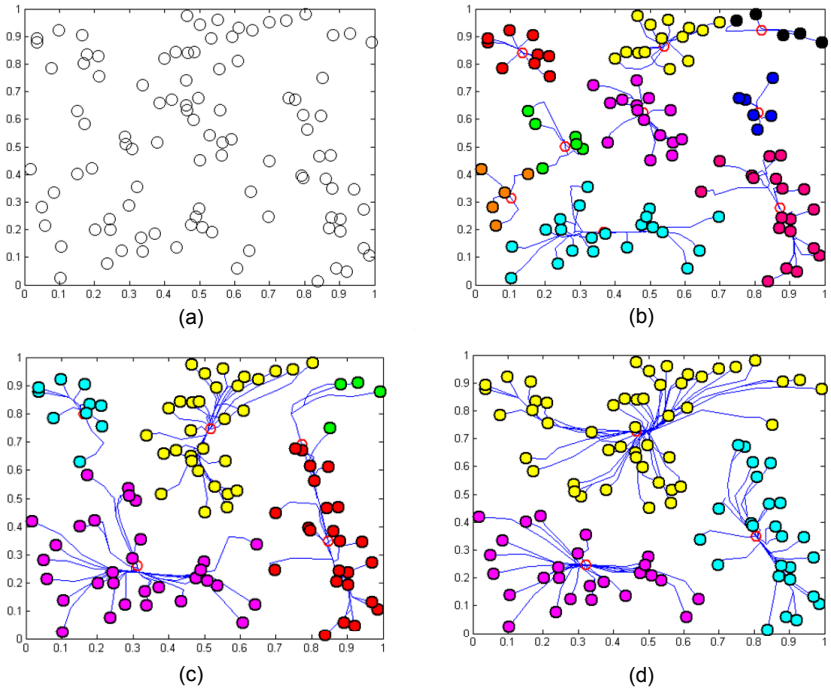


Figure B.1: Mean Shift Clustering. (a) The initial data set. (b) Mean shift trajectories and clustering with a truncated Gaussian Kernel ($\lambda = 0.15$), 8 iterations and 9 clusters. (c) Mean shift process with a truncated Gaussian Kernel ($\lambda = 0.2$), 9 iterations and 5 clusters. (d) Mean shift process with a truncated Gaussian Kernel ($\lambda = 0.25$), 7 iterations and 3 clusters.

$(m(T) = T)$. When T is S, the mean shift algorithm is called a blurring process, indicating the successive blurring of the data set, S.

The kernel we are using for computing the view sets is a truncated Gaussian kernel, which is a Gaussian kernel $e^{-||x||^2}$ multiplied by a flat kernel.

$$G_\lambda^\beta(x) = \begin{cases} e^{-\beta||x||^2}, & \textbf{if } ||x|| \leq \lambda \\ 0, & \textbf{if } ||x|| > \lambda \end{cases} \tag{B.2}$$

The computational cost of an iteration of mean shift is $O(n^2)$, where n is the size of the data set. However, it is possible to reduce the time complexity to $O(nlogn)$, using a better storage of the data by considering only neighboring points in the computation of the mean. For more detail about mean shift the reader is referred to [22]. Figure B.1 shows the results of mean shift clustering using a set of points as data set. Figure B.2 and B.3 show some results of the mean shift clustering algorithm using the position of environment maps surrounding an object of interest as data set for simulated and real data, respectively (see also section 6.2.1).

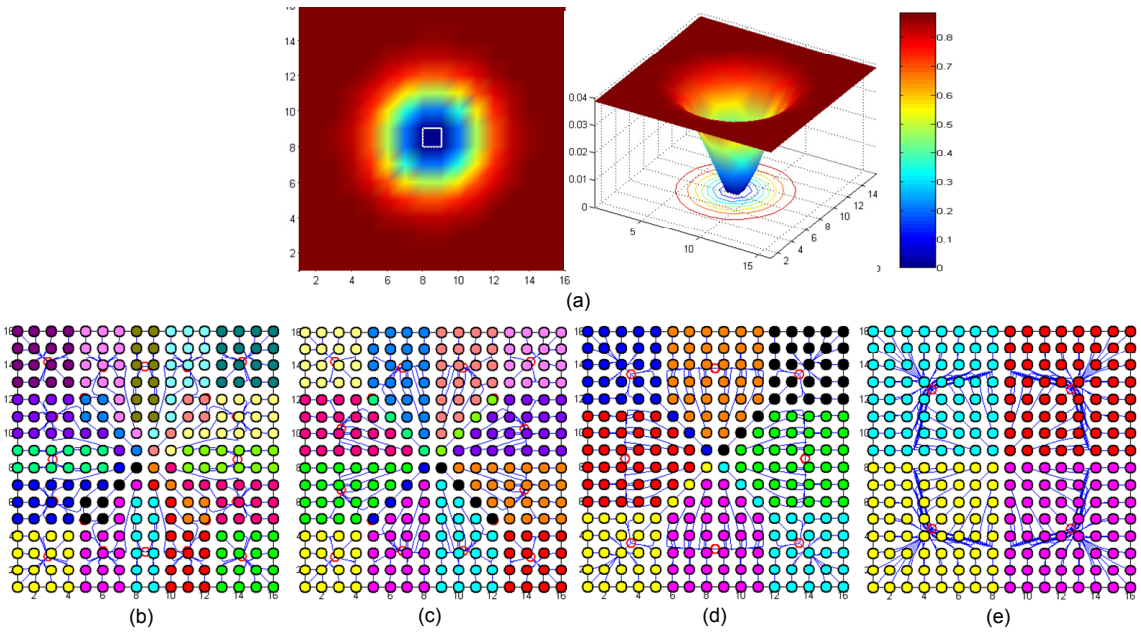


Figure B.2: Mean Shift Clustering. (a) Simulated denstiy function. (b) Gaussian Kernel ($\beta = 0.6, \lambda = 2$), 17 iterations and 20 clusters. (c) Gaussian Kernel ($\beta = 0.9, \lambda = 5$), 14 iterations and 16 clusters. (d) Gaussian Kernel ($\beta = 0.5, \lambda = 4$), 14 iterations and 8 clusters. (e) Gaussian Kernel ($\beta = 0.2, \lambda = 4$), 10 iterations and 4 clusters.

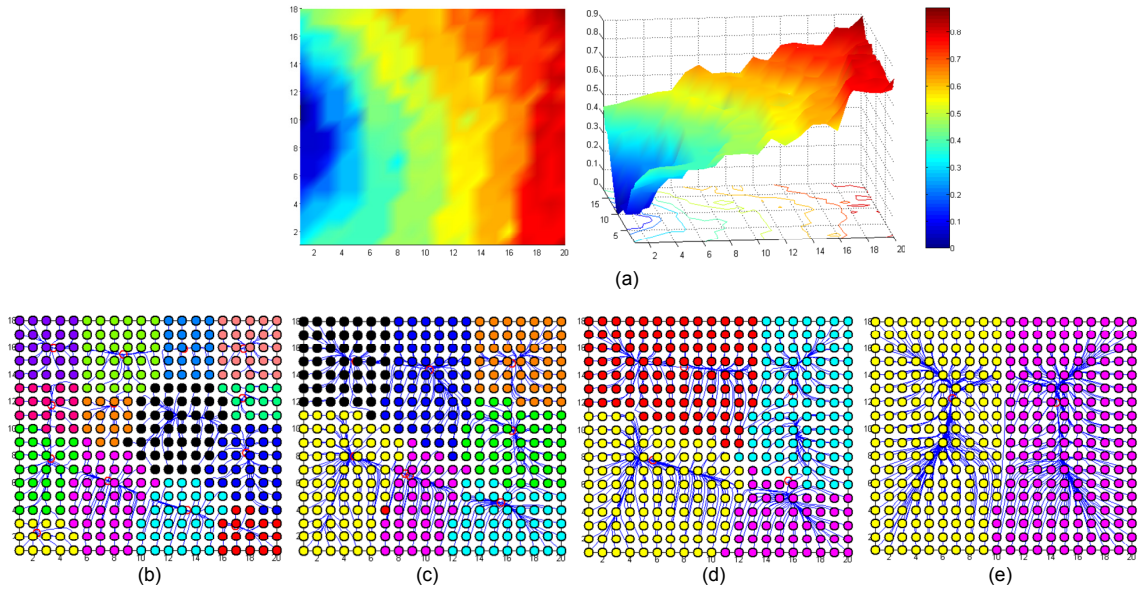


Figure B.3: Mean Shift Clustering. (a) Similarity value distribution. (b) Gaussian Kernel ($\beta = 1, \lambda = 3$), 14 iterations and 14 clusters. (c) Gaussian Kernel ($\beta = 0.2, \lambda = 4$), 14 iterations and 8 clusters. (d) Gaussian Kernel ($\beta = 0.2, \lambda = 6$), 30 iterations and 4 clusters. (e) Gaussian Kernel ($\beta = 0.1, \lambda = 7$), 14 iterations and 2 clusters.

Appendix C

A simple stereo vision based tracker

This section presents a robust model-based tracker using stereo vision [100]. The combined use of a 3D model with stereoscopic analysis allows accurate pose estimation in the presence of partial occlusions by non rigid objects like the hands of the user. Furthermore, using a second camera improves the stability of tracking and also simplifies the algorithm.

Introduction

Model-based visual tracking methods rely on a 3D model of the target object(s) and try to compute a 3D pose that correctly re-projects the features (e.g. points, edges, line segments) of a given 3D model into the 2D image. After an initialization of the system where initial matches between 2D features and the corresponding 3D model features are established, 2D tracking algorithms like KLT [83] are used to track the features from frame to frame [43, 145, 12]. The resulting 2D-3D matches of every frame are used for pose estimation using standard methods e.g. Tsais algorithm [139].

There are mainly three major problems arising when tracking 2D features. First, the matched features may drift or even be completely wrong (outliers) when pixel-based correlation techniques establish frame-to-frame correspondences. Secondly, since the camera can move, the initial features may not be visible all the time, so new features need to be found and tracked properly at run time. And thirdly, due occlusions e.g. caused by users

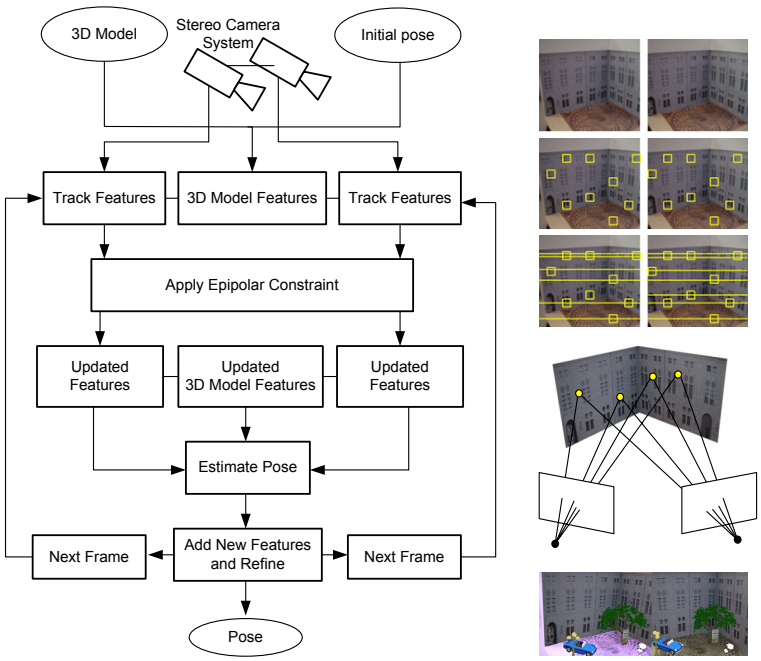


Figure C.1: Overview of the model-based stereo tracking system.

hands, the target object may be only partially visible

To handle these problems new visible features need to be added and matched online. Several approaches to monocular tracking [43, 145] try to detect and discard outliers matches by using robust estimation techniques like RANSAC, which tend to be quite time consuming. We show a tracker that incorporates stereoscopic vision with the 3D model for this purpose. Stereoscopic analysis provides the important epipolar constraint [48]. By applying this constraint to the stereo image pair, outliers (false correspondences from monocular tracking) can be easily detected and rejected, avoiding the more time consuming robust estimation techniques. Using a second camera also improves the accuracy of the tracking. Furthermore, it improves greatly the stability and simplifies the algorithm. The requirement of an existing 3D model is, in practice, not an issue since such models already exist in many applications or can be created using either automated techniques or commercially available products. In this demo we used the 3D modeling program Canoma based on single images taken from the target objects.

System overview

Two cameras are mounted side by side as a stereo camera system. We use the method proposed by Zhang [154] to determine the intrinsic matrices of both cameras and the relative transformation between them.

Using a few user selected 2D-3D correspondences, the model is registered with the images. From then on, the system tracks the optical flow of salient features using the KLT tracker [83], rejects outliers based on the epipolar constraint, and updates the pose of the camera in every frame (see Figure C.1). A feedback loop supplies new salient feature points in each frame to make the tracking more stable under various conditions,

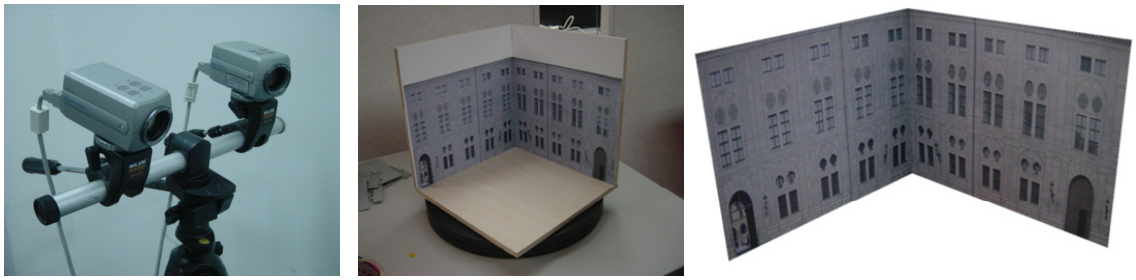


Figure C.2: (a) Stereo camera rig with two Sony DFW-VL 500 Firewire cameras. (b) A test scene consisting of two planar textured planes. (c) The 3D model.

e.g. occlusions by the users hands. The algorithms are described in more detail in the following two sections.

Stereo 3D tracking

First we detect strong corners in both images using the Shi-Tomasi algorithm [131]. The feature points are then tracked in each camera image independently using a pyramidal implementation of the Lucas-Kanade algorithm [83]. However the matched points may drift or even be wrong. Therefore, the epipolar constraint is applied to reject outlier matches. The epipolar constraint states that if a point $\mathbf{p}$ in the first image and a point $\mathbf{q}$ in the second image correspond to the same 3D point in the real world, they must satisfy the following equation $\mathbf{q}^{\mathbf{T}} F \mathbf{p} = 0$, where F is the fundamental matrix that is the algebraic representation of the epipolar geometry between two images [48]. This equation means that point q must pass through the epipolar line defined as $F\mathbf{p}$ in the second image and vice versa. By applying this constraint to the stereo image pair, outliers can be easily rejected and the pose of the camera is estimated using Tsai's algorithm [139].

Adding New Features

To ensure that there will be a sufficient number of feature correspondences for pose estimation in the consecutive frames, new stable features are added based on the following two criteria:

Texture: The feature point has rich texture information. For this purpose strong corners are selected from the image based on [131], then back-projected to the 3D model in order to get their 3D coordinates, considering only the interest points that are on the object surface.

Visibility: The feature must be visible in both cameras. A feature point is visible in both images if the respective 3D model points are close enough to each other. Adding new features by every frame alleviates the tracker drifting problem and improves the accuracy and stability of the tracker.

Experimental Results

We have implemented both monocular and stereo based trackers and tested the algorithms with several live real data. The software was developed using Microsoft Visual C++ and OpenGL under both Windows 2000 and XP. Furthermore we use the computer vision library OpenCV, which is distributed by Intel and is freely available online. The current implementation runs in real-time (30-40 FPS) with a resolution of 320x240 on a Laptop (DELL Precision 50) with 2.0 GHz Pentium 4 CPU. For user interaction a platform independent graphical user interface was built using GLUI (see Figure C.3(a)). Using such an interface most important parameters can be changed on-line and their effect on tracking can be seen immediately. This interface was used to find a configuration of parameters with the best tracking results. Due to the platform independence of the whole system, the software can be run on other platforms (Unix, Linux, Macintosh) with little modifications. Figure C.2 (a) and (b) shows the stereo camera rig and the test scene used for testing the tracking algorithms. To build a 3D model we used the program Canoma from MetaCreations to reconstruct a 3D scene using several images of it. Figure C.2(c) shows the 3D model of the target object used for experiments.

We ran the monocular and stereo tracking algorithms with several target objects. Figure C.3(b) shows the first monocular tracking results of a sequence taken from the outside of our AR Lab at TUM. The tracking was done using the monocular technique by removing the epipolar constraint. In this case after a while the tracking results poor accuracy and the model starts to drift.

In Figure C.3(c) the results of a sequence are shown using the stereo-based approach. In this sequence a rectified texture of the facade of a building was used. No drifting can be seen due to the strong epipolar constraint.

Last Experiment was done using a 3D model rather than a planar one. Figure C.4 shows two views of a test scene consisting of two textured surfaces and the respective reconstructed 3D model. The scene was then augmented with several virtual objects shown.

Conclusions

We have presented a simple but robust method for real-time 3D tracking using two cameras. We showed that the rich information from the stereo cameras (or multiple cameras) enables the tracking system to achieve a much higher level of robustness than the monocular version. Furthermore, this approach can still continue tracking and produce reasonable results when one of the cameras is fully occluded.

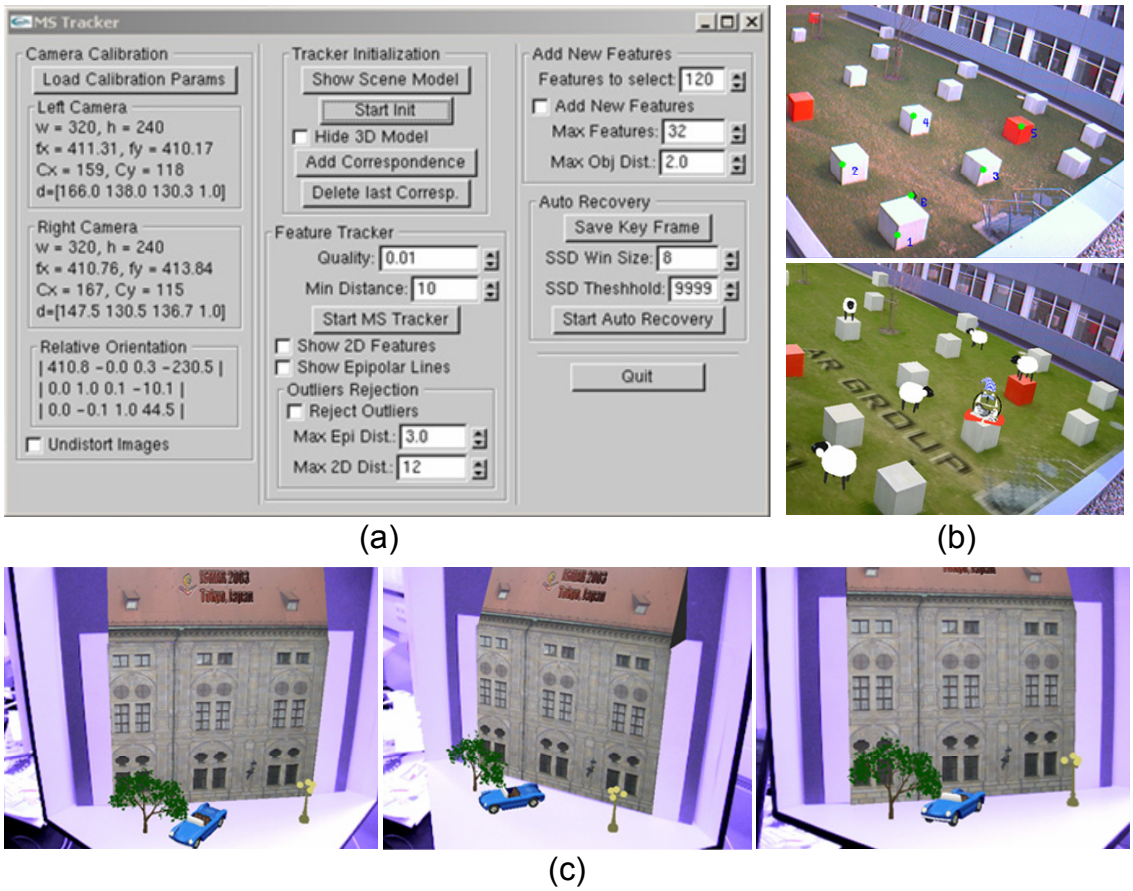


Figure C.3: (a) The graphical user interface of the model-based tracking system. (b) Sheep on the loose: window view of the AR Lab at TUM and the augmented view with virtual objects. (c) Tracking results on a sequence of images.

Figure C.4: Stereo tracking results: Augmented stereo images with virtual objects.

Appendix D

The FixIt project

This section describes an AR system coined *FixIt* aimed at assisting workers in diagnosing machine malfunctions in industrial settings [63, 33].

Industrial machines or robots need to be inspected and repaired by workers periodically or on demand. One of the major problems in diagnosing the problem to be fixed is to establish a mapping between the internal control state of the computer and the actual physical state of the machine. To this end, the maintenance person needs to have a clear understanding of both, as well as their expected and actual dynamic changes over time. By overlaying virtual information of the control system directly onto the machine while it is in operation, AR has the potential to help workers obtain a better understanding of the reasons for malfunctions. The result is an intricate new, highly immersive net of interactions and relationships in a man-computer-machine triangle. To this end, they need a system that helps them diagnose problems while the system is running to identify signaling malfunctions. We show using a Fischer Technik robot, how the current robot state (motors on/off, switches pressed, etc.) overlaid on the respective components of the robot so that the mechanic can see which pieces are malfunctioning or which cables are not propagating signals correctly.

Figure D.1 depicts the schematic of the FixIt system. It consists of three major components:

- A tracking component is required to determine the current physical state of the robot,

as well as current pose of the camera or the viewer representing the maintenance person.

- A control component that sends commands to the robot e.g. to move its arm or rotate. The same commands are send to the visualization component.
- A visualization component, overlaying the control state information on to the robot, according to the camera's current pose.

For analyzing malfunctions, workers can request the robot from the robot application side to perform a set of physically reachable motions, e.g. to move the tip of the robot to certain three-dimensional positions. The control program internally divides the request into a sequence of control commands steering the individual motors.

Since both the worker and the robot move, tracking is required both for determining the current worker position and for determining the current physical shape of a machine such that augmentations are actually placed correctly onto individual machine parts while these are moving. In the case of a markerless tracking system, this means that both the detection and tracking system need to be able to handle articulated objects and the respective dynamics.

We have built a prototype of the FixIt system was presented on ISMAR 2003 [33]. For the demonstration we built a toy robot (welding robot) by Fischer technik that was connected to a computer running the application (see Figure D.2(a)). For the visualization an animated 3D model of the robot is required to account for any potential robot movements. Figure D.2(b) shows a VRML model of the Fischer Technik robot. This model was composed from a library of basic models describing individual Fisher Technik parts. As depicted in Figure D.1 the visualization component receives input both from the robot control component and from the tracking component. It is able to steer the animation of the virtual model and highlight active units of the robot according to recently issued robot control commands assisting the malfunction diagnosis state (see Figure D.2(c)).

This tool can be used to exploit the redundancy between issued control commands and visually tracked robot positions to automatically identify and diagnose malfunctions. In this case, there is an interesting discrepancy between the internal (virtual) state of the machine and its real (physical) state. If due care is being taken to formalize this discrepancy and to determine it from physical measurements (tracking data) and control status information, it can form the basis for very powerful diagnosis for machine repair personnel.

Yet, the discrepancy alone will not suffice. It needs to be supplemented with program context data which indicates what the robot control program is trying to achieve while issuing a certain sequence of robot control commands.

The formalization of such program context, as well as the standardized indication of a program control state (for an arbitrary machine) and the visualization of discrepancies between the virtual and real robot state are important visualization issues that need to be addressed in order to generate authoring systems that will be suitable as augmented debugging aides for larger sets of robots.

Within this project, we have been able to lay the ground work toward exploring these and other exciting issues related to an online diagnosis of malfunctions of machines while they are in operation.

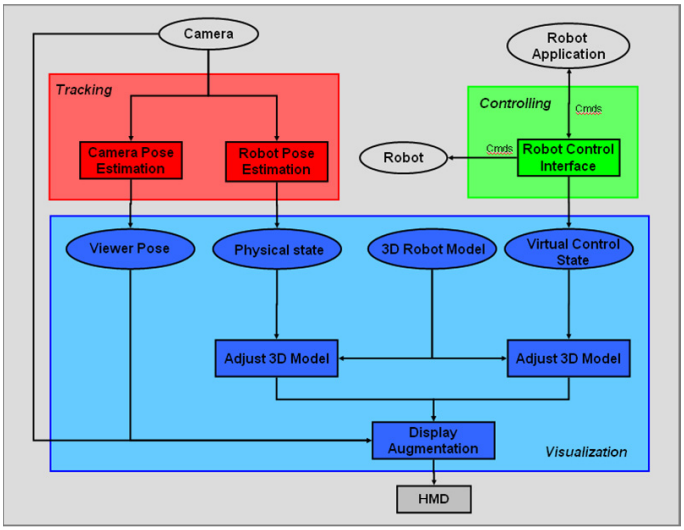


Figure D.1: Schematic of the FixIt system.

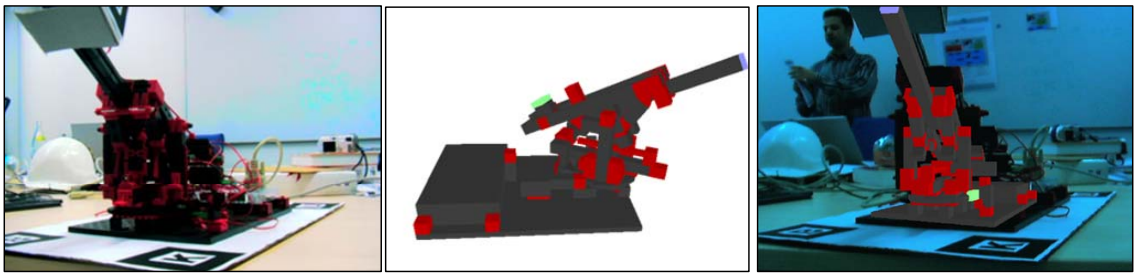


Figure D.2: (a) Robot. (b) 3D model. (c) Overlayed image with the control state of the robot.

Bibliography

[1] E. H. Adelson and J. R. Bergen. *Computational Models of Visual Processing, Chapter 1: The Plenoptic Function and the Elements of Early Vision.* The MIT Press, Cambridge, Mass, 1991.

[2] N. Allezard, M. Dhome, and F. Jurie. Recognition of 3d textured objects by mixing view-based and model-based representations. *ICPR*, 2000.

[3] Yali Amit and Donald Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–1588, 1997.

[4] M. Appel and N. Navab. 3D reconstruction from co-registered orthographic and perspective images: theoretical framework and applications. In *International Conference on Pattern Recognition*, pages IV: 21–26, 2002.

[5] M. Appel and N. Navab. Registration of technical drawings and calibrated images for industrial augmented reality. *Machine Vision and Applications*, 13(3):111–118, 2002.

[6] A.R.T. Advanced real time tracking. *www.ar-tracking.com.*

[7] ARTESAS. *http://www.artesas.de/.*

[8] Ronald Azuma. A survey of augmented reality. *Presence*, 6(4):355–385, 1997.

[9] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics and Applications*, 21(6):34–47, 2001.

[10] R. Bauernschmitt, E.U. Schirmbeck, M. Groher, P. Keitler, M. Bauer, H. Najafi, G. Klinker, and R. Lange. Navigierte platzierung endovaskulaerer aortenstents. *Zeitschrift fuer Kardiology*, S3:116, 2004.

[11] A. Baumberg. Reliable feature matching across widely separated views. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages I: 774–781, 2000.

[12] Reinhold Behringer, Jun Park, and Venkataraman Sundareswaran. Model-based visual tracking for outdoor augmented reality applications. In *ISMAR*, page 322. IEEE Computer Society, 2002.

[13] J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages 1000–1006, 1997.

[14] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell*, 24(4):509–522, 2002.

[15] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–255, 1992.

[16] Jean-Yves Bouguet. Camera calibration toolbox for matlab. *http://www.vision.caltech.edu/bouguetj/calib/doc/index.html*.

[17] Boujou2d3. *http://www.boujou.com*.

[18] Gary R. Bradski. Computer vision face tracking for use in a perceptual user interface.

[19] M. Brown and D. Lowe. Invariant features from interest point groups. In *British Machine Vision Conference*, page Poster Session, 2002.

[20] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, 986.

[21] T. P. Caudell and D. V. Mizell. Augmented reality: An application of heads-up display technology to manual manufacturing technology. *Proceedings of Hawaii International Conference on System Sciences*, pages 659–669, 1992.

[22] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Anal. Mach. Intell*, 17(8):790–799, 1995.

[23] K. Chia, A. Cheok, and S. Prince. Online 6dof augmented reality registration from natural features. *International Symposium on Mixed and Augmented Reality (ISMAR'02)*, October 2002.

[24] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 2002.

[25] J. L. Crowley. A representation for visual information, phd thesis. In *CMU Robotics Institute*, 1982.

[26] J. L. Crowley and A. C. Parker. A representation for shape based on peaks and ridges in the difference of low-pass transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:156–170, 1984.

[27] Dan Curtis, David Mizell, Peter Gruenbaum, and Adam Janin. Several devils in the detail: Making an ar app work in the airplane factory. *IWAR*, pages 47–60, 1998.

[28] A. Davison and D. Murray. Simultaneous localization and map-building using active vision for a robot. *PAMI*, July 2002.

[29] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *International Conference on Computer Vision*, pages 1403–1410, 2003.

[30] D. Dementhon and L. S. Davis. Model-based object pose in 25 lines of code. *ECCV*, 1992.

[31] Tom Drummond and Roberto Cipolla. Real-time tracking of complex structures with on-line camera calibration. *Image Vision Comput*, 20(5-6):427–433, 2002.

[32] Tom Drummond and Roberto Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. Pattern Anal. Mach. Intell*, 24(7):932–946, 2002.

[33] Florian Echtler, Hesam Najafi, and Gudrun Klinker. Fixit. *In Demonstration Session at the International Symposium on Augmented and Mixed Reality (ISMAR 2002)*, page 137, 2002.

[34] Florian Echtler, Fabian Sturm, Kay Kindermann, Gudrun Klinker, Joachim Stilla, Joern Trilk, and Hesam Najafi. The intelligent welding gun: Augmented reality for experimental vehicle construction. In S.K Ong and A.Y.C Nee, editors, *Virtual and Augmented Reality Applications in Manufacturing, Chapter 17*. Springer Verlag, 2003.

[35] Olivier Faugeras. *Three-Dimensional Computer Vision*. The MIT Press, 1993.

[36] V. Ferrari, T. Tuytelaars, and L. J. Van Gool. Simultaneous object recognition and segmentation from single or multiple model views. *International Journal of Computer Vision*, 67(2):159–188, April 2006.

[37] V. Ferrari, T. Tuytelaars, and L Van Gool. Integrating multiple model views for object recognition. *CVPR*, 2004.

[38] Vittorio Ferrari. *Affine invariant regions++*. PhD thesis, ETH Zurich, 2004.

[39] François Fleuret and Donald Geman. Coarse-to-fine face detection. *International Journal of Computer Vision*, 41(1/2):85–107, 2001.

[40] Wolfgang Foerstner. A framework for low level feature extraction. In *ECCV '94: Proceedings of the third European conference on Computer Vision (Vol. II)*, pages 383–394, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[41] Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, September 1977.

[42] K. Fukunaga and L. D. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Trans. Information Theory*, 21(1):32–40, January 1975.

[43] Y. Genc, S. Riedel, F. Souvannavong, C. Akinlar, and N. Navab. Marker-less tracking for ar: A learning-based approach. *ISMAR*, 2002.

[44] R. M. Haralick, H. Joo, C.-N. Lee, X. Zhuang, and M. B. Kim. Pose estimation from corresponding point data. *IEEE Trans. Systems, Man, and Cybernetics*, 6(19), November 1989.

[45] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision, Volume II.* Addison-Wesley, 1992.

[46] C. Harris. *Active Vision. Chap. 4: Tracking with rigid models.* MIT Press, 1992.

[47] C.G. Harris and M.J. Stephens. A combined corner and edge detector. *Fourth Alvey Vision Conference*, 1998.

[48] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2004.

[49] William Hoff and Tyrone Vincent. Analysis of head pose accuracy in augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 6(4), October-December 2000.

[50] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternion. *J. Opt. Soc. Amer.*, A-4:629–642, 1987.

[51] D. P. Huttenlocher and S. Ullman. Object recognition using alignment. In *Image Understanding Workshop*, pages 370–380, 1987.

[52] F. Jurie. Solution of the simultaneous pose and correspondence problem using gaussian error model. *CVIU*, 1999.

[53] T. Kadir and M. Brady. Saliency, scale and image description. *International Journal of Computer Vision*, 45(2):83–105, November 2001.

[54] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. In *European Conference on Computer Vision*, pages Vol I: 228–241, 2004.

[55] R. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, pages 35–45, 1960.

[56] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a videobased augmented reality conferencing system. *Proc. 2nd Intl Workshop on Augmented Reality (IWAR).*

[57] Y. Ke and R. Sukthankar. PCA-SIFT: a more distinctive representation for local image descriptors. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages II: 506–513, 2004.

[58] Georg Klein. *Visual Tracking for Augmented Reality.* PhD thesis, University of Cambridge, 2006.

[59] Georg Klein and Tom Drummond. Robust visual tracking for non-instrumented augmented reality. In *Proc. Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'03)*, pages 113–122, Tokyo, October 2003.

[60] Georg Klein and David Murray. Full-3d edge tracking with a particle filter. In *Proc. British Machine Vision Conference (BMVC'06)*, Edinburgh, September 2006. BMVA.

[61] G. Klinker, D. Stricker, and D. Reiners. Augmented reality for exterior construction applications. In W. Barfield and T. Caudell, editors, *Augmented Reality and Wearable Computers*. Lawrence Erlbaum Press, 2001.

[62] Gudrun Klinker, Allen Dutoit, Martin Bauer, Johannes Bayer, Vinko Novak, and Dietmar Matzke. Fata morgana – a presentation system for product design. In *International Symposium on Aumgented and Mixed Reality ISMAR 2002*, 2002.

[63] Gudrun Klinker, Hesam Najafi, Tobias Sielhorst, Fabian Sturm, Florian Echtler, Mustafa Isik, Wolfgang Wein, and Christian Truebswetter. Fixit: An approach towards assisting workers in diagnosing machine malfunctions. In Emmanuel Dubois, Philip D. Gray, Daniela Trevisan, and Jean Vanderdonckt, editors, *MIXER*, volume 91 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004.

[64] Karl-Rudolf Koch. *Parameter Estimation and Hypothesis Testing in Linear Models*. Springer Verlag, Berlin, Heidelberg, 1999.

[65] J. J. Koenderink. The structure of images. *Biological Cybernetics*, 50:363–370, 1984.

[66] Ryo Kurazume, Ko Nishino, Zhengyou Zhang, and Katsushi Ikeuchi. Simultaneous 2d images and 3d geometric model registration for texture mapping utilizing reflectance attribute. *Proc. of Fifth Asian Conference on Computer Vision (ACCV)*, January 2002.

[67] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[68] V. Lepetit, L. Vacchetti, , and P. Fua. Fully automated and stable registration for augmented reality applications. *International Symposium on Augmented Reality (ISMAR)*, September 2003.

[69] Vincent Lepetit, Pascal Lager, and Pascal Fua. Randomized trees for real-time keypoint recognition. *CVPR*, 2005.

[70] Vincent Lepetit, Julien Pilet, and Pascal Fua. Point matching as a classification problem for fast and robust object pose estimation. *CVPR*, 2004.

[71] J. P. Lewis. Fast normalized cross-correlation. *Industrial Light & Magic*.

[72] Y. Li, Y. Tsin, Y. Genc, and T. Kanade. Object detection using 2d spatial ordering constraints. *CVPR*, 2005.

[73] T. Lindeberg. Scale-space theory: A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, 21(2):224–270, 1994.

[74] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, November 1998.

[75] T. Lindeberg and J. Garding. Shape-adapted smoothing in estimation of 3-d depth cues from affine distortions of local 2-d brightness structure. *Image and Vision Computing*, 15:415–434, 1997.

[76] Yonghuai Liu. Improving icp with easy implementation for free form surface matching. *Pattern Recognition*, 37(2):211–226, 2004.

[77] Yonghuai Liu, Longzhuang Li, and Baogang Wei. 3d shape matching using collinearity constraint. *Proceedings of IEEE 2004 International Conference on Robotics and Automation*, April 2004.

[78] D. Lowe. The viewpoint consistency constraint. *Inter. J. Computer Vision*, pages 57–72, 1987.

[79] D. G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, pages 1150–1157, 1999.

[80] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.

[81] D.G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *IJCV*, 8(2), 1992.

[82] Chien-Ping Lu, Gregory D. Hager, and Eric Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE PAMI*, 22(6):610–622, 2000.

[83] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. DARPA Image Understanding Workshop*, pages 121–130, 1981.

[84] J. Matas, D. Koubaroulis, and J. V. Kittler. Colour image retrieval and object recognition using the multimodal neighbourhood signature. In *European Conference on Computer Vision*, pages I: 48–64, 2000.

[85] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, 2002.

[86] Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based rendering system. *Proceedings of SIGGRAPH'95, Computer Graphics Proceedings, Annual Conference Series*, pages 39–46, August 1995.

[87] Jason Meltzer, Stefano Soatto, Ming-Hsuan Yang, and Rakesh Gupta. Multiple view feature descriptors from image sequences via kernel principal component analysis. *ECCV*, 2004.

[88] MetaCreations. *http://www.metacreations.com*.

[89] K. Mikolajczik and C. Schmid. A performance evaluation of local descriptors. *CVPR*, 2003.

[90] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *International Conference on Computer Vision*, pages I: 525–531, 2001.

[91] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *European Conference on Computer Vision*, page I: 128 ff., 2002.

[92] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, October 2004.

[93] K. Mikolajczyk, T. Tuytelaars, Schmid, A. C. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 2004.

[94] F. Mindru, T. Moons, and L. VanGool. Recognizing color patterns irrespective of viewpoint and illumination. *CVPR*, 1999.

[95] E. N. Mortensen, H. Deng, and L. G. Shapiro. A SIFT descriptor with global context. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages I: 184–190, 2005.

[96] H. Murase and S. K. Nayar. Visual learning and recognition of 3-D objects from appearance. *International Journal of Computer Vision*, 14(1):5–24, January 1995.

[97] H. Murase and S. K. Nayar. Detection of 3D objects in cluttered scenes using hierarchical eigenspace. *Pattern Recognition Letters*, 18(4):375–384, April 1997.

[98] Hesam Najafi, Yakup Genc, and Nassir Navab. Fusion of 3D and appearance models for fast object detection and pose estimation. In P. J. Narayanan, Shree K. Nayar, and Heung-Yeung Shum, editors, *ACCV (2)*, volume 3852 of *Lecture Notes in Computer Science*, pages 415–426. Springer, 2006.

[99] Hesam Najafi, Yakup Genc, and Nassir Navab. Fusion of 3D and appearance models for fast object detection and pose estimation. *submitted to International Journal of Computer Vision*, 2007.

[100] Hesam Najafi and Gudrun Klinker. Model-based tracking with stereovision for AR. In *Proc. of International Symposium on Augmented and Mixed Reality*, pages 313–314. IEEE Computer Society, 2003.

[101] Hesam Najafi, Nassir Navab, and Gudrun Klinker. Automated initialization for marker-less tracking: A sensor fusion approach. In *Proc. of International Symposium on Augmented and Mixed Reality (ISMAR 2004)*, pages 79–88. IEEE Computer Society, 2004.

[102] N. Navab. Scene augmentation via the fusion of industrial drawings and uncalibrated images with a view to marker-less calibration. *Proc. 2nd IEEE and ACM Intl Workshop Augmented Reality*, pages 125–33, 1999.

[103] N. Navab. Canonical representation and three view geometry of cylinders. *International Journal of Computer Vision (IJCV)*, 70(2):133–149, 2006.

[104] N. Navab, B. Bascle, M. Loser, B. Geiger, and R. Taylor. Visual servoing for automatic and uncalibrated needle placement for percutaneous procedures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-00)*, pages 327–334, Los Alamitos, 2000. IEEE.

[105] N. Navab, Y. Genc, and M. Appel. Lines in one orthographic and two perspective views. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25(7):912–917, July 2003.

[106] Nassir Navab. Developing killer apps for industrial augmented reality. *IEEE Computer Graphics and Applications*, 24(3):16–20, 2004.

[107] Shree K. Nayar, Sameer A. Nene, and Hiroshi Murase. Real-time 100 object recognition system. *PAMI*, 1996.

[108] Ulrich Neumann and Suya You. Natural feature tracking for augmented-reality. *IEEE Transactions on Multimedia*, 1(1), 1999.

[109] D. Nister. An efficient solution to the five-point relative pose problem. *CVPR*, 2003.

[110] M. Ozuysal, V. Lepetit, F. Fleuret, and P. Fua. Feature harvesting for tracking-by-detection. In *European Conference on Computer Vision*, volume 3953, pages 592–605, 2006.

[111] J. Pilet, V. Lepetit, and P. Fua. Real-time non-rigid surface detection. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages I: 822–828, 2005.

[112] Julien Pilet, Vincent Lepetit, and Pascal Fua. Augmenting deformable objects in real-time. In *ISMAR*, pages 134–137. IEEE Computer Society, 2005.

[113] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. *ICCV*, 1998.

[114] A. R. Pope and D. G. Lowe. Learning appearance models for object recognition. In *Object Representation in Computer Vision II*, page 201, 1996.

[115] F. Preparata and M. Shamos. *Computational Geometry, An Introcusion*. New York: Springer, Berlin, Heidelberg, 1986.

[116] David Pritchard and Wolfgang Heidrich. Cloth motion capture. *Comput. Graph. Forum*, 22(3):263–272, 2003.

[117] RealViz. *http://www.realviz.com*.

[118] H. Rehbinder and B. Ghosh. Multi-rate fusion of visual and inertial data. *In Proceedings of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 97–102, September 2001.

[119] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1508–1511, October 2005.

[120] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using affine-invariant patches and multi view spatial constraints. *CVPR*, 2003.

[121] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. Segmenting, modeling, and matching video clips containing multiple moving objects. *CVPR*, 2004.

[122] Kiyohide Satoh, Shinji Uchiyama, Hiroyuki Yamamoto, and Hideyuki Tamura. Robust vision-based registration utilizing bird's-eye view with user's view. *International Symposium on Augmented Reality (ISMAR)*, October 2003.

[123] Frank Sauer. Ramp, - a medical augmented reality system. *MICCAI satellite workshop AMI-ARCS*.

[124] F. Schaffalitzky and A. Zisserman. Automated scene matching in movies. In *International Conference on Image and Video Retrieval*, pages 186–197, 2002.

[125] C. Schmid. A structured probabilistic model for recognition. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages II: 485–490, 1999.

[126] Cordelia Schmid and Roger Mohr. Combining greyvalue invariants with local constraints for object recognition. In *CVPR*, pages 872–877. IEEE Computer Society, 1996.

[127] Cordelia Schmid and Roger Mohr. Local gray value invariants for image retrieval. *PAMI*, 1997.

[128] Thomas B. Sebastian, Philip N. Klein, and Benjamin B. Kimia. Recognition of shapes by editing shock graphs. In *ICCV*, pages 755–762, 2001.

[129] A. Selinger and R. C. Nelson. A perceptual grouping hierarchy for appearance-based 3D object recognition. *Computer Vision and Image Understanding*, 76(1):83–92, October 1999.

[130] Amit Sethi, David Renaudie, David J. Kriegman, and Jean Ponce. Curve and surface duals and the recognition of curved 3D objects from their silhouettes. *International Journal of Computer Vision*, 58(1):73–86, 2004.

[131] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 593–600, Los Alamitos, CA, USA, June 1994. IEEE Computer Society Press.

[132] G. Simon, A. Fitzgibbon, and Zisserman A. Markerless tracking using planar structures in the scene. *International Symposium on Augmented Reality (ISMAR'00)*, October 2000.

[133] Iryna Skrypnyk and David G. Lowe. Scene modelling, recognition and tracking with invariant image features. In *ISMAR*, pages 110–119. IEEE Computer Society, 2004.

[134] S. M. Smith and J. M. Brady. SUSAN - a new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, May 1997.

[135] I. E. Sutherland. A head-mounted three-dimensional display. In *AFIPS Conference Proceedings*, volume 33, pages 757–764, 1968.

[136] Swain and Ballard. Color indexing. *IJCV: International Journal of Computer Vision*, 7, 1991.

[137] R. Szeliski and S. B. Kang. Recovering 3D shape and motion from image streams using non-linear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, 1994.

[138] Jörg Traub, Marco Feuerstein, Martin Bauer, Eva U. Schirmbeck, Hesam Najafi, Robert Bauernschmitt, and Gudrun Klinker. Augmented reality for port placement and navigation in robotically assisted minimally invasive cardiovascular surgery. In Heinz U. Lemke, Kiyonari Inamura, Kunio Doi, Michael W. Vannier, Allan G. Farman, and Johan H. C. Reiber, editors, *CARS*, volume 1268 of *International Congress Series*, pages 735–740. Elsevier, 2004.

[139] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using of the shelf tv cameras. *IEEE Journal of Robotics and Automation*, 1987.

[140] Yanghai Tsin, Yakup Genc, and Visvanathan Ramesh. A very fast and jitter-free tracker by model-guided edge detection and matching. *Technical Report, Siemens Corporate Research, Inc.*, 2005.

[141] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 586–590, Hawai, June 1992.

[142] T. Tuytelaars and L. J. Van Gool. Content-based image retrieval based on local affinely invariant regions. In *Int. Cof. on Visual Information Systems*, pages 493–500, 2000.

[143] T. Tuytelaars and L. J. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *British Machine Vision Conference*, 2000.

[144] T. Tuytelaars and L. J. Van Gool. Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision*, 59(1):61–85, August 2004.

[145] L. Vacchetti, V. Lepetit, and P. Fua. Fusing online and offline information for stable 3d tracking in real-time. *CVPR*, 2003.

[146] L. Vacchetti, V. Lepetit, and P. Fua. Stable real-time 3d tracking using online and offline information. *PAMI*, 2004.

[147] Luca Vacchetti, Vincent Lepetit, and Pascal Fua. Combining edge and texture information for real-time accurate 3D camera tracking. In *ISMAR*, pages 48–57. IEEE Computer Society, 2004.

[148] L. Van Gool, T. Moons, and D. Ungureanu. Affine/photometric invariants for planar intensity patters. *ECCV*, 1996.

[149] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proc. CVPR*, 1:511–518, 2001.

[150] Martin Wagner, Asa MacWilliams, Martin Bauer, Gudrun Klinker, Joseph Newman, Thomas Pintaric, and Dieter Schmalstieg. Fundamentals of ubiquitous tracking. *Second International Conference on Pervasive Computing, Hot Spots section*, 2004.

[151] P. Wunsch and G. Hirzinger. Registration of cad-models to images by iterative inverse perspective matching. *13th International Conference on Pattern Recognition*, pages 77–83, August 1996.

[152] Suya You and Ulrich Neumann. Fusion of vision and gyro tracking for robust augmented reality registration. In *VR*, 2001.

[153] Zhengyou Zhang. Iterative point matching for registration of free-form curves. *Technical Report, Inria-Sophia Antipolis*, (1658), March 1992.

[154] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell*, 22(11):1330–1334, 2000.

[155] B. Zitova and J. Flusser. Image registration methods: a survey. *IVC*, 21, October 2003.