Institut für Informatik
der Technischen
Universität München

Sensor-Aided Visual Camera Localization and Tracking for Handheld Augmented Reality

Daniel Kurz

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.-Ing. Darius Burschka
Prüfer der Dissertation:	1.	Univ.-Prof. Gudrun Johanna Klinker, Ph.D.
	2.	Prof. Tom Drummond, Ph.D. Monash University / Australien

Die Dissertation wurde am 30.04.2014 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 27.08.2014 angenommen.

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Erforschung neuartiger Ansätze zur Verbesserung bildbasierter Kameralokalisierungsverfahren für Handheld Augmented Reality Anwendungen. Dabei orientieren sich die entwickelten Verfahren an den Bedürfnissen von Anwendungsszenarien aus der Praxis. Diese können zum einen Beschränkungen mit sich führen, die das generelle Problem der Kameralokalisierung vereinfachen. Auf der anderen Seite sollten solche Verfahren unter beliebigen Kameraposen zuverlässig funktionieren, was eine große Herausforderung darstellt. Wir untersuchen, wie sich dazu Sensoren wie GPS, Kompass und Inertialsensoren eignen, die in aktuellen Mobilgeräten zusammen mit einer Kamera verbaut sind.

Die meisten Objekte und Umgebungen, welche in Handheld Augmented Reality Anwendungen als Referenz zur Bestimmung der Kamerapose dienen, sind statisch und haben eine bekannte Orientierung relativ zur Schwerkraft. Gebäude und Landmarken, welche bei Anwendungen im Freien üblicherweise die Referenz darstellen, haben darüber hinaus eine bekannte und statische globale Position. Wir präsentieren verschiedene Ansätze, solche Kenntnisse über die absolute Orientierung (und Position) von Objekten, zusammen mit Sensoren, welche die absolute Orientierung (und Position) der Kamera messen, sinnvoll zur Unterstützung von Bildverarbeitungsverfahren zu nutzen. Insbesondere zeigen wir, wie sich dadurch sowohl die Invarianz als auch die Eindeutigkeit der Beschreibung visueller Bildmerkmale steigern lässt. Bildbasierte Lokalisierungsverfahren können dadurch schneller und robuster werden. Außerdem resultieren sie in Kameraposen, die deutlich genauer sind, als Posenschätzungen die ausschließlich auf den Messwerten der Sensoren basieren. Darüber hinaus stellen wir ein Verfahren vor, welches die Invarianz von Beschreibungen visueller Merkmale in einem Vorverarbeitungsschritt durch die Synthese mehrerer Ansichten des Objektes steigert. Dieses Verfahren benötigt keine Sensorwerte und kann zur Beschreibung beliebig orientierter Objekte genutzt werden. Es kann allerdings zusätzliche Invarianz gewinnen, wenn es sich um ein statisch orientiertes Objekt handelt, und Messwerte der Schwerkraft existieren. In dieser Arbeit zeigen wir außerdem, wie Template-Tracking Verfahren von Schwerkraftmessungen profitieren können, wenn sie ein horizontales planares Objekt (etwa ein Magazin auf einem Tisch) tracken.

Da verfügbare Testdatensätze für Bildverarbeitungsalgorithmen meist über keine Messwerte von zusätzlichen an der Kamera angebrachten Sensoren verfügen, untersuchen wir in dieser Arbeit weiterhin verschiedene Verfahren, entsprechende Datensätze zu erzeugen. Basierend auf diesen zeigen wir, dass die entwickelten Lokalisierungsverfahren verschiedenen State-of-the-Art Methoden überlegen sind. Das gilt sowohl für das Erkennen und Tracken planarer Objekte, als auch für Kameralokalisierung im Freien. Ein Teil der entwickelten Testdaten wurde der Öffentlichkeit zu Forschungszwecken zur Verfügung gestellt. Insbesondere haben wir den ersten Testdatensatz für Kameralokalisierung im Freien veröffentlicht, welcher aus Sequenzen von über 45.000 Kamerabildern besteht, die mit einem Mobiltelefon aufgenommen wurden. Die Bilder zeigen verschiedene Gebäude von außen und zu jedem Bild sind die Messwerte von Schwerkraft, Kompass-Orientierung, und GPS-Position vorhanden. Darüber hinaus beinhaltet der Datensatz Ground-Truth Informationen über die Kameraposen der Kamerabilder sowie detaillierte Informationen über die Geometrie und Textur der Umgebung.

Abstract

This thesis explores novel approaches to improve visual camera localization and tracking for handheld Augmented Reality (AR). The proposed methods are inspired by the needs of real-world use cases, which on the one hand can introduce constraints that simplify the general problem of camera localization. On the other hand, they require methods that work reliably for any camera viewpoint, which is an important challenge. In order to achieve this, we take advantage of auxiliary sensors, i.e. GPS, an electronic compass and inertial sensors that are attached to the camera in most handheld devices.

In handheld AR applications, the majority of objects or environments serving as reference for visual camera localization and tracking are static, and have a known orientation with respect to gravity. Landmarks and buildings, which usually form the reference in outdoor environments, furthermore have a static and known global location. We present different means to exploit knowledge on the absolute orientation (and location) of real objects and sensor measurements of the absolute orientation (and location) of the camera, to aid low level computer vision algorithms. In particular, we increase both the invariance and the distinctiveness of visual feature descriptors, which can improve visual camera localization in terms of robustness and speed. The resulting camera pose is much more accurate than what could be derived only from the sensor readings. We further present an approach that increases the invariance of such descriptors to changes in viewpoint using an offline pre-processing step. This creates a plurality of synthetic views of the object. While this method does not rely on sensor readings and can also be applied to moving (i.e. non-static) objects, it can additionally benefit from the measured direction of gravity when dealing with objects with a static absolute orientation. Finally, we show how dense template-based tracking of horizontal planar objects, such as a magazine on a table, can benefit from inertial sensor readings.

We further observe a lack of datasets for quantitative evaluation of sensor-aided computer vision methods. Therefore, we present different means to create such datasets, which contain readings of auxiliary sensors associated to the provided camera images. Based on these, we show that our proposed methods outperform state-of-the-art methods for both tracking planar objects, and localizing a handheld camera outdoors in real-time. Some of the test data developed in this work has been made available to the research community. In particular, we describe the first publicly available test database for outdoor handheld camera localization comprising of over 45,000 real camera images of an urban environment, readings of the sensors attached to the camera and ground truth information including the 6DoF camera pose and the geometry and texture of the environment.

Acknowledgments

Many people contributed to this thesis and I would like to thank all of them. First of all, I thank Prof. Gudrun Klinker, Ph.D. for supervising this thesis, for trusting in my work, and for giving me the opportunity to pursue this degree as an external doctoral candidate at the Technische Universität München. I further would like to thank Prof. Tom Drummond, Ph.D. for agreeing to serve as second examiner of this thesis, as well as Prof. Dr. Darius Burschka for chairing the board of examiners.

I'm especially grateful to Dr. Selim Ben Himane for supervising me and the work contributing to this thesis at Metaio. I learned a lot from him about computer vision in particular and research in general. This work would have never been possible without Selim's support and guidance. I also greatly appreciate the support of Peter Meier and Dr. Thomas Alt, who gave me the opportunity to pursue this degree on the job, at Metaio, under excellent conditions.

The content of this thesis is to a great extent based on publications which were co-authored by Dr. Selim Ben Himane, Dr. Sebastian Lieberknecht, Thomas Olszamowski, Peter Meier, Alexander Plopski, and Prof. Gudrun Klinker, Ph.D. In addition to these people, I would like to thank Darko Stanimirović and Marion März for their help on the outdoor ground truth dataset, which is part of this thesis. I'm also grateful to all the other colleagues at Metaio, who built the software framework most evaluations described in this work are based on. Thanks a lot for inspiring discussions, helpfulness, sharing expertise, and for creating a great environment to work in.

I would further like to thank Prof. Dr. Oliver Bimber for introducing me to Augmented Reality and for raising my interest in academic research during my studies at the Bauhaus-Universität Weimar.

Most importantly, I thank my parents Irmgard Schenk-Kurz and Dr. Lothar Kurz for their support and trust in me as well as Anna Keiderling for her love and patience.

Finally I appreciate the fundings which supported the work on this thesis. This work was supported in part by the German Federal Ministry of Education and Research (BMBF, reference number 16SV5745, PASSAge), the German Federal Ministry of Economics and Technology (BMWi, reference number 01MS11020A, CRUMBS), and the European 7th Framework Program, under grant VENTURI (FP7-288238).

Contents

List of Figures

List of Tables

1 Introduction

Nowadays most of the information relevant for humans exists digitally, and the majority of this information has a semantic or spatial relation to entities in the real world, such as objects, people, or places. There is incredibly much digital information available but often the information relevant in a certain context is not at hand when needed. User interfaces providing access to this flood of data should focus on two requirements: *quick* access to *relevant* information, and *presentation* of the information in an *intuitive and natural* way. Augmented Reality is a technology that aims at both of these aspects, and it becomes increasingly important in our lives.

Augmented Reality (AR), as the concept of seamlessly integrating virtual 3D content spatially registered into the real world in real-time [Azum 97], is currently becoming ubiquitous. It recently made its way from research labs to the mass market. The two fundamental enablers for AR moving mainstream are mature computer vision algorithms and affordable off-the-shelf mobile hardware such as camera-equipped mobile phones and tablet PCs. The dense integration of a computer with a display, cameras, different communication interfaces, and a variety of auxiliary sensors make these devices interesting for Augmented Reality. Their pervasiveness makes them an interesting platform for commercial mass market applications.

The World Wide Web provides digital information on pretty much anything we encounter in our everyday lives. For example for each public place or landmark its name and history can be obtained, potentially along with information on what the respective place or landmark used to look like in former times. For every product, such as groceries or electronic devices, there exist digital price comparisons, customer reviews, and potentially user manuals explaining how to maintain the product. Related to a poster promoting a film, digital information such as reviews, trailers, and showtimes in nearby cinemas are available. There is also increasingly more digital information on fellow human beings obtainable from the World Wide Web, including profiles on social networks or private websites.

It is further possible to query information not related to a particular entity, but related to a location and its surrounding. For example at lunch time, information on all restaurants, cafés and cafeterias nearby might be of interest.

Augmented Reality aims at making such digital information available in a natural and intuitive way. For example by simply looking at the Eiffel tower, an Augmented Reality view might automatically determine that we are facing the Eiffel tower by means of object recognition. It may then show general information, such as the height and year of construction of the tower, inside the view, and it might further overlay spatially registered where the restaurants and the observation platform for visitors are located within the tower. For an electronic device, such as a printer, an Augmented Reality view could overlay instructions from a user manual, such as which buttons to press and which lever to pull in order to replace the ink cartridge. By overlaying those instructions spatially registered in 3D onto the view, the user does not need to establish a spatial mapping between the coordinate system of the instructions and the actual printer, which can be challenging with printed manuals. The trailer for a film could be overlaid registered with a poster as if the poster was an interactive display. When looking for a restaurant nearby, a convenient way to visualize the options to a user based on Augmented Reality, shows the location of the restaurants overlaid on the view of the environment. This way the user immediately knows where to go once he or she decided for a restaurant.

There are different means to implement such Augmented Reality views, including approaches based on video projectors that project virtual information right onto the real object or environment. Other approaches use optical see-through displays, which enable displaying virtual contents while the user can still see the real object or environment through the display. Currently the most applications are based on video see-through AR, where an image of the real object or environment is captured by a camera and subsequently shown on a display with overlaid virtual contents representing digital information related to the real object or environment.

This thesis is concerned with handheld Augmented Reality, which is a special case of video-see-through AR where the user holds a device in their hands that combines a camera, a display and a computer. Video-see-through AR requires recognition of the scene and then estimating the pose, i.e. the position and orientation, of the camera with respect to the real scene to be able to superimpose virtual 3D content spatially registered with the camera image on the display. A scene might thereby refer to one or more objects or an environment. The majority of this thesis deals with methods that enable estimating the camera pose, which is often also referred to as (visual) camera localization. Note that the problem of estimating the pose of a camera with respect to an object is the same as to estimate the pose of the object with respect to the camera coordinate system. The only difference is that the second determines the inverse of the transformation the first tries to estimate.

(a) (b) (c)

Figure 1.1: Examples for handheld Augmented Reality: Marker tracking (Courtesy of Vienna University of Technology) (a), planar object tracking (b) at the example of a magazine cover and non-visual tracking based on sensors (c) in junaio's location-based channels.

1.1 Handheld Augmented Reality

With the evolution of mobile phones and tablet computers, video-see-through AR, which previously primarily ran on desktop computers, was adopted to run on mobile and handheld devices. The result is what today is known as handheld AR and probably the most widely used form of Augmented Reality. Early implementations, e.g. [Wagn 03, Wagn 05], placed fiducial markers in the scene to enable camera pose localization and tracking, similarly as shown in figure 1.1 a. These markers were designed to be easily automatically detectable in a digital camera image, which enables 6DoF camera pose estimation relative to the marker, in order to augment a virtual train in this example.

While fiducial markers enable solving the problem of real-time pose estimation of a handheld device's camera, it is not the ideal solution particularly in real-word applications outside a research lab. One problem with markers is that they need to be added to the real environment or object. Furthermore, depending on the application, their position and orientation relative to the real environment or object needs to be determined. This renders the process of making a certain real object ready to be augmented undesirably cumbersome. Another aspect is that markers often do not look very appealing and occupy a significant amount of space.

Other researchers worked on methods to localize and track (planar) textured objects directly, but these approaches were too expensive to run on a mobile device in real-time. Wagner *et al.* [Wagn 08] ported two common approaches to mobile devices and applied a variety of simplifications and modifications to the original methods to achieve real-time pose tracking from natural features on a mobile phone. Until today, most handheld AR applications are based on tracking one or more planar textured objects. Examples include CD cases, advertisements, posters, catalog pages, and magazine covers, as for example shown in figure 1.1 b. Millions of magazines that come with additional Augmented Reality contents are sold every week.

One very nice property of planar objects from the viewpoint of an application developer is, that a reference model of such object, which enables localizing and tracking it, is a simple (fronto-parallel) image of the object.

(a) (b) (c)

Figure 1.2: Handheld AR using visual tracking of non-planar 3D objects and environments. AR games can use a miniature model of the Augmented City as a reference (a) and AR applications for car service and maintenance as based on tracking the real vehicle (b). Outdoor handheld AR application are based on visual tracking of nearby buildings (c).

Trackers for planar objects can be employed for objects that are not exactly planar but close-to-planar up to a certain extent. For example the pages of a catalog are in reality usually slightly bent, i.e. non-planar. However, there is a need for tracking general 3D objects that are not even close to planar. Here, simultaneous localization and mapping (SLAM) systems are commonly used, e.g. [Klei 09], that reconstruct a sparse 3D map of a real object (or environment) whilst tracking it. Saving the established map of an object and loading it again enables applications such as AR games on a miniature version of the Augmented City (see figure 1.2 a). Recently, also bigger objects with more challenging material properties (e.g. specular surfaces) could be used as reference. For example, an AR manual for service and maintenance of a car uses the actual vehicle as reference for camera localization and tracking, cf. figure 1.2 b, to display service and maintenance instructions registered in 3D. In this case, the reference model of the car is based on CAD data.

One other way of determining the camera pose is based on the variety of auxiliary (non-visual) sensors available in recent mobile devices, which are rigidly attached to the camera. These are particularly useful in wide-area outdoor AR applications. As has been done long before the occurrence of smartphones, e.g. [Fein 97], a GPS receiver delivers a coarse absolute location, inertial sensors provide the orientation relative to gravity and an electronic compass gives the missing orientation with respect to North. One major advantage of camera localization based on the non-visual sensors mentioned above, is that no reference model describing the visual appearance of the environment is needed. As a result, such approaches are completely invariant to changes in appearance, e.g. due to illumination or occlusions, and they even work in complete darkness. While the measurements of inertial sensors are quite accurate and reliable, the accuracy of the GPS receivers contained in off-the-shelf mobile phones is in the order of meters. Also both GPS and digital compasses are highly depending on their environment, which makes them unreliable.

Still, the camera pose computed from such sensor readings can be used in mobile Augmented Reality browsers such as junaio [Meta 14c] to overlay the position of distant points of interest (POIs) onto the camera image, cf. figure 1.1 c. Here, a precise spatial registration of virtual and real content is not

crucial. However, the accuracy of such pose estimation is not yet accurate enough for overlaying three-dimensional computer graphics models onto the real environment, such as displayed in figure 1.2 c. On the other hand, approaches to camera localization which are entirely based on the camera image and do not use any markers or fiducials, usually have difficulties dealing with large-scale applications and outdoor environments. This is further elaborated in chapter 3 and the major motivation of this work.

This thesis deals with the question how the readings of the sensors contained in current mobile devices can be exploited to aid visual camera localization and tracking. Since both entirely image-based approaches and approaches based only on non-visual sensors currently are not capable of providing the required precision and robustness for all kinds of real-world AR applications, this work tries to find ways to get the best out of the combination of information from a camera image and the auxiliary sensors of mobile devices, namely GPS, compass, and inertial sensors. As will be explained in the following, the contributions made in this thesis do not only improve camera localization outdoors, but they also improve the robustness of localization and tracking of both planar and general 3D objects for many other use cases.

1.2 Contributions of this Thesis

This work makes a series of contributions in the context of handheld Augmented Reality, particularly in the field of visual camera localization and tracking. We present novel approaches that take advantage of the auxiliary sensors built into handheld devices to aid camera localization and tracking based on visual information, i.e. digital camera images. We further provide comprehensive evaluation of our methods on realistic test data. Approaches to create suited datasets including all required sensor readings and ground truth camera poses are further proposed in this work. In the following, we list all major contributions in order of appearance in this thesis.

- An approach to extend existing evaluation datasets for visual camera localization by synthesizing realistic inertial sensor measurements based on the ground truth poses (section 4.3)
- The first comprehensive dataset to evaluate camera localization outdoors including handheld camera sequences with ground truth poses and the corresponding readings of a GPS receiver, a magnetometer (i.e. electronic compass), and inertial sensors (section 4.4)
- An approach to describe visual features on static objects or environments based on their appearance relative to gravity, which enables distinguishing similar looking structures at different absolute orientations (section 5.1)
- An approach to describe visual features on horizontal surfaces in a way invariant to the angle between the camera's optical axis and the surface normal, which increases invariance to out-of-plane rotations using image rectification based on the measured direction of gravity (section 5.2)

- An offline approach to learn a representative set of feature descriptors to describe an object such that it can be robustly localized from a range of desired viewpoints (section 5.3)
- An approach to provide camera features with an absolute spatial context based on GPS, compass, and inertial sensors, which enables distinguishing similar looking structures at different locations and scales, which are commonly found in urban environments (section 5.4)
- An approach to improve tracking of horizontally oriented templates in terms of convergence and speed based on rectification of the camera image given inertial sensor measurements (section 6.1)
- An approach to enable an edge-based template tracking method to support full 6DoF tracking of horizontal templates instead of supporting only similarity transforms, thanks to inertial sensors (section 6.2)
- An approach that makes virtual augmentations aware of gravity and articulated parts be aligned with the direction of gravity, which may improve the visual realism of augmentations (section 6.3)

In sum, our contributions result in an improved invariance and distinctiveness of visual feature descriptors thanks to the support of auxiliary sensors. We further improved the invariance to out-of-plane rotations of image registration methods that are used for camera pose tracking. For a user, these improvements result in Augmented Reality applications which succeed to initialize tracking a certain object or environment faster and from a broader range of supported camera poses. This makes such AR applications feel more natural. We additionally made contributions to improve the naturalness of Augmented Reality scenes by making virtual objects aware of gravity measured with inertial sensors.

Note that while cameras are sensors as well, throughout this thesis the term *sensors* refers to non-visual (auxiliary) sensors.

1.3 Publications and Collaborations

This thesis is based on work that has been carried out in collaboration with several partners and colleagues. The majority of implementations and experiments done in this thesis are based on Metaio's codebase, which was of great value and provided a solid state-of-the-art framework for video-see-through AR on mobile platforms. This framework is the result of the work of dozens of colleagues at Metaio.

All major contributions of this thesis have been published in the proceedings of international conferences or in journal articles. The following gives an overview of these publications and the collaborators involved therein.

CVPR 2011 Full Paper

DANIEL KURZ AND SELIM BENHIMANE. "**Inertial sensor-aligned visual feature descriptors**". In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. [Kurz 11a]

This paper proposes to align the orientation of visual feature descriptors with the direction of gravity measured with inertial sensors which are attached to the camera. While SELIM BENHIMANE supervised the work and contributed valuable input and scientific expertise, the idea, implementation, and evaluation results of this work come from DANIEL KURZ, who also wrote the majority of the paper.

ISMAR 2011 Full Paper

DANIEL KURZ AND SELIM BENHIMANE. "**Gravity-Aware Handheld Augmented Reality**". In: *Proc. IEEE/ACM Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2011. [Kurz 11b]

This paper investigates how different stages in handheld Augmented Reality applications can benefit from knowing the direction of gravity measured with inertial sensors. It presents approaches to improve the description and matching of feature points, detection and tracking of planar templates, and the visual quality of the rendering of virtual 3D objects by incorporating gravity. SELIM BENHIMANE supervised this work and the writing of this paper, and he carried out the experiments described in the section on "Inertial Sensor-Aided Template Tracking". For the remaining part of the paper, the ideas, implementations and evaluation results originate from DANIEL KURZ. This publication was nominated for the *Best Paper Award.*

TrakMark 2011 Workshop Paper

DANIEL KURZ, SEBASTIAN LIEBERKNECHT, AND SELIM BENHIMANE. "**Benchmarking Inertial Sensor-aided Localization and Tracking Methods**". In: *The 2nd Int. Workshop on AR/MR Registration, Tracking and Benchmarking (TRAKMARK)*, 2011. [Kurz 11c]

This paper investigates means to benchmark methods for camera pose localization and tracking that in addition to a camera image make use of inertial sensor measurements. It presents different simple means to generate benchmarks for inertial sensor-aided localization and tracking methods and most considerably shows how existing datasets, that do not have inertial sensor data, can be exploited. SEBASTIAN LIEBERKNECHT particularly contributed to this work by building and calibrating the setup used in this paper where a mobile phone was rigidly attached to a mechanical measurement arm. Using this setup, ground truth camera poses could be measured for image sequences with corresponding sensor readings captured with the mobile phone. SELIM BENHIMANE supervised the work and DANIEL KURZ contributed the initial idea, implementation and evaluation of the synthesized sensor readings, and wrote the majority of the paper.

Computers & Graphics Journal Article

DANIEL KURZ AND SELIM BENHIMANE. "**Handheld Augmented Reality Involving Gravity Measurements**". In: *Computers & Graphics*, 2012. [Kurz 12a]

This is an invited journal article based on the paper "Gravity-Aware Handheld Augmented Reality". It not only includes the work presented in "Benchmarking Inertial Sensor-aided Localization and Tracking Methods" but it also describes a novel and adaptive approach to Gravity-Rectified Feature Descriptors which – depending on the measured gravity vector – only performs rectification if beneficial. As for the previous papers, the main contributions were made by DANIEL KURZ, while SELIM BENHIMANE supervised this work.

ISMAR 2012 Full Paper

DANIEL KURZ, THOMAS OLSZAMOWSKI, AND SELIM BENHIMANE. "**Representative Feature Descriptor Sets For Robust Handheld Camera Localization**". In: *Proc. IEEE/ACM Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2012. [Kurz 12b]

This paper presents a method to automatically determine a set of feature descriptors that describes an object such that it can be localized under a variety of viewpoints. Based on a set of synthetically generated views, local image features are detected, described and aggregated in a database. The proposed method evaluates matches between these database features to eventually find a set of the most representative descriptors from the database. The main contributions of this work come from both THOMAS OLSZAMOWSKI and DANIEL KURZ. The paper was mainly written by DANIEL KURZ and the work was supervised by SELIM BENHIMANE.

ISMAR 2013 Poster (*Best Poster Award*)

DANIEL KURZ, PETER GEORG MEIER, ALEXANDER PLOPSKI, AND GUDRUN KLINKER. "**An Outdoor Ground Truth Evaluation Dataset for Sensor-Aided Visual Handheld Camera Localization**". In: *Proc. IEEE/ACM Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2013. [Kurz 13]

In this poster, we introduce the first publicly available test dataset for outdoor handheld camera localization comprising of over 45,000 real camera images of an urban environment captured under natural camera motions and different illumination settings. For all these images the dataset not only contains readings of the sensors attached to the camera, but also ground truth information on the geometry and texture of the environment and the full 6DoF ground truth camera pose. The original idea and the majority of work involved in creating this dataset come from DANIEL KURZ. He received valuable help by ALEXANDER PLOPSKI, who was a working student at Metaio GmbH at that time. As such, he was mainly in charge of capturing and labeling the sequences and he performed the

registration of the laser scans with the common world coordinate system. Besides the authors, this work was supported by DARKO STANIMIROVIĆ and MARION MÄRZ who worked on the recovery of the ground truth camera orientation. This work was supervised by PETER GEORG MEIER and GUDRUN KLINKER and it won the *Best Poster Award* at ISMAR 2013.

VISAPP 2014 Full Paper

DANIEL KURZ, PETER GEORG MEIER, ALEXANDER PLOPSKI, AND GUDRUN KLINKER. "**Absolute Spatial Context-Aware Visual Feature Descriptors for Outdoor Handheld Camera Localization**". In: *Proc. International Conference on Computer Vision Theory and Applications (VISAPP)*, 2014. [Kurz 14]

In this paper, we present a framework that enables 6DoF camera localization in outdoor environments by providing visual feature descriptors with an Absolute Spatial Context (ASPAC). These descriptors combine visual information from the image patch around a feature with spatial information, based on a model of the environment and the readings of sensors attached to the camera, such as GPS, accelerometers, and a digital compass. This paper further describes the dataset initially presented in "An Outdoor Ground Truth Evaluation Dataset for Sensor-Aided Visual Handheld Camera Localization". Besides supervising the work, PETER GEORG MEIER contributed to the idea of using information about the approximate depth of a feature to aid its description. ALEXANDER PLOPSKI did contributions to the dataset, as described above, and furthermore developed the process to convert laser scans into reference feature maps for visual camera localization, as used in this paper. The approach described in this paper is mainly based on ideas of DANIEL KURZ who also wrote the majority of the paper. This work was supervised by GUDRUN KLINKER.

Patent Applications

Besides the peer-reviewed academic publications listed above, 5 patent applications were filed in the scope of this work.

1.4 Thesis Overview

This first chapter, which provided an introduction to the present thesis, is followed by chapter 2, which presents fundamentals, state-of-the-art, and related work. The second chapter is meant to make the reader familiar with what is crucial for the comprehension of the following and puts the contributions of this work in context. The next chapter 3 then describes some identified problems and limitations of state-of-the-art approaches to visual feature description, which were the main motivation of this thesis. One approach pursued in this work to tackle the identified problems, is to make use of the auxiliary sensors in modern mobile devices in addition to the camera image.

An objective and quantitative evaluation of the impact of any proposed method requires suited tests and test data. The available datasets for the evaluation of feature descriptors or visual camera localization and tracking do not contain sensor readings. This caused the need to develop evaluation methods and datasets in the scope of this thesis, that allow for evaluation of the proposed and developed approaches, which exploit sensor readings. Chapter 4 presents different means of evaluation and according datasets that were developed and employed in this work.

The main contributions of this thesis are explained in detail in chapter 5. This chapter presents four different approaches to improve visual feature description for different real-world use cases. This includes methods that are particularly suited for vertical surfaces, e.g. posters or screens, in section 5.1. Another approach described in section 5.2 is designed for horizontal planar objects, such as magazines lying on a table. There is also described an approach which is universally applicable to any (planar or non-planar) object at any orientation, as well as a way of detecting and describing visual features which is particularly well-suited for outdoor applications (see section 5.3). The chapter presents evaluation results for all proposed methods and shows how they outperform state-of-the-art approaches.

The focus of this thesis clearly lies on visual feature detection and particularly description, which is a fundamental part of a pipeline for video-see-through AR. There it is commonly used to establish initialization of the camera tracking. However, chapter 6 takes a look at how also the remaining parts of this pipeline, i.e. frame-to-frame tracking and rendering, can benefit particularly from the measured direction of gravity. Chapter 7 finally concludes this thesis with a summary and an outlook to possible future work.

2 Fundamentals and Related Work

This thesis is concerned with (handheld) video-see-through Augmented Reality, which combines aspects of computer vision, computer graphics and human computer interaction. Since the contributions made in this work predominantly are in the field of computer vision, this chapter will explain the relevant fundamentals and related work from a computer vision standpoint.

In this work, we are particularly interested in 6 Degrees-of-Freedom (DoF) camera pose localization and tracking, which aims at determining the 3DoF position and 3DoF orientation of a camera with respect to the coordinate system of a certain object or environment. This chapter describes all basic principles and the steps involved on the way from a camera image to the pose of the camera. We particularly go into the details of those parts that are closely related to the contributions of this thesis and put the latter in context. In addition to a camera image, many of the methods proposed in this work make use of auxiliary sensors contained in mobile devices, which we will first have a closer look at.

2.1 Auxiliary Sensors in Mobile Devices

Current mobile devices, such as mobile phones and tablet computers, usually do not only contain a digital camera, but additionally different auxiliary sensors which are capable of measuring the device's absolute or relative position or orientation. In this work, we make use of the following sensors which are built into nearly all current off-the-shelf mobile devices.

Three-Axis Accelerometer: A three-axis accelerometer is capable of measuring linear acceleration applied to the device in three dimensions. Most notably, on earth, the gravitational force applies a continuous acceleration of $\mathbf{g} \approx 9.81\mathrm{m}/\mathrm{s}^2$ to the device. Therefore the sensor enables measuring the three-dimensional gravity vector, which provides two of the three degrees of the

device's absolute orientation. Any further acceleration, e.g. applied by a user who moves the device, adds to the measured acceleration vector. Accelerometers were initially built into handheld devices to make sure GUI elements remain upright for any device orientation. They are also frequently used in games which are controlled by the orientation of the device. In this work, we use a three-axis accelerometer (in combination with a three-axis gyroscope, which will be explained next) to obtain a precise measurement of the direction of gravity. We then make use of this direction to aid computer vision algorithms when dealing with objects or environments that have a static orientation with respect to gravity.

Three-Axis Gyroscope: A three-axis gyroscope measures angular velocity around three orthogonal axes, i.e. pitch, roll, and yaw, based on inertia. As opposed to accelerometers, it enables measuring rotation about gravity. In combination with a three-axis accelerometer, it provides (in theory) sensing of all 6DoF of camera motion, as it further allows to separate the measured acceleration vector into gravity and the remaining acceleration, often referred to as user acceleration. Throughout this thesis, we never explicitly use readings of a gyroscope, but only benefit from its capability to isolate gravity from the overall acceleration measured with an accelerometer.

Three-Axis Magnetometer: A three-axis magnetometer (also referred to as electronic compass) provides a measurement of the strength and the direction of magnetic fields. When measuring the earth's magnetic field, it is capable of providing the heading of the device with respect to magnetic north, i.e. an absolute orientation about the gravity axis. As also the case for analog compasses, the measured heading can be very inaccurate with errors of over 90 degrees in certain cases. Particularly, the measurement can be heavily affected by additional magnetic fields in the environment. Nevertheless, we use the measured camera heading to improve visual camera localization outdoors in this thesis.

GPS Receiver: The Global Positioning System (GPS) is a space-based satellite navigation system providing an absolute position on earth, usually defined by latitude, longitude, and altitude. The system is based on GPS satellites and therefore requires an unobstructed line of sight to at least four such satellites to function properly. As a result, the precision of the location obtained from GPS can be very inaccurate in (urban) canyons and particularly indoors. Outdoors, and under ideal conditions, the (civilian) GPS receivers built into off-the-shelf mobile devices provide a position with a confidence in the order of several meters. Military-grade GPS receivers provide better accuracy but are out of the scope of this thesis, which deals with approaches based on off-the-shelf hardware. In this work, we use the rough location of the camera obtained from GPS to aid visual camera localization outdoors.

Throughout this thesis, we use the term *inertial sensors* and make use of the *gravity vector* measured by these. Inertial sensors are sensors based on inertia, and therefore include accelerometers and gyroscopes. Since we are only interested in the gravity vector, we do not deal with the raw readings

of these sensors but access the information after processing them on a higher level. Precisely, we obtain the *gravity* via Apple's Core Motion Framework [Appl 14b], which provides a three-dimensional vector describing the gravity acceleration expressed in the coordinate system of the device. The transformation between this coordinate system and the coordinate system of the camera, i.e. the hand-eye calibration, is documented and allows transforming the gravity vector into the camera coordinate system, such that it can be used to aid computer vision methods. In order to obtain the heading and the location of a handheld device as measured with a magnetometer and a GPS receiver, we use Apple's Core Location Framework [Appl 14a].

There are increasingly more auxiliary sensors contained in recent mobile devices, such as ambient light sensors and proximity sensors, which are not used in this work.

2.2 Camera Calibration and Registration

Video-see-through AR is based on the idea of capturing an image of a real environment with a camera and then displaying this image with virtual (3D) contents superimposed spatially registered with the real environment. To achieve this registration, which is crucial for seamless integration of virtual contents, the parameters of the real camera need to be known so they can be applied to a virtual camera which renders the virtual contents.

Precisely what is needed for this purpose is the *camera matrix* $\mathbf{P}$, which is a (3×4) matrix of rank 3, that transforms 3D points $\mathbf{X}$ in the environment to 2D pixel positions in the camera image. Multiplying a 3D point $\mathbf{X} = [x,\ y,\ z,\ 1]^\top$ in homogeneous coordinates with the camera matrix $\mathbf{P}$ results in the projective coordinates $\mathbf{x} = [wu,\ wv,\ w]^\top$ of this point

$$\mathbf{x} = \mathbf{P}\,\mathbf{X} \tag{2.1}$$

from which the image pixel coordinates u and v can be easily computed.

The camera matrix contains both *extrinsic parameters*, i.e. parameters indicative of the position and orientation of the camera, and *intrinsic parameters* which model how points are transformed to pixel coordinates inside the camera. While extrinsic parameters change when moving the camera or the reference environment, the intrinsic parameters usually do not – unless, for example, an optical zoom is changed or the imaging sensor moves with respect to the lens of the camera. Since the cameras built into mobile devices mostly do not allow for changes of the intrinsic parameters, we assume them to be constant. Therefore they can be estimated offline leaving only the estimation of the remaining extrinsic parameters during runtime. When substituting $\mathbf{P}$ by a product of an intrinsic matrix and an extrinsic matrix, the above equation becomes

$$\mathbf{x} = \mathbf{K}\ [\mathbf{R}\ \mathbf{t}]\ \mathbf{X} \tag{2.2}$$

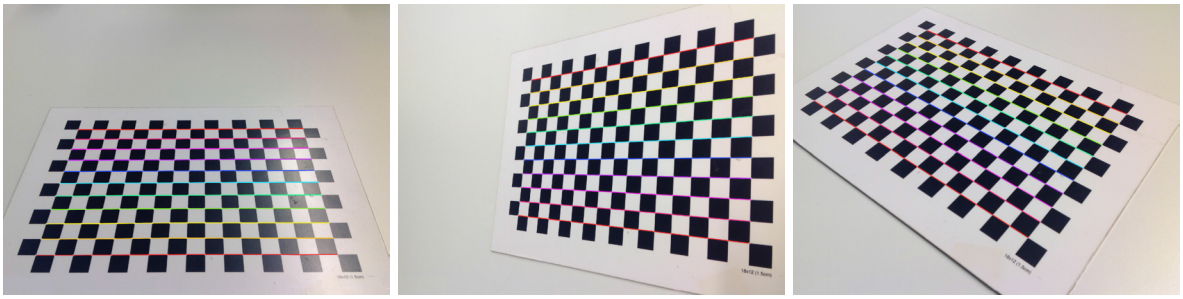


Figure 2.1: Three exemplary images used to calibrate the intrinsic parameters of a camera. The colored lines indicate that the checkerboard was successfully detected.

where $[\mathbf{Rt}]$ is the (3×4) extrinsic matrix of the camera composed of the (3×3) rotation matrix $\mathbf{R}$ and the three-dimensional translation vector $\mathbf{t}$. In our notation $\mathbf{K}$ is the (3×3) intrinsic matrix of the camera defined as

$$\mathbf{K} = \begin{bmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.3}$$

where f_u and f_v represent the focal lengths (in pixels), and u_0 and v_0 are the coordinates of the principal point (in pixels). This model is referred to as pinhole camera model. Note that in this work we assume the skew parameter s to be zero and we do not consider any radial distortions in real-time applications.

2.2.1 Estimating Intrinsic Camera Parameters

The commonly used camera calibration method by Zhang [Zhan 00] calibrates the intrinsic (and extrinsic) parameters of a camera based on at least three images capturing a planar pattern under different orientations. The planar pattern can be arbitrary but is usually chosen such that a homography $\mathbf{H}$, i.e. a (3×3) matrix describing a projective transformation, between the model plane of the pattern and its image in the camera image can be easily determined. Commonly used are checkerboard patterns, such as shown in figure 2.1.

The model plane is defined on $z = 0$, i.e. all points $\mathbf{X}$ on the pattern have the coordinates $[x,\ y,\ 0,\ 1]^\top$. This simplifies the projection of world points onto the image plane as follows. Let $\mathbf{r}_i$ be the i-th column vector of $\mathbf{R}$, then equation 2.2 becomes

$$\begin{aligned} \mathbf{x} &= \mathbf{K}\ [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{r}_3\ \mathbf{t}]\ \ [x,\ y,\ 0,\ 1]^\top && (2.4) \\ &= \mathbf{K}\ [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{t}]\ \ [x,\ y,\ 1]^\top. && (2.5) \end{aligned}$$

The idea of this method now is that the determined homography $\mathbf{H}$ corresponds to the product of the unknown intrinsic camera matrix $\mathbf{K}$ and the remaining part of the unknown extrinsic camera matrix after simplification as shown above and a scale factor λ to account for the universal scale ambiguity.

$$\mathbf{H} = [\mathbf{h}_1\ \mathbf{h}_2\ \mathbf{h}_3] = \lambda\ \mathbf{K}\ [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{t}] \tag{2.6}$$

The knowledge that $\mathbf{r}_1$ and $\mathbf{r}_2$ are orthonormal, leads the following two constraints on the intrinsic parameters $\mathbf{K}$ given the homography $\mathbf{H}$.

$$\mathbf{h}_1^\top \mathbf{K}^{-\top} \mathbf{K}^{-1} \mathbf{h}_2 = 0 \ \text{ and} \tag{2.7}$$

$$\mathbf{h}_1^\top \mathbf{K}^{-\top} \mathbf{K}^{-1} \mathbf{h}_1 = \mathbf{h}_2^\top \mathbf{K}^{-\top} \mathbf{K}^{-1} \mathbf{h}_2 \tag{2.8}$$

By exploiting the fact that $\mathbf{B} = \mathbf{K}^{-\top}\mathbf{K}^{-1}$ is symmetric and can be parametrized as a 6-dimensional vector $\mathbf{b}$ leads to a closed-form solution of $\mathbf{b}$ from which both the intrinsic camera parameters $\mathbf{K}$ and the extrinsic parameters for all images can be computed given at least three images. The result is finally refined using Maximum likelihood estimation based on the Levenberg-Marquardt algorithm [Marq 63]. For all details of this method, the reader is referred to the original paper [Zhan 00], which furthermore describes how radial distortion parameters can be estimated.

As explained earlier, the camera model used in this thesis does not include radial distortions or skew. Throughout this work, we assume that the intrinsic camera parameters $\mathbf{K}$ are known. In fact they were calibrated offline using this method.

2.2.2 Estimating Extrinsic Camera Parameters

Depending on the use case, the extrinsic camera parameters $[\mathbf{Rt}]$ either need to be determined in an arbitrary coordinate system, or with respect to the coordinate system of a known object or environment. The former applies if no a priori model of the scene exists, and the task is to determine the pose (i.e. extrinsic camera parameters) relative to some arbitrary starting pose. Here SLAM (Simultaneous Localization And Mapping) [Davi 07, Klei 07] methods are usually applied that build a map of the environment whilst tracking the camera pose.

This thesis deals with estimating the extrinsic parameters of a camera with respect to a known (i.e. modeled) environment or object in real-time based on an image taken by the camera. This process is also referred to as *pose estimation*, *camera localization* or *camera registration*. This task further needs to be divided into incremental frame-to-frame tracking, i.e. if the camera pose of a previous camera frame is known and expected to be similar to the current pose, and initial registration, when no previous information is available. Incremental tracking is handled in section 2.4, and this thesis mainly deals with the initialization of tracking.

Coarse initialization can be done manually, e.g. [Drum 02, Bles 09], which however is not always desirable in real-world applications. Most automatic approaches to this problem, and particularly those we are interested in here, are based on a set of correspondences between 3D points on the model and corresponding 2D points in an image. We use this for initializing tracking of both planar objects and general 3D objects. How to establish such correspondences will be discussed in section 2.3 and a majority of this thesis deals with exactly this problem.

2.2.2.1 The Perspective-n-Point (PnP) Problem

Let us assume we have a set of 2D-3D correspondences between points in the image and points on the model, then the problem of determining the 3DoF position and 3DoF orientation of the camera from n of such 2D-3D point correspondences is known as the *Perspective-n-Point (PnP)* problem.

At least $n = 3$ correspondences are needed to obtain a finite number of solutions to the problem [Knei 11]. With known intrinsic camera parameters $\mathbf{K}$, 4 or more correspondences generally result in a unique solution. Common approaches to this problem include POSIT [Deme 95], Direct Linear Transformation (DLT) [Hart 04], EPnP [Lepe 09], and various approaches to P3P, e.g. [Knei 11]. There are also approaches that include inertial sensor readings in the pose estimation from point correspondences, e.g. [Li 04]. In the context of equation 2.2 such correspondences are pairs of $\mathbf{x}$ and $\mathbf{X}$ and the problem is to find $[\mathbf{R}\ \mathbf{t}]$ based on these pairs and a known $\mathbf{K}$.

If the model points are known to be co-planar, a projective transformation in form of a (3×3) homography matrix $\mathbf{H}$ can be found instead of estimating the 6DoF pose [Hart 04]. Homographies can be estimated based on point correspondences or based on dense image registration for planar objects, as discussed in section 2.4. Having $\mathbf{K}$ and a homography $\mathbf{H}$ mapping a planar object in the camera image to the coordinate system of a reference model enables computing the 6DoF pose.

The correspondence-based approaches discussed above assume all input correspondences to be correct. In reality, particularly if these correspondences are obtained from natural features in the scene, as we will discuss in section 2.3, this is usually not the case. Instead these methods provide many putative correspondences and it is not known which of these are correct correspondences, i.e. inliers, and which are wrong, i.e. outliers. Outliers may for example result from partial occlusions of the object in the camera image or from matches of features in background clutter. In order to estimate a camera pose based on such kind of data, methods are required which are *robust* to outlier correspondences.

2.2.2.2 Robust Pose Estimation

Robustification in the following context of pose estimation refers to being able to deal with outliers, i.e. wrong correspondences. A widely used approach to this problem is RANdom SAmple Consensus (RANSAC) [Fisc 81]. This iterative approach randomly samples a small subset of m correspondences (where m would usually be 3 or 4 in this context) which are considered hypothetical inliers. This

hypothesis is then tested by estimating the pose from the sampled subset of correspondences using any of the methods discussed above and afterwards testing for all remaining correspondences how well they agree with the found model, i.e. pose. This can for example be done by projecting all 3D points into the image based on the estimated pose and known intrinsic camera parameters, and then declare those correspondences as inliers, i.e. agreeing with the model, where the projected position and the 2D position of the correspondence differ by less than a threshold. This approach is iteratively repeated with different sampled subsets and finally the best fitting model, i.e. pose, e.g. with the most inliers, is kept and potentially further refined.

The Progressive Sample Consensus (PROSAC) algorithm [Chum 05] further improves this approach by considering the similarity of correspondences, which results from the matching stage. The assumption of this method is that correspondences with a higher similarity between the appearance in the model and in the image, are more likely to be correct, i.e. inliers. Instead of randomly sampling subsets of correspondences, PROSAC consequently draws samples from a progressively larger set of correspondences with the highest similarity. As a result, a correct pose is usually found in significantly less iterations, which makes the pose estimation process significantly faster.

2.3 Establishing Point Correspondences

Many applications in the field of computer vision require finding corresponding points in two or more images of the same scene or object under varying viewpoints. Examples include wide baseline stereo matching for 3D reconstruction, object recognition, and – as discussed above and most important for this thesis – camera pose estimation.

A common way to gain such correspondences is to first detect features, such as corners or blobs, from the individual images that are expected to have a high repeatability. That is, feature detection methods are designed such that they ideally detect features corresponding to the same physical points in a scene when imaged from different viewpoints. In order to establish correspondences between features of different images, the features need to be described such that they can be compared in terms of their similarity, and eventually be matched.

Changes in camera viewpoint, which result in changes in scale and orientation of imaged features in the camera image, should ideally not affect the feature description. To achieve invariance to such transformations, many approaches perform a normalization of the image patch around detected features before describing them to factor out the effect of such transformations. A feature description method then essentially is a function that maps a (normalized) image patch to a vector, which is referred to as the *feature descriptor* or sometimes *feature vector*. Note that also the feature description method is often referred to as feature descriptor. Once a descriptor has been computed for a feature, it enables its comparison and therefore its matching with a corresponding feature in another image, e.g. by finding the nearest neighbor of a feature vector.

The two main requirements for a good descriptor are distinctiveness, i.e. feature points corresponding to two different physical points result in different descriptors, and invariance to changes in viewpoint, illumination and image noise. This is to ensure that features corresponding to the same physical point in different images result in close descriptors with respect to a certain similarity measure.

Another common approach to match image features is based on feature classifiers. In these approaches finding the corresponding database feature for a given query feature is formulated as a classification problem, where every database feature is a class, and the classifier determines the class with the highest probability for the query feature. This will be further explained in section 2.3.5.

A variety of feature detectors and feature descriptors exists, and in theory, detectors can be arbitrarily combined with descriptors. Heinly *et al.* [Hein 12] evaluate different combinations of feature detectors and descriptors and come to the conclusion that certain combinations match better than others. Mikolajczyk and Schmid [Miko 05b], however, come to the result that the ranking of different feature descriptors is mostly independent of the used feature detectors.

Note that the terms (local image) feature, interest point, and keypoint are used interchangeably in literature and throughout this thesis. Furthermore this work focuses on point features, while there are approaches dealing with different features of an image, for example edges.

2.3.1 (Point) Feature Detection

The motivation to detect features in an image and to limit matching to only these features instead of matching all possible points in an image is firstly to reduce computational cost because significantly less points need to be described and matched using such approach. Another advantage of such sparse approaches over dense approaches that consider all pixels is, that sparse approaches may better deal with partial occlusions of an object or environment to recognize. But feature detection also aims at describing and matching only those parts of an image that most likely are distinct and invariant to changes in illumination and viewpoint. The same scene or object viewed from two different positions should ideally result in interest points that correspond to the same physical points. Therefore, most feature detectors focus on salient image features such as corners and blobs. As opposed to edges, point features provide a two-dimensional constraint.

2.3.1.1 Corner Detectors

The detection of corners in an image has been widely studied. In the following we discuss approaches to corner detection that are commonly used, including those that we make use of in this thesis. Any method to detect corners first needs to consider how to define a corner and we will see that different approaches use different definitions.

In the early approach of Moravec [Mora 80], a corner is a pixel in an image for which there are large intensity variations in every direction. For each pixel, this approach computes the average differ-

ence (often referred to as autocorrelation) between a patch centered around the pixel and the patches centered around four shifted pixel positions. If the pixel in question is located in a region with (approximately) uniform intensity, the differences resulting from all patch comparisons will be small. If the pixel is located on an edge, the differences will be small for shifts along the edge and large for shifts perpendicular to the edge. For pixels located on a corner, the computed differences will be large for all shifts. Therefore Moravec uses the minimum over all differences as an indicator of the *cornerness* of a pixel. Finally his approach detects those pixels as corners which have a local maximum in this cornerness above a certain threshold, which is referred to as non-maximum suppression.

The commonly used Harris corner detector [Harr 88] (sometimes also called Plessey corner detector) builds upon this work and overcomes some of its limitations. Instead of using a discrete set of four shifts, this approach considers all possible small shifts in a circular window around a pixel, which makes the detector isotropic. In fact, they use an approximation when estimating the autocorrelation based on first order image derivatives. Instead of using the minimum over all differences as a cornerness response, which often results in edges being wrongly detected as corners, this approach considers the variation of differences by looking at the two eigenvalues of the structure tensor. These eigenvalues are proportional to the principal curvatures of the local autocorrelation function. Only if both eigenvalues are large, this means that the autocorrelation function is sharply peaked, which means the pixel is a corner. Instead of actually computing the eigenvalues, Harris and Stephens define a cornerness response function based on the determinant and the trace of the structure tensor, which correspond to the product and the sum of the eigenvalues, and their computation is less expensive. Eventually, again non-maximum suppression is performed.

Similarly, Shi and Tomasi [Shi 94] come to the conclusion that for *good features to track*, both eigenvalues of the structure tensor must be large. They compute the eigenvalues and accept only those pixels as corners where the smaller of the two eigenvalues is above a defined threshold. This approach, as well as the other two approaches above ([Mora 80, Harr 88]) are capable of detecting both edges and corners. When configured to detect corners, i.e. points with two dominant gradient directions in a region around them, they do in fact also detect blobs, i.e. local extrema in intensity.

A different approach to define the properties of a corner has been pursued by the SUSAN (Smallest Univalue Segment Assimilating Nucleus) corner detector [Smit 95]. In a circular region around a pixel, this approach counts the number of pixels with a similar intensity as the center pixel. For pixels located in approximately uniform image regions, this is the case for a majority of the tested pixels. At an edge, the ratio of pixels with a similar intensity decreases until close to 50%, whereas for pixels at a corner, this ratio is even lower. In fact, this approach assigns all pixels where the ratio of the tested pixels with similar intensity is below 50% a non-zero cornerness score based on the ratio. Subsequently, non-maximum suppression is performed to find corners. A further test is carried out to exclude blobs (e.g. a single bright pixel surrounded by dark pixels) or thin lines from the detection and to only keep corners. To this end, the center of gravity over all pixels with a similar intensity in the circular region is computed and it is requested that it is sufficiently far away from the center

pixel and that all pixels connecting the center of gravity and the center pixel are within those pixels with a similar intensity. The SUSAN corner detector outperforms the approaches discussed above particularly in terms of a more accurate localization accuracy, but it also runs faster than the other approaches.

Another corner detector, which was specifically designed for efficiency, is the FAST (Features from Accelerated Segment Test) feature detector [Rost 05]. The condition for a pixel to be detected as a corner is simply that on a Bresenham circle of radius 3 centered around the pixel, the intensities of at least 12 contiguous pixels of the 16 pixels on the circle are all above or below the intensity of the center pixel by a threshold. This initial version already included an acceleration approach by first checking the intensities of four pixels on the circle at 90 degrees steps. Because a corner according to the condition stated above requires that three of these four pixels have an intensity above or below the intensity of the center pixel by the threshold, the method can skip further tests for pixels not satisfying this test. Here again non-maximum suppression is performed. According to their experiments, this implementation of FAST is more than 4 times faster than SUSAN.

The same authors later showed how this feature detector could be further speeded up using machine learning [Rost 06]. Based on a set of training images, their approach first computes for each pixel in each training image if it is a corner or not using a slow implementation of the segment test criterion where simply all 16 location on the circle around a candidate pixel are tested. They further compute for each pixel the state of all 16 pixels on the circle, i.e. if their intensity is darker, similar, or brighter. Given this information, they select the pixel on the circle which yields the most information about whether the candidate pixel is a corner or not based on entropy. After having selected the most informative pixel on the circle, this procedure is recursively repeated on the three resulting subsets, i.e. the candidate pixels for which the selected pixel is darker, similar, or brighter than the candidate pixel. This finally results in a decision tree that correctly classifies all corners seen in the training set and therefore is expected to generally perform similarly to the original FAST corner detector. This implementation is up to twice as fast as the original implementation because less pixel intensity comparisons are needed, which allows for real-time processing even on mobile devices, e.g. [Wagn 08, Klei 09]. More importantly, this approach allows for an efficient implementation of the segment test requiring only 9 (instead of 12) contiguous pixels to fulfill the condition. This detector FAST 9 outperforms FAST 12 particularly in terms of repeatability.

In a further stage called FAST-ER [Rost 10], again machine learning is applied to learn a decision tree for corner detection. Based on three images with known transformations, the optimization of the decision tree evaluates a cost function which demands repeatability of the detector. This approach further evaluates the intensities on a thicker circle around a candidate pixel, i.e. more possible pixels. It outperforms FAST in terms of repeatability with little loss of efficiency.

FAST and FAST-ER use training images to learn decision trees, which therefore perform particularly well for scene structures present in the training images. AGAST (Adaptive and Generic Corner

Detection Based on the Accelerated Segment Test) [Mair 10] tries to find an optimal decision tree implementing the accelerated segment test of FAST in an extended configuration space that evaluates a single question per tree node whether a pixel is darker, not darker, similar, not brighter, or brighter in intensity than the center pixel. The question might further involve the answer of the preceding question. They build two binary decision trees – one for textured image regions and one for homogeneous image regions – and can switch between the two at no extra cost based on the tree node they end up in. This approach further speeds up the detector while not affecting its repeatability.

While [Mair 10] evaluates using different mask sizes for feature detection, the approaches described so far are per se not scale-invariant. A common approach to add scale invariance to these corner detectors is to run them on the levels of image pyramids, which represent an image at different discrete scales. BRISK [Leut 11] uses the AGAST detector on pyramid levels and determines a continuous feature scale which might be in-between the pyramid levels. This is similar to scale-invariant feature detectors such as SIFT, which will be discussed next.

2.3.1.2 Blob Detectors

A common class of feature detectors aims at detecting blobs, i.e. regions in an image which differ in intensity from the area around the blob. These approaches are computationally more expensive than corner detectors, and therefore not yet suitable for real-time approaches to camera localization on handheld devices. However, well known feature description methods, such as SIFT [Lowe 04] and SURF [Bay 08], which are used for evaluation in this thesis, are based on blob detection.

SIFT (Scale-Invariant Feature Transform) [Lowe 04] aims at detecting locations in an image which are invariant to scale. An image to detect features in is successively smoothed with a Gaussian kernel and sampled. Subtracting two successive smoothed images that were convolved with Gaussians of different kernel sizes results in Difference-of-Gaussian (DoG) images. The feature detection is based on finding local 3D extrema in a scale-space pyramid of DoG images. Thereby the 3D location of an extrema results in both the 2D position of a feature in the image as well as a continuous scale of the feature. Even though there exist optimized implementations on GPUs, e.g. [Heym 07], SIFT is currently not efficient enough for real-time processing on handheld devices.

The feature detector in SURF [Bay 08] is based on box filters that approximate second order Gaussian derivatives, and can be evaluated very fast using integral images, to approximate a Determinant of Hessian (DoH) blob detector. Instead of using image pyramids for scale space analysis, this approach scales the size of the box filter which can be done at no extra cost using integral images. For localizing features, 3D extrema of the approximated DoH are found using non-maxima suppression. The third dimension thereby is the size of the box filter applied. Similarly as for SIFT, the 3D extrema locations result in a 2D position of the feature in the image as well as an interpolated (continuous) scale.

2.3.2 Feature Normalization

As discussed above, after detection, features are commonly assigned a scale and an orientation to normalize the feature before description. The normalization can either be implemented by warping the image patch around a feature to a normalized (canonical) coordinate system, or by transforming the coordinates of the descriptor method accordingly. The estimation of the normalization parameters may be (partially) obtained in the course of feature detection, or they may be determined in a separate step afterwards. Note that the invariance discussed below affects the descriptor and is not necessarily linked to the invariance of the feature detector.

2.3.2.1 Scale Invariance

The scale of a feature is usually determined by the feature detector. From the discussion of both SIFT [Lowe 04] and SURF [Bay 08] we can see that a continuous scale is determined because feature detection takes place in scale space. Similarly the feature detector of BRISK [Leut 11] provides a continuous scale. Other approaches provide discrete scales when detecting features on discrete pyramid levels, e.g. [Wagn 08, Klei 09]. In order to avoid the potentially expensive step of building image pyramids and detecting features at each level in every frame during runtime, Wagner [Wagn 08] performs feature detection at different scales only offline on reference images. For each camera image, in which features are detected only at one scale, this approach assumes that one of the pyramid levels used offline is similar to the current scale.

All approaches discussed above aim at assigning a scale to features, such that they can be matched from different distances. When an object doubles its distance to the camera, and therefore appears at half the size, the scale of the feature corresponding to each point on this object should ideally half. Note that this scale does not have a known relation to physical (or absolute) scale, e.g. millimeters. Particularly, the scale invariance discussed above cannot distinguish between scale (in the camera image) as a result of distance to the camera and the physical scale of a feature.

When absolute depth information is available, e.g. from combined range-intensity data, then it is possible to assign features a physical scale or to detect features at desired physical scales [Smit 12]. Additionally, this approach may prevent confusions (i.e. mismatches) between similar looking features at different physical scales as discussed in section 3.1.1 and shown in figure 3.1. However, it is important to keep in mind that this approach to scale invariance depends on range data.

2.3.2.2 Rotation Invariance by Orientation Assignment

Invariance to in-plane rotation of image features is often achieved by orientation assignment. Based on image intensities in a patch around a feature, one or more repeatable orientations in the range $[0°, 360°)$ are determined for each feature.

SIFT [Lowe 04] computes the orientation from local image gradients in a patch around the feature. These orientations are then accumulated in a histogram weighted by the gradient magnitude and the distance to the feature with a Gaussian mapping. The orientation corresponding to the highest peak in the histogram and up to three additional orientations with at least 80% of the highest peak, are eventually used as feature orientation. If more than one orientation is assigned to a feature, this means that the feature description stage will create multiple descriptors for this feature – one for each orientation.

The orientation assignment in SURF [Bay 08] first convolves the image with two orthogonal first-order Haar wavelets scaled with the feature's scale in a circular region around the feature position. The responses are then interpreted as gradient orientation vectors and the dominant orientation is found by computing the sum of all orientation vectors within a sliding orientation window. The longest of the summed up vectors finally defines the feature orientation.

The smoothed local gradient at a feature position is used in [Brow 05] to determine the orientation of a feature. An efficient implementation of orientation assignment which follows a similar idea has been proposed by Taylor and Drummond [Tayl 11]. The difference in intensity of opposite pixels on a circle around the feature position is determined resulting in eight differences with an orientation based on the pixel position. These are then interpreted as two-dimensional vectors with a magnitude according to the intensity difference and concatenating them results in a single vector which defines the feature orientation.

ORB [Rubl 11] uses the position of the centroid of intensities in a patch around the feature, relative to the feature, to determine the feature orientation. This was initially proposed in [Rosi 99] and is very efficient to compute. There are more approaches to determine a feature orientation based on image intensities around the feature, which are proposed and discussed in [Rosi 99] and [Gaug 11b].

Another approach to remain invariant to rotation is to assign all possible orientations to a feature and to compute descriptors for all orientations as proposed for BRIEF [Calo 12]. This is particularly applicable when feature description and matching are very efficient, which is the case for this binary descriptor.

Bay *et al.* [Bay 08] showed that omitting the orientation normalization for Upright-SURF, outperforms regular (rotation-invariant) SURF descriptors in terms of discriminative power. However, this approach forces the user to keep the camera in an upright orientation, which limits the field of possible applications.

If the feature description method itself is invariant to in-plane rotation, as for example in [Schm 97], then the step of orientation assignment is not needed and can be skipped.

In section 2.3.8 we show how exploiting the direction of gravity measured with inertial sensors allows for orientation assignment which is not only computationally efficient but also provides an increased discriminative power without constraining the orientation of the camera.

2.3.2.3 Invariance to Out-Of-Plane Rotations

Invariance to translation (as a result of feature detection), uniform scale, and in-plane orientation, as discussed above, results in invariance to similarity transformations. The remaining degrees of freedom involved in a perspective transform are often expected to be taken care of by the feature description method. However, there exist approaches that aim at detecting features and normalize them with an affine transformation, e.g. Harris Affine [Miko 04]. A good overview of affine feature (or region) detectors is given in [Miko 05a]. Due to their computational complexity, these detectors are usually not used for real-time applications on mobile devices.

Invariance to out-of-plane rotations can further be achieved when provided with the depth of the camera pixels, e.g. using an RGB-D camera. The 3D normal vector of a feature can be determined to create a viewpoint-invariant patch [Wu 08] of the feature for subsequent description.

Similarly to BRIEF's [Calo 12] approach to rotation invariance, an image retrieval method [Chen 09] deals with out-of-plane rotations by using reference images from canonical viewpoints covering the whole range of out-of-plane rotations to support. Based on feature descriptors of those images, they then create a specific scalable vocabulary tree for every canonical viewpoint. The assumption here is, that for any viewpoint, there is a canonical viewpoint sufficiently close to it (with a similar out-of-plane rotation), such that invariance to out-of-plane rotations is not needed anymore. Note that with this approach the memory required for the reference description scales linearly with the number of canonical viewpoints. Similarly feature descriptors with little invariance to changes in viewpoint can be created for many different viewpoints in an offline stage if the descriptor is memory-efficient [Tayl 09].

Affine SIFT (ASIFT) [More 09] achieves full affine invariance by creating affine warps of an image for varying camera axis orientations and matching between such warps of two different images by means of SIFT. A two-resolution approach that first matches on downsampled images reduces the complexity to about twice that of SIFT, making this approach infeasible for real-time applications on mobile devices.

In section 2.3.8 we argue that our proposed approaches to out-of-plane invariance of feature descriptors are superior to the methods explained above because they neither require expensive image-based estimation of affine transforms per feature, nor do they require a depth-sensing camera and they do not increase the memory footprint of the reference description of an object.

2.3.3 Feature Description

Once a feature has been detected and normalized, it can be described by a feature descriptor based on the normalized image patch around the feature. According to the classification of [Hein 12], feature descriptors include patch-based descriptors, real-valued descriptors, binarized descriptors, and binary descriptors.

2.3.3.1 Patch-based Feature Descriptors

The most straightforward approach to describe a feature is to use the (normalized) image patch around the feature as description. The similarity of two features can then be evaluated by computing a similarity between the two patches. Common metrics used in this context are the sum of absolute differences (SAD), sum of squared differences (SSD), normalized cross correlation (NCC), or mutual information (MI). Patch-based feature descriptors are efficient to compute. However, they require a considerable amount of memory to be stored when using larger patches and computing similarities can be expensive in contrast to the descriptors described below. Most importantly, patch-based feature descriptors do not provide invariance to inaccuracies in the feature position or the assigned orientation and scale.

2.3.3.2 Real-valued Feature Descriptors

A general approach which we use throughout this thesis creates real-valued vectors as feature descriptors. While the (normalized) image patched itself can also be seen as a vector when e.g. concatenating the rows, the methods discussed in the following usually aim at creating a feature descriptor which has a lower dimension than the number of pixels in the underlying image patch, and most importantly, they aim at providing better invariance to inaccuracies in normalization than patch based feature descriptors.

An overview of different real-valued feature descriptors is given in [Miko 05b] and the majority of these descriptors are based on histograms of salient structures in the image patch, such as gradients. We briefly explain the concepts of common real-valued feature descriptors below.

The descriptor of SIFT [Lowe 04] computes the gradient magnitude and orientation for sample points in the normalized image patch and weights them according to their position with a Gaussian kernel centered around the feature. The descriptor vector is finally built by concatenating histograms of gradient orientation for different square subregions in the normalized patch. Using (4×4) subregions with 8 orientation bins each, this results in a 128-dimensional feature vector. To account for changes in contrast, the feature vector is normalized to unit-length which provides true invariance to affine changes in the image intensities. To also handle non-linear illumination changes, all large gradient magnitudes in the feature vector are capped to 0.2 followed by another normalization.

The SURF [Bay 08] descriptor is based on a similar concept as SIFT, but uses different means to improve efficiency. The normalized image patch is also divided in (4×4) subregions and in each subregion, the responses of a horizontal and a vertical Haar wavelet are computed for a set of uniformly sampled points. Again the responses are weighted with a Gaussian centered around the feature. The feature descriptor vector is computed by concatenating for each of the 16 subregions the summed up horizontal and vertical wavelet responses as well as the summed up absolute responses, resulting in a 64-dimensional vector as descriptor. This vector is eventually normalized to achieve invariance to affine intensity transforms such as changes in brightness or contrast.

There are more feature descriptors based on histograms of gradients, for example HOG (Histograms of Oriented Gradients) [Dala 05], GLOH (Gradient Location and Orientation Histogram) [Miko 05b], and MOPS (Multi-scale Oriented Patches) [Brow 05]. Instead of building a histogram of gradients, CS-LBP [Heik 09] is based on a histograms of local binary patterns (LBP), which are more efficient to compute and provide superior matching performance in their evaluation.

Rather than tuning the spatial layout and parameters of (real-valued) feature descriptor methods manually, Winder and Brown [Wind 07] break up the description method into stages based on which a generic descriptor can be built. They then learn optimal parameters for the individual processing stages based on a ground truth dataset of images with accurate match and non-match information. This results in parameterizations which outperform handcrafted descriptors such as SIFT in terms of matching accuracy.

A completely different approach to feature description is pursued by PCA SIFT [Yan 04]. Based on normalized image patches of features detected in a set of training images, they compute a vector representation by concatenating the horizontal and vertical gradient maps for each patch, to build an eigenspace. Principal Components Analysis (PCA) [Joll 05] is applied to the covariance matrix of these vectors and the resulting top n eigenvectors are used to build a projection matrix. This projection matrix enables projecting an gradient image vector of an image patch to be efficiently projected to a compact feature vector of length n. Note that the image patches in this case are not arbitrary but have in common that they were detected as features by SIFT's detector and they are normalized with respect to gradient orientation and scale. This explains why a compact representation of the patches is still distinctive. According to the results in the paper, PCA SIFT is more distinctive and more robust than SIFT using a projection onto feature vectors with only 20 dimensions.

A low-dimensional descriptor is more memory-efficient and enables faster matching. There are other approaches to dimension reduction of real-valued descriptors, e.g. using PCA on descriptors (not gradient patches) extracted from training images [Vale 12] or based on Linear Discriminant Projections [Cai 11].

Other approaches, such as Phony-SIFT [Wagn 08], which is a simplification of the SIFT descriptor that performs at frame rate on mobile phones, use quantized values (bytes) instead of floating point precision and less histogram bins to reduce the feature vector dimension and memory footprint.

2.3.3.3 Binarized Feature Descriptors

The class of binarized feature descriptors aims at converting real-valued feature descriptors into binary descriptors, which can be matched very efficiently based on their Hamming distance.

An example for such kind of approaches is LDAHash [Stre 12], which uses Linear Discriminant Analysis (LDA) to find a projection matrix from the n-dimensional space of a real-valued feature descriptor, e.g. SIFT, to an m-dimensional space. Thereby m is significantly smaller than n. They

further find an m-dimensional vector to be added to the projected vector, such that binarization of the vector by computing the sign of each element results in a binary vector representation in which the descriptor vectors of features corresponding to the same physical point result in binary descriptors with a small Hamming distance while descriptors corresponding to different physical points result in distant binary descriptors.

While the binarization of feature descriptors allows for efficient matching, the computation of the descriptor itself is expensive when based on descriptors such as SIFT.

2.3.3.4 Binary Feature Descriptors

A class of local feature descriptors which recently became popular is that of binary descriptors. These descriptors are efficient to compute since they are based on simple comparisons of pixel intensities in a patch around a feature and concatenate the results of the tests to form a binary descriptor vector. These vectors have a relatively small memory footprint and they enable efficient distance computation and matching analogously as for binarized descriptors, discussed above.

The first popular binary descriptor was BRIEF (Binary Robust Independent Elementary Features) [Calo 12], which uses a randomized but fixed set of 128, 256, or 512 pixel intensity comparisons centered around the feature location in a Gaussian distribution. Each intensity comparison results in the binary value of one dimension of the feature descriptor vector. Because the detector for BRIEF does not operate in scale space and no orientation assignment is performed, BRIEF is not invariant to changes in scale nor orientation.

The missing rotation invariance of BRIEF was motivation for ORB (Oriented FAST and Rotated BRIEF) [Rubl 11], which adds orientation assignment by means of an intensity centroid as discusses above. ORB also uses a different sampling layout for intensity comparison tests as a result of a machine learning algorithm which aims at maximizing the descriptor's variance while minimizing the correlation under changes in orientation.

Another more advanced binary descriptor is BRISK (Binary Robust Invariant Scalable Keypoints) [Leut 11] which provides invariance to changes in both scale and orientation. The approach uses the AGAST detector on pyramid levels and determines a continuous feature scale which might be in-between the pyramid levels. In contrast to the binary feature descriptors discussed above, the sample positions of BRIEF are uniformly sampled on circles of different radius around the feature position. Instead of sampling the intensity of a single pixel, this approach uses the image intensities in a Gaussian distribution around each sample point for binary tests.

A comparison of the binary descriptors discussed here can be found in [Hein 12].

2.3.3.5 Feature Description Based on Multiple Observations

A memory-efficient and fast approach to feature matching based on learning the appearance of features [Tayl 09] uses simple representations based on quantized histograms of pixel intensities which are only invariant to small changes in viewpoint. Larger viewpoint variations are supported by learning independent representations from different viewpoints, whereby for all these viewpoints synthetic images are generated by means of warping. Based on the empirical intensity distributions for samples around a feature observed in the learning stage, this approach aims at identifying intensity bins which are rarely observed. The binary descriptor of reference features carries a 1 for such rarely observed intensity bins while all other bits are 0. This is then used to define a dissimilarity with features from a live frame for which only the observed bins carry a 1 in the descriptor. The dissimilarity can be computed very efficiently with bitwise operations. While this approach is very efficient during runtime and allows for robust feature matching in $2.3\mu s$, it requires an offline training phase which takes about an hour per target and thereby limits the range of possible applications.

Different methods aim at camera localization based on structure from motion 3D point clouds. For every 3D point there usually exist multiple descriptors resulting from the images used for reconstruction of this point. As these descriptors are highly redundant, they can either be clustered [Irsc 09] or averaged [Li 10] to reduce the number of reference descriptors. To further reduce the dataset and speed up matching, greedy approaches to an NP-hard set cover problem are used. If an image retrieval step is used to initialize the localization [Irsc 09] the approach tries to find the minimal subset of views, that covers the 3D point cloud. Another approach aims at finding a minimal subset of points that covers all camera images [Li 10].

This thesis proposes a novel approach which is also based on synthetic views of an object to create feature descriptors for. As will be discussed in section 2.3.8 our approach finds the most representative subset of descriptors based on matches between the descriptors of different views.

Learning the appearance of a feature under different viewpoints is also the basis of feature classifiers, which will be explained in section 2.3.5.

2.3.4 Matching Feature Descriptors

Given two sets of features which partially correspond to the same physical points, the stage of feature matching aims at determining for each feature from the first set if there is a feature corresponding to the same physical point in the second set and if so, which one it is.

Using nearest neighbor search based on the descriptor distance leads to a potential match for each feature. The distance measure to be used depends on the descriptor. For real-valued descriptors usually the (squared) Euclidean distance is computed while binary descriptors are compared with respect to Hamming distance. There are different approaches to determine if the nearest neighbor actually is a good match and therefore likely a feature corresponding to the same physical point. A good indicator is

the ratio between the distance to the closest descriptor and the distance to the second closest descriptor as proposed in [Lowe 04], which needs to be below a threshold for a match to be considered good and kept for further processing.

For large sets of features, approximate nearest neighbor (ANN) search, e.g. Best Bin First [Beis 97] or locality-sensitive hashing [Gion 99, Rubl 11], can significantly speed up matching without sacrificing too much matching precision.

Another approach to speed up feature matching is indexing of features e.g. by their sign [Rost 05, Bay 08]. As will be discussed in section 2.3.6, additional distinctive feature attributes, such as an index, can be used to further speed up matching by constraining the features to match against.

2.3.5 Feature Matching using Feature Classifiers

Feature classifiers also aim to identify for a given image feature the corresponding reference feature in a database. This can be formulated as a classification problem, where every reference feature is a class, and the classifier determines the class with the highest probability for a given current feature. An offline training phase is required, where the classifier is trained with different possible appearances of a feature, usually gained by randomly warped patches.

Randomized Trees [Lepe 06] use these to estimate the probabilities over all classes for every leaf node, while the inner nodes contain binary decisions based on image intensity comparisons. Ferns [Ozuy 07] are also based on binary intensity difference tests. However, instead of a tree, they use a non-hierarchical structure which provides better scalability. While the previous methods aim at training all possible visual appearances of a feature as a single class, which provides invariance to changes in the viewpoint, other approaches [Hint 11] train the appearances from different viewpoints as different classes. Thereby, the classifier is not only able to determine the identity of a feature, but also its pose.

Thanks to the training stage provided with different appearances of a feature, classifiers in general provide a good invariance to out-of-plane rotations. However, the probabilities require a lot of memory, which makes them unfeasible for a large number of features on mobile devices. On mobile phones, simplified versions of FERNS have been used in real-time applications to localize the camera with respect to planar objects [Wagn 08]. It is important to keep in mind that these approaches require a time-consuming offline learning stage for new features which limits their application field to use cases where the visual appearance of features is known long before they can be matched.

2.3.6 Additional Distinctive Feature Attributes

Besides the descriptor (i.e. feature vector) or classifiers there are different approaches to store more information with features, e.g. an index. These can be used to add distinctiveness and thereby both improve and speed-up feature matching.

The authors of FAST [Rost 05] propose to assign features for which the center is darker than the segment on the surrounding circle a negative sign while assigning those where it is brighter a positive sign. The two sets of features can then be treated separately during matching, resulting in faster matching and preventing mismatches of positive and negative features. The same approach has been used in SURF [Bay 08], where the sign of a feature corresponds to the sign of the Laplacian which has been computed during detection. Similarly as for FAST, this sign distinguishes bright blobs on dark backgrounds from the reverse situation.

An approach to outdoor camera localization by Arth *et al.* [Arth 12] stores with each reference feature a 3D *normal vector* as the average over all vectors connecting the 3D position of the feature and all cameras observing it in the offline structure-from-motion process. During runtime, i.e. for live images, mapped to a panorama in this case, every current feature is assigned a 3D normal vector based on the camera orientation obtained from inertial sensors and magnetometer. During matching, this approach then only compares features with a similar normal vector, resulting in more accurate and faster localization.

The approach described in [Frit 10] uses EXIF information stored with digital images to gain information on the metric size of objects shown in the image. Thereby, they may prevent erroneously matching similar looking features that differ in physical scale.

In section 2.3.8 we show how we determine absolute spatial attributes, such as absolute orientation, absolute scale and absolute 3D position, of camera features to improve their distinctiveness and speed up matching.

2.3.7 Comparisons and Evaluations of Feature Detectors and Feature Descriptors

A variety of researchers reviewed, compared and evaluated different feature detectors and their combination with feature descriptors. A very elaborate comparison of different feature detectors can be found in [Rost 10] as well as in this survey [Tuyt 08]. Combinations of feature detectors and feature descriptors have for example been evaluated in [Miko 05b, Hein 12, Luo 09, Gaug 11a, Lieb 09].

Generally, over all tests and situations, SIFT provides the best results particularly in the presence of geometric transforms, such as changes in scale or rotation. For non-geometric transforms, such as blur or JPEG compression, [Hein 12] find BRIEF to outperform SIFT. In practice a tradeoff between matching accuracy and runtime needs to be made, particularly for real-time applications on handheld devices, where SIFT is too expensive to use.

2.3.8 Relation to the Contributions of this Thesis

The main contributions of this thesis are concerned with feature descriptors. More specifically, we propose different means to improve the invariance and distinctiveness of visual feature descriptors.

All these approaches can be used to improve any feature descriptor in combination with any feature detector. Our work advances state-of-the-art as described in the sections *Feature Normalization* (section 2.3.2) and *Additional Distinctive Feature Attributes* (section 2.3.6) by taking advantage of the auxiliary sensors contained in modern handheld devices.

Concerning in-plane orientation assignment, we propose to align the orientation of feature descriptors with gravity measured with inertial sensors in section 5.1. In contrast to approaches that gain a reproducible feature orientation from the intensities of neighboring pixels to remain invariant against rotation, this approach results in clearly distinguishable descriptors for congruent features in different orientations.

To account for out-of-plane rotations when imaging horizontal surfaces, section 5.2 proposes to rectify the camera image based in the measured direction of gravity. This eliminates the effects of out-of-plane rotations in the image and therefore results in more similar descriptors of a physical point on a horizontal surface when imaged from different steep angles.

Assuming that invariance to in-plane rotation and scale is provided, we also propose a learning-based offline method explained in section 5.3 to improve invariance to out-of-plane rotations without any modification to the runtime method. Based on matches between synthetic views of an object covering the degree of out-of-plane rotations that should be supported, this method determines a representative set of descriptors which enables localizing the object even from steep angles. As opposed to approaches discussed above, our approach does neither increase the computational complexity during runtime (as the case for [Miko 05a] and [More 09]), not does it increase the memory consumption of the set of reference descriptors (as the case for [Chen 09]).

Compared to the methods described in [Irsc 09, Li 10], our approach explained in section 5.3 does not attempt to solve a set cover problem. We are not interested in a minimal set of features that covers a set of views, but in a subset of a given size that provides the most representative description of an object. Thereby, we do not cluster or average descriptors belonging to the same physical point but simply let the algorithm decide how many and which descriptors for a certain feature result in the most matches and therefore are expected to provide best localization results.

In addition to our contributions towards improving the invariance of feature descriptors we further add distinctive attributes to features. The absolute orientation of the dominant gradient direction relative to gravity is used in section 5.1 to add distinctiveness to features on (close to) vertical and static surfaces. In outdoor applications we propose to compute the Absolute Spatial Context (i.e. the absolute position, absolute orientation, absolute scale) of a feature in section 5.4. This adds significantly more distinctiveness to camera features than the surface normal as proposed in [Arth 12] and therefore vastly improves camera localization outdoors based on a single image.

All of our contributions to improve feature descriptors can be analogously applied to feature classifiers. We do not explicitly evaluate the applicability to feature classifiers in this work. However, all our improvements that are related to the feature normalization stage between detection and description,

can similarly be applied between feature detection and feature classification. It is also easily possible to enhance feature classifiers with additional distinctive feature attributes, which may significantly improve the distinctiveness of feature classifiers.

2.4 Frame-to-Frame Tracking

Given the camera pose of a previous frame, approaches to (incremental) frame-to-frame tracking aim at determining the camera pose for a live camera image based on the image and the previous pose. Under the assumption that the camera motion since the previous frame is small, this problem is easier than that of tracking initialization as explained above and therefore allows for different approaches.

One common class of frame-to-frame tracking approaches is based on tracking image gradients which correspond to edges in a model of the object to track, e.g. [Drum 02, Klei 06]. Other approaches are based on tracking point features from frame to frame, e.g. [Klei 07, Wagn 09]. In this case the descriptor for a feature becomes a patch (or template) around the feature as discussed in section 2.3.3.1. However, instead of detecting features in each image, this approach only searches in the neighborhood of the position of the feature in the previous frame. Approaches based on tracking point features are particularly robust but require the object or environment to track to be textured.

Another class of approaches uses dense tracking which takes advantage of the information of all pixels on an object. Algorithms such as Forward Additive (FA) [Luca 81], Forward Compositional (FC) [Shum 00], Inverse Compositional (IC) [Bake 04] and Efficient Second-Order Minimization (ESM) [Benh 04] can be used to track planar objects (also called templates) from frame-to-frame by estimating a homography, as discussed in section 6.1. Dense approaches can further be used to track non-planar scenes of unknown structure [Newc 11] or partially known structure [Criv 14].

2.4.1 Relation to the Contributions of this Thesis

We propose a method to aid any kind of template tracking when dealing with horizontal planar objects. A measurement of the gravity direction enables gravity-rectification of the camera image, which simplifies perspective transformations to similarity transforms. This allows for using faster template trackers which converge more often for example because they only estimate an affine transform. We further show how this approach enables an edge-based template tracking method to estimate 6DoF camera motion, even though the method itself (without gravity-rectification) cannot support perspective transforms on mobile devices due to memory limitations.

2.5 Auxiliary Sensor-Aided Visual Localization and Tracking

The idea to combine the information obtained from auxiliary, in particular inertial, sensors with those from a rigidly connected visual sensor, i.e. camera, is the foundation of a variety of work on camera tracking. Inertial sensors usually sample at much higher frame rates than cameras, and the inertial sensor measurements corresponding to the camera motion since the previous camera image, is usually available before a new camera image becomes available. There are different approaches to take advantage of the complementary nature of the two kinds of sensors and to compensate for the weaknesses in each of them.

Various approaches to sensor fusion of visual sensors and inertial sensors exist to improve filter-based visual SLAM, e.g. [Pini 07]. In this case inertial sensor measurements provide an input in addition to the results of visual tracking to obtain a better estimate of the camera trajectory. Particularly, inertial sensors enable the estimation of absolute scale in monocular SLAM systems [Nutz 11], which is crucial in many applications.

The visual tracking in this case benefits only *indirectly* from the inertial sensor measurements, e.g. as a result of more accurately localized map features. In this thesis, we are particularly interested in how the readings of auxiliary sensors can be used to *directly* aid visual camera localization and tracking by supporting low level computer vision methods.

For example changes in camera orientation measured with a gyroscope attached to a camera are used to predict the position of feature points that have been used in the previous image in a new camera image [You 99, Hwan 09]. This enables the visual feature tracking to cope with more rapid and unpredicted camera motion without losing track.

For edge-based tracking, Klein and Drummond [Klei 04] use a three-axis gyroscope to provide a pose prediction for the visual tracker, which makes it more robust to large displacements resulting from fast camera motion. They additionally scale the kernel used for edge detection according to inertial sensor readings, which makes the visual tracker cope much better with blur as a result of rapid motion. Furthermore this work shows how the biases of the gyroscope can be re-calibrated during run time given the poses obtained from visual tracking.

To improve feature matching in catadioptric images Bazin *et al.* [Bazi 08] measure the relative change in orientation of the camera between two images using a gyroscope and warp the second image to be aligned with the first one before matching SIFT features of the two images. They do not use the knowledge of the two images being aligned in the SIFT computation which explains why they could not report on any significant improvement when matching these highly rotational invariant feature descriptors.

Bleser and Stricker [Bles 09] present sensor fusion algorithms to predict the appearances of features by rendering a 3D model of the scene they aim to track with a pose predicted based on inertial sensors and the pose of the previous frame.

Ventura and Höllerer [Vent 12] use a gyroscope to estimate the changes in orientation of a handheld device since a query image has been sent to a server for camera localization. This estimated change in rotation is then applied to the pose obtained from the server response before it is used to initialize a visual tracker running on the handheld device.

The gravity measured with inertial sensors is used in [Lee 11] to automatically rectify reference images of planar areas on the ground plane or vertical surfaces. During tracking they do however not use the inertial sensor information anymore.

For outdoor camera localization, there are approaches that in addition to inertial sensors take advantage of a coarse absolute camera location obtained from a GPS as well as a coarse absolute heading measured with a magnetometer. Note that this combination of sensors provides a full 6DoF camera pose – even though it is usually inaccurate.

An approach to wide-area localization proposed by [Reit 07] uses this coarse sensor-based pose as a prior for model-based tracking of an urban environment. Because their tracker requires the prior to be much more accurate than the precision usually obtained from GPS, they attempt initialization with a set of prior poses sampled around the original GPS position.

Chen *et al.* [Chen 11] exclude distant database images from matching based on GPS readings of a mobile phone for visual landmark recognition.

Arth *et al.* [Arth 12] exploit the coarse position and orientation of a mobile phone obtained from sensors to partition 3D reference features of an outdoor environment according to their position and orientation (camera heading). During runtime, they then match camera features only against those reference features located in the cell where the camera is according to GPS. They also only match against reference features resulting from camera views with a heading similar to the one currently measured with the attached compass. This increases both the robustness and speed of their 6DoF localization method.

2.5.1 Relation to the Contributions of this Thesis

The majority of approaches discussed above use inertial sensors to improve frame-to-frame tracking in terms of their robustness to (rapid) motion. The motion estimate from inertial sensors is used to tell a visual tracker where to look for certain features and in some cases also what these features might look like. In addition to inertial sensor readings, these approaches require a pose estimate for the previous camera frame.

Our approaches presented in section 5.1, section 5.2, section 5.3, and section 5.4 aim at improving the initial localization of a camera, i.e. without having a pose for any previous frame. In general, all our methods are based on sensor measurements of *absolute* orientations (i.e. with respect to gravity), whereas all gyroscope-based approaches discussed above make use of measurements of an orientation *relative* to the previous frame.

Our method proposed in section 5.4 shares some of the concepts explained above to increase the distinctiveness of visual feature descriptors. Instead of using a coarse sensor pose to project the reference model into the camera image, as in [Reit 07], we use it to project the camera features onto a coarse model of the environment. Thereby, we gain their Absolute Spatial Context comprising of the coarse 3D position, absolute scale, and absolute orientation, making it possible to constrain the set of reference features to match against in 3D space. This not only makes it easier to account for the different accuracies of the different sensor readings, but more importantly, the set of reference features to match against is determined for every camera feature individually. This then makes it possible to deal with repetitive visual features. While all features corresponding to windows on a building façade would fall into the same orientation bin, and most likely the same position bin (according to the partitioning proposed by [Chen 11] or [Arth 12]), our proposed method can help distinguishing them.

2.6 Evaluation Datasets

Meaningful evaluation and comparison of computer vision algorithms requires suited test data and means to quantify an algorithm's performance. This might for example be repeatability for feature detectors, the accuracy of matches for feature descriptors, or the accuracy of a camera pose for localization and tracking methods. The most reliable way of evaluating such methods is to compare their results with *ground truth*. There exist a variety of datasets including ground truth information, which have been made available to the research community.

The Oxford dataset [Miko 05a] comprises of sets of images capturing eight different scenes with a camera. The images in each set differ either in terms of spatial transforms, such as changes in viewpoint for planar scenes, or zoom and rotation for non-planar scenes, or in terms of non-spatial transforms such as blur, changes in light, and JPEG compression. The homographies between image pairs are provided as ground truth, which enables verifying correspondences between all images and therefore the evaluation of both feature detectors and feature descriptors. In total, the dataset comprises of 48 images.

The Strecha dataset [Stre 08] provides camera images of six different buildings along with the corresponding ground truth intrinsic and extrinsic parameters of the camera. The dataset further includes a ground truth 3D triangle model of each building, which is not only suited to evaluate dense stereo algorithms, but also feature detectors, feature descriptors, and camera localization methods.

Heinly *et al.* [Hein 12] published a dataset including images with ground truth homographies or intrinsic and extrinsic camera parameters for more different scenes.

Other datasets contain video sequences instead of single images, which additionally allows for evaluation of short-baseline frame-to-frame tracking. Metaio's publicly available template tracking dataset [Lieb 09] comprises of videos with 48,000 camera frames in total. The images are captured

by an industrial camera and contain eight different planar templates under different camera motions. The ground truth poses, i.e. extrinsic camera parameters, for every frame were determined with a measurement arm attached to the camera.

Gauglitz *et al.* [Gaug 11a] present a similar dataset with more different camera motions, i.e. 16 videos for each of the six different planar objects. Ground truth is provided in the form of homographies mapping the planar template in each image to a canonical reference coordinate frame.

The City of Sights [Grub 10] dataset comprises of video sequences capturing a 3D miniature city with corresponding ground truth poses obtained from a robotic arm moving the camera, an optical infrared tracking system performing outside-in camera tracking, and a mechanical measurement arm attached to the camera. This dataset furthermore includes digital 3D models of the miniature city as well as paper models that enable rebuilding the miniature city anywhere.

While all previous datasets provide (RGB) images or video sequences without depth information, the dataset of Sturm *et al.* [Stur 12] provides registered RGB-D sequences of different 3D objects and scenes captured with an active stereo camera (Microsoft Kinect). Again ground truth poses for all frames are provided.

None of the datasets discussed above includes readings of auxiliary sensors attached to the camera, such as inertial sensors, an electronic compass, or GPS. The San Francisco Landmark Dataset [Chen 11] includes a city-scale database of panoramic images taken from a vehicle as well as a set of query images taken with a variety of different camera phones. These query images come with an associated GPS location but do not include any measurements of orientation, as from inertial sensors or an electronic compass. More importantly, this dataset does not include ground truth poses for the query images but only labels of which landmarks are present in which query image. While this is suited for the evaluation of landmark recognition approaches, it is not well suited to evaluate feature detectors, feature descriptors, or camera localization methods.

It is generally a tedious task to determine the ground truth pose for real camera images – particularly in wide area outdoor environments. [Irsc 09] do not have ground truth information and therefore measure the effective number of inliers to rate if camera localization succeeded or not. [Vent 12] synthesize camera images as unwarped parts of omnidirectional images. For ground truth, they use the position of the omnidirectional camera determined in the SfM process to create the reference map. Similarly, [Arth 12] simulate online-created panoramic images as subsets of existing full panoramas for evaluation, and manually set the corresponding ground truth position.

Datasets with ground truth information for wide area outdoor environments exist in the robotics research domain, e.g. [Wulf 07], which consequently do not contain handheld camera footage and motion.

2.6.1 Relation to the Contributions of this Thesis

The majority of methods proposed in this thesis requires a measurement of the direction of gravity for each camera image, which we obtain by means of inertial sensors attached to the camera. Because none of the available evaluation datasets includes such information, we create our own simple evaluation datasets in section 4.1 and section 4.2 that include inertial sensor readings. We furthermore present an approach to exploit existing datasets with ground truth poses but without sensor readings in section 4.3, by synthesizing realistic sensor readings from the ground truth pose.

Our dataset, which we introduce in section 4.4, contains sequences of an urban outdoor environment taken with an off-the-shelf mobile phone, which include the readings of GPS, compass and the direction of gravity, for each camera frame. Most importantly, it comprises of ground truth information on the geometry and texture of the environment, and the full 6DoF ground truth camera pose for every single frame. Our dataset is the first one to include all these information which are crucial for the evaluation of handheld camera localization methods in outdoor environment. We made it publicly available to foster further research in this area, which is very important for real-world Augmented Reality applications in the future.

3 Problem Statement and Motivation

This chapter explains the motivation for this thesis, and the problems it aims to tackle. For visual localization, there are two important properties of a feature and its description. It is crucial, that a feature is described in a *distinctive* way, such that it is possible to distinguish it from other features. Furthermore, it is important, that the description of a feature is *invariant* to the camera viewpoint. Both is provided up to a certain extend in state-of-the-art methods, but there are exceptions which are crucial in real-world applications and therefore need to be improved.

A major difficulty for camera localization approaches that need to be able to deal with a large number of objects or large environments, is that due to the visual repetitiveness in man-made environments, distinctiveness of visual features is often not provided. No matter which approach is used to model or describe these features in a localization method, already the features themselves are visually not distinguishable. As a result, such features cannot be reliably matched.

The second problem which motivated this work is the limited invariance of feature descriptors and also template tracking methods to out-of-plane rotations. This may lead to failure of localization methods when observing objects from viewpoints significantly different from those used for modeling the reference description. The effect often is, that Augmented Reality applications only work well from certain viewpoints, which is clearly not acceptable for successful real-world applications.

In this thesis, we show that both problems can be addressed with the help of sensors which off-the-shelf mobile phones are equipped with. We also show how offline pre-processing can be used to improve the invariance of visual feature descriptors to out-of-plane rotations.

3.1 Repetitive Visual Features Everywhere

The world around is highly repetitive. In urban environments, man-made objects dominate the visual appearance. These objects not only include buildings but also cars, traffic signs, billboards, and even the clothes people wear. Since these objects were mass-produced, individual exemplars cannot be visually distinguished.

Visual camera localization is often based on local features in an image. Therefore, it can only work reliably, if either every feature itself is unique and distinct, or if the composition of features visible in the image is unique and distinct. Obviously, this not the case with repetitive visual features as they are widespread everywhere. Therefore, approaches to camera localization solely relying on visual information cannot be employed in large-scale applications.

3.1.1 Scales of Visual Repetitiveness

Visual repetitiveness occurs at different scales which might require different ways of dealing with it. In the following, examples are given for scales at which visual repetitiveness is commonly found.

Large-Scale The same car can be located at many distant places at the same time, or the same front door can be installed in different buildings in different cities around the world. Therefore seeing only this car or only the front door does not provide enough information to tell its (global) location, e.g. if is located in Munich, Berlin, or Sydney. These individual objects and their features, which are visually not distinguishable, differ by their *(large-scale) location*, which is not apparent given only the information contained in an image, i.e. a photo.

Building-Scale The windows of a building usually form a significant part of a building's visual appearance. Since a façade in the most cases contains many identical windows, they all look the same and given only the view of a single window, it is impossible to distinguish, which window at which location we observe. In this case, repetitive visual structures only differ by their *(building-scale) location*, which is not apparent from an image only.

Window-Scale When looking at a single window, visual repetitiveness in an even smaller scale becomes a problem. The four corners of a window usually look locally the same. A human being is able to distinguish them since we have a sense telling us where down is and therefore we can identify e.g. the upper left corner and distinguish it from e.g. the lower left corner. Given only a photo of a single corner, often does not allow to tell its orientation if the camera orientation is unknown. The four corners of a window differ by *absolute orientation* (and window-scale location), and therefore cannot be distinguished given only an image.

Small-Scale It is possible to zoom in further and to analyze visual repetitiveness in smaller scales e.g. patterns on a wallpaper or a carpet, or blades of grass on a lawn that all look very similar. In this work, however, we focus on structures and features that can be imaged with off-the-shelf consumer cameras with image resolutions ranging from QVGA with (320×240) pixels to XGA with (1024×768) pixels from a distance, and therefore do not consider repetitive structures at small scale.

At Different Scales There are also visual features and objects that occur at different physical scales. For example letters and the logos of common brands are apparent at different scales in urban environments, e.g. on billboards and illuminated advertising. Another important example are rectangular features at building façades, which exist at different scales, see figure 3.1. While these features do not necessarily look the same, their appearance is often very similar and feature description methods are invariant to the remaining differences, e.g. contrast. Features as shown in figure 3.1 differ by *physical scale*, which is not apparent given an image only.

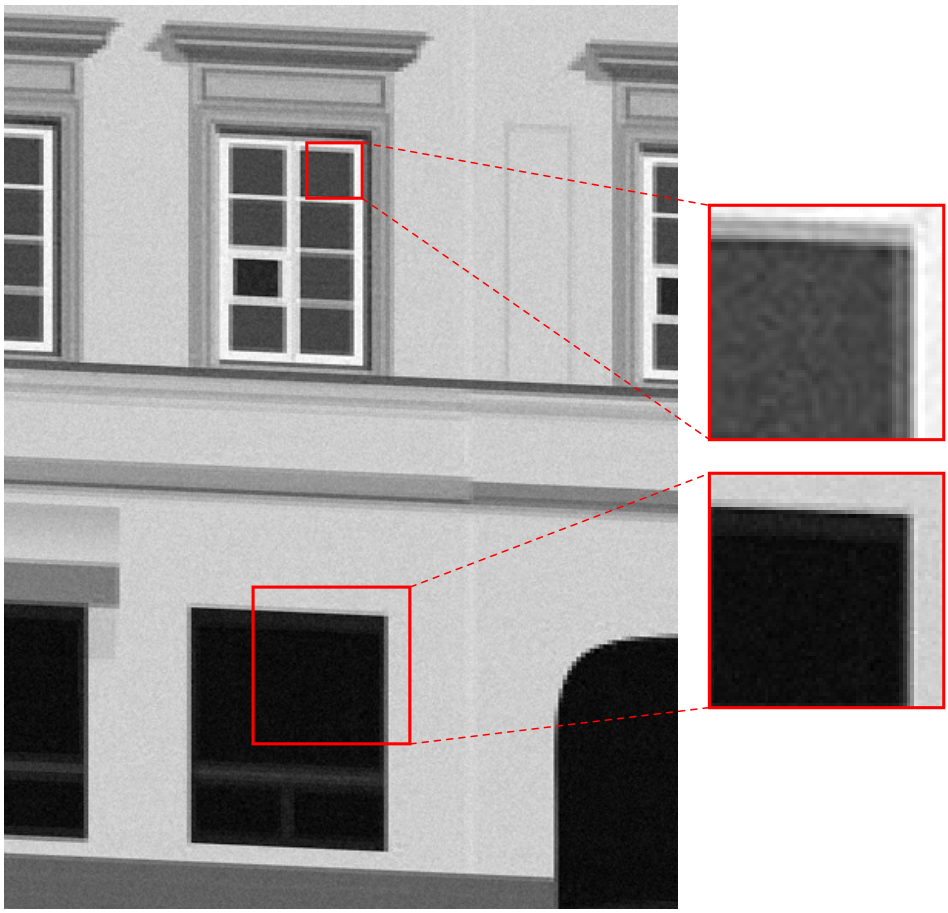


Figure 3.1: Building façades often include similar looking rectangular features at different physical scales. These cannot be reliably distinguished based only on the information contained in the normalized image patches around them, which are shown as insets on the right.

3.1.2 Overcoming Visual Repetitiveness in Urban Environments

In Augmented Reality applications that use a single (planar) object as reference, it might not be crucial to be able to distinguish the features describes above. But for real-world applications, that need to deal with and distinguish between a large number of objects or environments, it is very important to resolve these ambiguities and to develop means to describe these features in a more distinctive way.

The approach pursued in this thesis is to solve this problem by incorporating information measured with auxiliary sensors, because it cannot be solved entirely image-based. These sensors, which are capable of measuring the (partial) location or the (partial) orientation of the camera in a global reference frame, can provide a valuable spatial context for visual information captured with a camera. We show throughout chapter 5 how such context helps overcoming visual repetitiveness.

3.2 Invariance of Visual Feature Descriptors

Besides being able to distinguish repetitive visual features from each other, it is also important to describe these features in a way that is invariant to changes in viewpoint. An unconstrained 6DoF motion of a camera results in different kinds of transformations visual features undergo in the camera image. The invariance of state-of-the-art visual feature detectors and descriptors to these different degrees of freedom varies significantly.

3.2.1 Degrees of Freedom of Invariance

From the perspective of a camera, 6DoF camera motion results in changes in translation, scale, in-plane rotation, and out-of-plane rotation of an imaged feature located on a planar surface. For features on non-planar objects, additional effects, such as self-occlusions, may occur. We will not discuss these, because we assume the surface around a feature to be locally planar, which is a commonly made assumption.

Translation Translation in the image plane is usually handled by detection of salient visual features (e.g. corners, or blobs), which provide a good invariance to translation. More details on state-of-the-art feature detectors can be found in section 2.3.1.

Scale Changes in scale result from changes in the distance between the camera and a feature and therefore are caused by camera translation. Most state-of-the-art feature detectors are scale-invariant by exploiting scale space. Image pyramids represent an image at different scales and feature detection is either performed on each pyramid level individually or considering multiple pyramid levels and detecting features at interpolated scales between the scales of the pyramid levels.

While the former results in discrete scale-invariance, the latter provides continuous scale-invariance. In general, scale-invariance of feature detectors works reliably in state-of-the-art methods. Details can be found in section 2.3.2.

In-Plane Rotation Invariance to in-plane rotations is achieved by assigning features one or more repeatable orientation(s), as discussed in section 2.3.2 and extensively evaluated in [Gaug 11b]. There are different approaches to orientation assignment based on pixel intensities in a region around a feature, which provide reliable results (see [Gaug 11b]). Normalizing the patch around a feature given this orientation provides full invariance to in-plane rotation. Other approaches describe a reference feature in a sampled subset of all possible orientations such that for any in-plane rotation there exists a reference feature description with a similar orientation [Calo 12].

Out-of-Plane Rotation Out-of-plane rotations result in perspective foreshortening of a feature in the image. Different approaches try to detect affine regions [Miko 05a] in an image which provide an anisotropic scaling for features to account for the effect of perspective foreshortening. While an affine transform can only approximate perspective foreshortening effects, these approaches are in practice not used in real-time applications on mobile devices because of their computational cost. More detailed information on this problem can be found in section 2.3.2. Invariance to changes in out-of-plane rotations is an important topic which has not been satisfyingly solved, yet.

3.2.2 Challenges and Approaches Pursued

In this thesis, we tackle the problem of invariance to out-of-plane rotations of feature descriptors and also of template tracking methods, which are substantial parts in many Augmented Reality applications. As discussed above, this degree of invariance presents a challenge in state-of-the-art while it is crucial for reliable real-world applications.

In chapter 5 and chapter 6 different approaches are explained and evaluated, that make use of the auxiliary sensors built into handheld devices, particularly inertial sensors, to achieve an improved invariance when dealing with objects at a known orientation with respect to gravity. We also investigate how offline pre-processing can be employed to further improve invariance of visual feature descriptors to out-of-plane rotations.

4 Evaluation Methods and Datasets

This thesis proposes different novel algorithms that can be used for camera pose localization and tracking for handheld AR. In order to compare these approaches with state-of-the-art methods, we need to be able to judge their results quantitatively. That is, given a camera pose, or a point correspondence, or a projective transformation between a reference template and a current camera image, an evaluation method needs to provide its level of accuracy.

The most reliable mean to quantitatively evaluate a localization method is to compare the obtained result with *ground truth*. However, as we will see in this chapter, creating realistic datasets comprising of image sequences, readings of auxiliary sensors, and the corresponding ground truth poses is a difficult and time-consuming task to undertake. This is the reason why we also used evaluation methods in this thesis, which do not include ground truth information.

The easiest way to create imagery with known camera parameters or a known transformation is to synthesize it. Computer graphics cameras allow for rendering virtual objects to images and thereby enable control over all parameters. While for synthetic data, not only the ground truth camera pose but also the structure and material of the objects is known, the main problem is that the resulting data is not realistic. Even if imaging artifacts, camera imperfections, and appearance changes depending on illumination are modeled, they still only cover a part of reality and the applied artifacts are again synthetically modeled and therefore not realistic. Thus, it is crucial to work with real imagery captured with a real camera, to obtain realistic results.

One important thing to keep in mind is that the majority of the methods discussed in this work make use of inertial sensor measurements in addition to camera images. Some approaches additionally use information from compass and GPS. For this reason using existing benchmark datasets for camera localization and tracking that do not contain any sensor readings, e.g. [Lieb 09, Gaug 11a], is not straightforward.

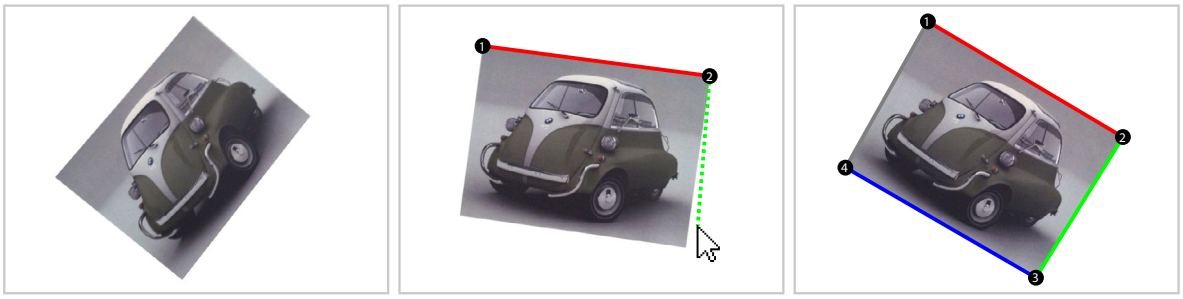


Figure 4.1: Manual selection of the four corners of a template in captured images is a simple approach to provide ground truth information.

This chapter presents four benchmarking methods and resulting datasets that were developed and used in this thesis and discusses their individual pros and cons. The evaluation methods and datasets are designed to provide meaningful results for handheld AR. Most of the methods aim at the evaluation of methods for localization and tracking of planar objects, which currently is the most commonly used technology in commercial handheld AR applications. The fourth dataset enables evaluation of camera localization methods for outdoor environments based on GPS, compass, inertial sensors, and a camera image. This comprehensive dataset is the first of its kind and we have made it available to the public for research purposes.

4.1 Manual Ground Truth Generation for Planar Objects

One obvious option to gain ground truth information for real images is to manually define the position of certain points of a reference model in every camera image. For planar objects, the transformation an object undergoes when imaged, can be fully described with a homography, which is a (3×3) matrix. This transformation can be computed with a closed-form solution given only four corresponding points. Based on this homography, not only a computed pose but also point correspondences between reference and current images can be easily validated. We use this method to benchmark feature descriptors by their matches in sections 5.1.3, 5.1.4.1 and 5.2.3. Therefore we save images with corresponding inertial sensor readings for each image and later on determine ground truth manually by clicking the four corners of the template, see figure 4.1.

While this approach provides in theory all information needed for reliable benchmarks, it is in practice only applicable for small datasets. Not only is the manual definition of points, e.g. by clicking on them, very time-consuming but also is it tedious and therefore error-prone. Ideally, all manually defined correspondences need to be double-checked which makes this procedure impractical for image sets containing thousands of images. In addition to that, motion blur, defocus, partial occlusions, and camera poses for which particular reference points are not visible in the camera image can make the precise manual definition of their position in the image impossible.

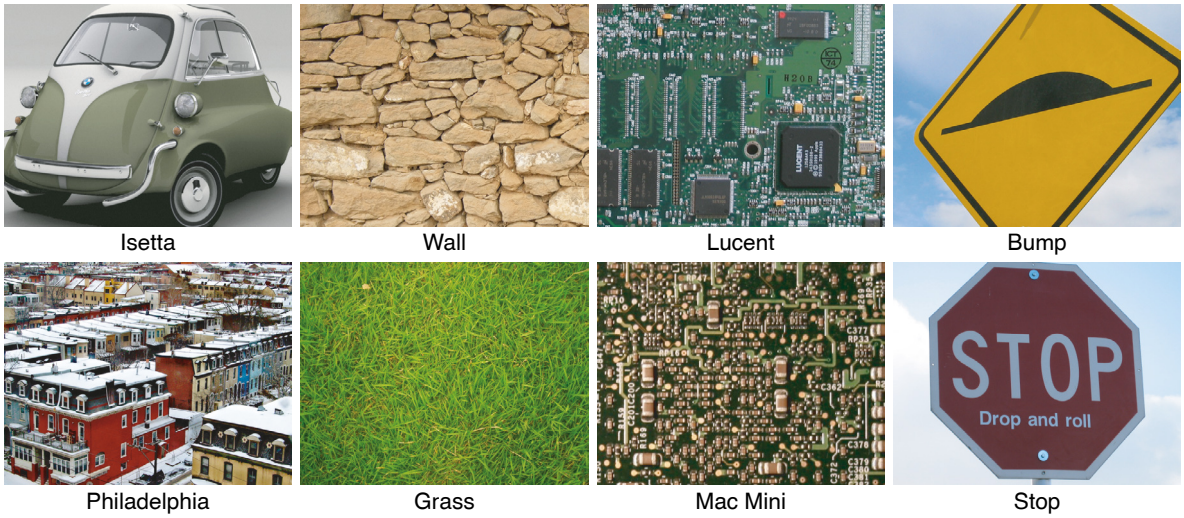


Figure 4.2: The templates used for evaluation through this thesis, as used in [Lieb 09].

4.2 Fully Automatic Pose Verification for Planar Objects

Benchmarking camera poses without any ground truth data can be performed by evaluating a similarity function which is assumed to correspond locally to the accuracy of the pose. We evaluate feature-based localization of a planar template based on the Zero-mean Normalized Cross-Correlation (ZNCC) between the reference template and the current camera image warped with the homography which led to the computed pose. The ZNCC provides normalized measures of correlation between -1, which corresponds to inversely correlated signals, and 1, which means the two signals (i.e. images in our case) are perfectly correlated. Eventually the resulting ZNCC values can either be used as a continuous quality measure or be thresholded to decide if localization was correct or not.

Using this simple approach, we evaluate *Gravity-Aligned Feature Descriptors* in section 5.1.4.2, *Gravity-Rectified Feature Descriptors* in section 5.2.4, and *Representative Feature Descriptor Sets* in section 5.3.4.2. In all cases, we record image sequences and store the corresponding gravity vector with each frame, which was measured using inertial sensors.

This method is very convenient, as it does not require any manual work nor any special hardware. It can be used to evaluate a large number of images and corresponding poses very efficiently. But unfortunately, it can be unreliable in particular cases. Obvious examples are repetitive structures in the environment. If the method is for instance supposed to determine the camera pose with respect to one particular window at a building façade and there are many similar looking windows, it is impossible for this method to tell if the correct window was chosen as reference or not. In general, a high ZNCC value does not guarantee a proper pose. Another important property of this method is, that for instance in the presence of motion blur, defocus, or partial occlusions, even the perfectly correct pose might result in a small ZNCC value.

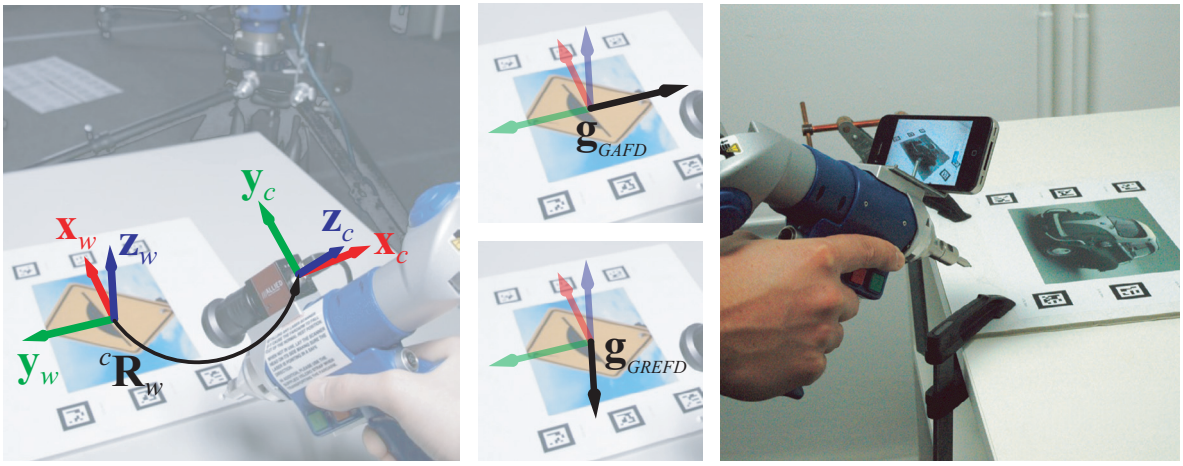


Figure 4.3: Visualization of the involved coordinate systems and the synthesized gravity vectors for the evaluation of GAFD and GREFD.

4.3 Exploiting Existing Benchmark Datasets

The ideal benchmarking dataset for localization and tracking comprises a large number of images taken by a real camera covering a broad range of practice-oriented tasks accompanied by ground truth poses. One example is the publicly available dataset by Metaio [Lieb 09, Meta 09], containing 48,000 camera frames of the eight different templates shown in figure 4.2 under different camera movements. There exist ground truth poses for every frame gained with a measurement arm attached to the camera as shown in figure 4.3 (left). However, this dataset does unfortunately not contain any inertial sensor measurements which are crucial for the methods we aim to evaluate.

Creating new sequences with a similar setup does not only require expensive hardware but also takes a lot of effort. Most importantly, it will result in new sequences that do not enable comparison with methods that were tested on the original dataset. If it was possible, we would ideally not take new sequences but instead record the inertial sensor measurements for the already existing sequences to ensure comparability. This section is based on the paper “Benchmarking Inertial Sensor-aided Localization and Tracking Methods” [Kurz 11c] and describes how existing datasets comprising ground truth poses can be expanded by (synthetic) inertial sensor readings.

We propose to synthesize inertial sensor measurements from the ground truth pose provided in existing benchmark datasets. Knowing the camera pose allows for transformation of any vector in a world coordinate system, including an arbitrarily defined gravity vector, into the camera coordinate system. The transformed vector can then be used in the same way an algorithm would use the gravity vector measured with inertial sensors. In the following, we explain and validate this procedure before eventually employing it using Metaio’s template benchmark dataset to evaluate localization of templates in section 5.1.4.3 and section 5.2.5.

4.3.1 Synthesizing Realistic Sensor Measurements

In order to validate the proposed method, it is essential to compare the synthetic gravity vectors used in this approach with gravity vectors measured with real inertial sensors. Therefore, we rebuilt the setup used in [Lieb 09] but replaced the industrial camera by a mobile phone (Apple's iPhone 4) which in addition to a camera is equipped with inertial sensors, cf. figure 4.3 on the right. The interested reader is referred to [Lieb 09] for details on the calibration procedure of the setup.

Using the calibrated setup, we record a sequence of 500 frames showing the *Isetta* template and six fiducial markers located on a horizontal surface (as in figure 4.3 on the right) while alternating view points and camera angles. The images have a resolution of (480×360) pixels and are recorded at approximately 25 Hz. With every camera image, we store a time-stamp and the corresponding gravity vector measured with inertial sensors on the phone. In parallel, a PC that is connected to the measurement arm, stores corresponding ground truth camera poses. After synchronization of ground truth poses and camera images based on the detected corners of the markers visible in the camera image, the residual of the reprojected corners is 0.94 pixels which makes the sequence usable as ground truth data.

Now that we have camera images with corresponding gravity measurements and ground truth poses, we aim to compare the real gravity measurements $\mathbf{g}_{sensor}(t)$ with the corresponding ground truth vector $\mathbf{g}_{gtruth}(t) = -{}^{c}\mathbf{R}_{w}(t)\mathbf{z}_{w}$, where $\mathbf{z}_{w} = [0,0,1]^{\top}$. As the print-out is located in a horizontal orientation, gravity corresponds to the the negative z-axis of the world coordinate system ($-\mathbf{z}_{w}$) which needs to be transformed to the camera coordinate system by the rotational part of the ground truth pose ${}^{c}\mathbf{R}_{w}$. The involved coordinate systems are illustrated in figure 4.3. Once we know the characteristics of the real sensor data, we are able to synthesize gravity vectors $\mathbf{g}_{synth}(t)$ based on $\mathbf{g}_{gtruth}(t)$ that behave comparably to the real measurements $\mathbf{g}_{sensor}(t)$.

The degradation model we use comprises of a noise term and a delay between the moment a camera image was taken and the point in time where the corresponding gravity vector was measured. Since the camera images were used for synchronization, they are in sync with the ground truth poses and therefore with the ground truth gravity vectors. Since this is not necessarily the case for real sensor measurements, we first measure the temporal offset between the real gravity measurements and the ground truth gravity vectors. To this end, we compute the delay δ^{*}, for which the sum over the absolute scalar products between the measured and the ground truth gravity vectors over all frames has its global maximum.

$$\delta^{*} = \arg\max_{\delta} \left(\sum_{t} \left| \mathbf{g}_{sensor}(t)^{\top} \mathbf{g}_{gtruth}(t+\delta) \right| \right) \tag{4.1}$$

For the device and sequence we used, the maximum is reached at δ^{*} = 16 ms, see figure 4.4 a, which will be used as offset in the following. After synchronization of the real measurements and the ground truth, we aim to quantify the distribution of the error between the noisy real measurements

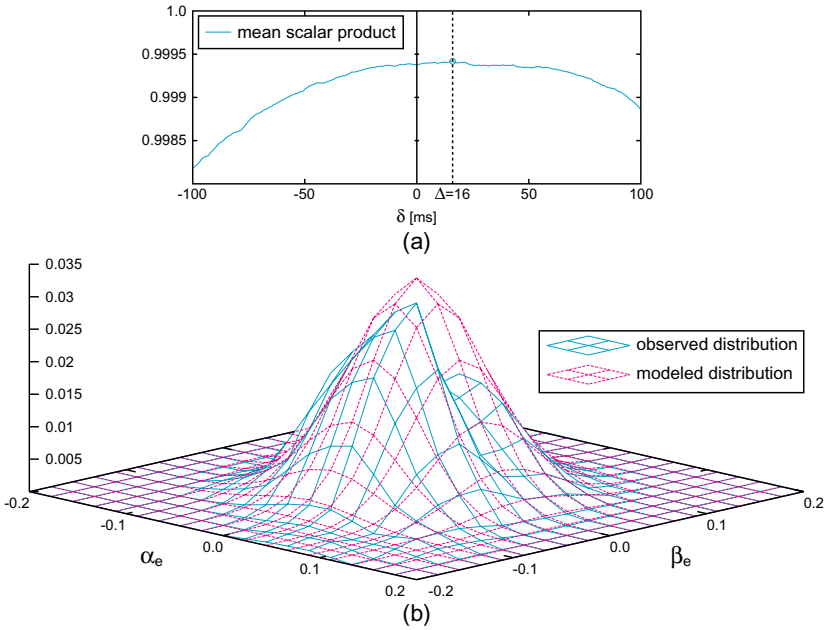


Figure 4.4: Correlation between the measured and the ground truth gravity vectors as a function of the delay between the two (a). The observed error distribution of the real gravity measurements can be modeled with a Gaussian distribution (b).

and ground truth. We therefore parametrize the normalized gravity vectors with two angles (α_s, β_s) and (α_g, β_g) which are computed as

$$\mathbf{g}_{sensor} = \begin{bmatrix} \cos\alpha_s \cos\beta_s \\ \sin\alpha_s \cos\beta_s \\ \sin\beta_s \end{bmatrix} \text{ and } \mathbf{g}_{gtruth} = \begin{bmatrix} \cos\alpha_g \cos\beta_g \\ \sin\alpha_g \cos\beta_g \\ \sin\beta_g \end{bmatrix}. \tag{4.2}$$

It turns out, that the distribution of the angular differences between sensor measurements and ground truth data $\alpha_e = (\alpha_s - \alpha_g)$ and $\beta_e = (\beta_s - \beta_g)$ can be modeled with a two-dimensional Gaussian distribution parametrized with expected value $\mu = (0,0)$ and variance $\sigma^2 = 0.00092$, as shown in figure 4.4 b. We use the Box–Muller transform [Box 58] to compute noise with the modeled distribution of the real inertial sensors to synthesize gravity vectors as

$$\mathbf{g}_{synth}(t) = \begin{bmatrix} \cos(\alpha_g(t+\delta^*)+X)\cos(\beta_g(t+\delta^*)+Y) \\ \sin(\alpha_g(t+\delta^*)+X)\cos(\beta_g(t+\delta^*)+Y) \\ \sin(\beta_g(t+\delta^*)+Y) \end{bmatrix} \tag{4.3}$$

where X and Y are two independent random variables with a normal distribution of standard deviation σ.

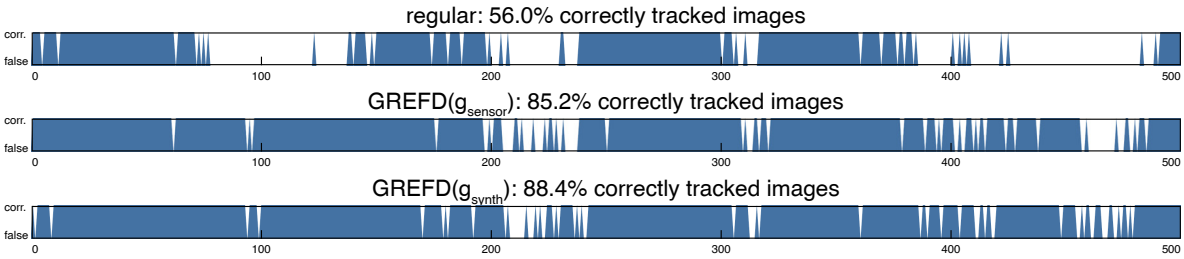


Figure 4.5: Distribution of correctly localized images in a sequence of 500 frames showing the *Isetta* template using regular feature descriptors, real gravity measurements in GREFD($\mathbf{g}_{sensor}$) and the proposed method in GREFD($\mathbf{g}_{synth}$). The latter two methods provide comparable results.

4.3.2 Validation of the Synthesized Sensor Measurements

Using the above, we create synthesized gravity vectors that are comparable in terms of noise and delay to those provided by the inertial sensors of an iPhone 4. In order to validate that using these vectors also provides comparable results on the template localization performance, we ran a template localization and tracking method on the above mentioned image sequence in three different configurations.

First, we use regular feature descriptors to receive a reference performance measure. We then run the localization and tracking method again using Gravity-Rectified Feature Descriptors (GREFD), which require a measurement of gravity. These descriptors are designed for (close to) horizontal surfaces and will be introduced in section 5.2. The method is evaluated using both the real gravity vectors measured with inertial sensors (GREFD($\mathbf{g}_{sensor}$)) and the synthesized gravity vectors (GREFD($\mathbf{g}_{synth}$)). The results are shown in figure 4.5 and clearly confirm that GREFD performs similarly when using the proposed method to synthesize gravity vectors compared with real sensor measurements both in terms of distribution and number of correctly localized images. As in [Lieb 09], a template localization is considered correct, if the reprojection error at the template corners is less than 10 pixels.

4.3.3 Resulting Dataset and Applicability

Using the procedure explained above, we synthesized gravity measurements for all 48,000 images contained in the original dataset. This has been done both under the assumption that the template is located upright on a vertical surface ($\mathbf{g}_{GAFD}$) to evaluate Gravity-Aligned Feature Descriptors in section 5.1.4.3 and assuming the template was located on a horizontal surface ($\mathbf{g}_{GREFD}$) for the evaluation of Gravity-Rectified Feature Descriptors in section 5.2.5 and Representative Feature Descriptor Sets in section 5.3.4.1. This data enables the evaluation of inertial sensor-aided localization and tracking methods for planar objects based on a comprehensive set of real camera images. While this is very important and valuable, it does not enable the evaluation of camera localization methods in outdoor environments, which will become increasingly important in the future.

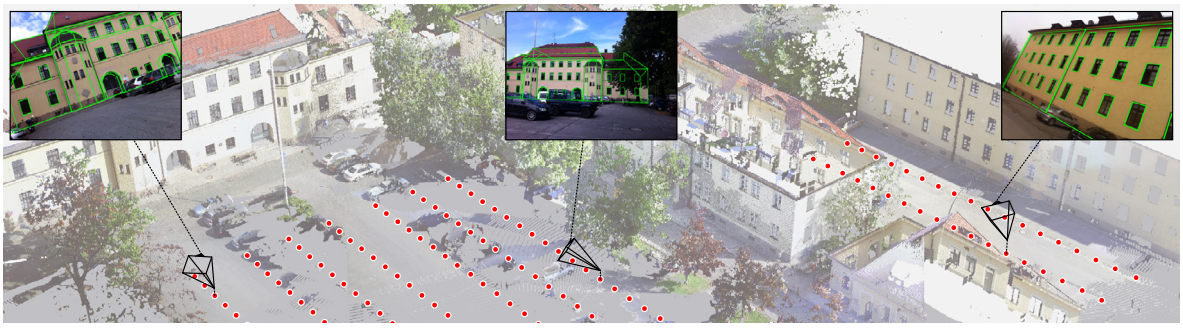


Figure 4.6: Outdoor wide-area ground truth dataset comprising a precise 3D model of the environment and over 45,000 camera images with sensor readings and 6DoF ground truth poses. Exemplary images are shown as insets with their ground truth poses rendered as frustra.

4.4 Outdoor 6DoF Ground Truth Dataset

As explained above, a quantitative evaluation of a 6DoF localization framework requires ground truth information. That is for a set of given (realistic) input data, i.e. camera image, camera intrinsics, all sensor readings, and the required reference models, we need the expected ground truth output data, i.e. a 6DoF pose.

There are ground truth datasets for visual localization and tracking available, but none of them fulfills the requirements to evaluate visual camera localization methods that are designed for handheld AR in outdoor environments and make use of the readings of inertial sensors, compass, and GPS. In this section, which is based on the poster "An Outdoor Ground Truth Evaluation Dataset for Sensor-Aided Visual Handheld Camera Localization" [Kurz 13], we describe our extensive procedure to create the first ground truth dataset for outdoor 6DoF handheld camera localization comprising of:

- a highly accurate, geo-referenced, and textured 3D model of a real urban environment spanning approximately 10,000 square meters,
- video sequences containing over 45,000 individual images of the environment with realistic handheld camera motion taken from different locations with an off-the-shelf mobile phone,
- the sensor readings of GPS, compass, and the gravity vector for each image of the sequences mentioned above, and
- an accurate 6DoF ground truth pose for every single camera image.

One important aspect of the design of this dataset is that it allows for easy expansion by adding more sequences taken with different devices, from different users, and under different illumination.

Figure 4.7: Color-coded individual laser scans

4.4.1 Model Acquisition

We chose an office park as a testing environment, which comprises of a large parking lot, different buildings, some parking lanes, and small streets. The covered area is approximately 100 by 100 meters wide (see figure 4.6).

As we aim to create a very precise and detailed model of the environment, Structure-from-Motion (SfM) methods that reconstruct a sparse point cloud based on a multitude of images taken from different positions, e.g. [Snav 08], are not suitable in this case. Instead, we used a FARO Focus 3D laser scanner to create nine, high-precision panoramic laser scans with texture information for different parts of the environment, see figure 4.7. The individual scans were then registered to a common coordinate system using proprietary software based on 3D-3D correspondences of registration spheres that were placed in the environment. The merged model has finally been geo-referenced based on the latitude and longitude of a set of building corners obtained from OpenStreetMap [Open 14].

As a result, we obtain a highly precise, dense and textured environment model which is referenced with respect to a global world coordinate system. The full registered model with color information is shown in figure 4.6, rendered from a bird's-eye view.

4.4.2 Sequence Recording at Known Camera Positions

There are two important aspects to keep in mind when recording sequences for testing. These are *relevance* for the targeted application and *universality*. It is important to use a capturing device and camera motions similar to those that can be expected to be used in real applications. And it is crucial, that the dataset comprises of a high variance in parameters, such as the camera pose, for the data to be considered universal and representative.

Figure 4.8: Sequence recording at a known camera position using a lead weighted string and measured survey points on the ground.

We decided to use an iPhone 4 mobile phone because it is a very common device and allows to obtain the GPS position, compass heading, and a measurement of the gravity vector for every image. The image resolution is set to (480×360) pixels and the camera's intrinsic parameters were calibrated offline using a checkerboard pattern and Zhang's method [Zhan 00]. We use a custom-made application to capture image sequences with all relevant sensor readings at a frame rate of $\sim$25 Hz and save them to files.

While the camera image is appended to a binary file storing raw image buffers in each frame, the remaining sensor readings are written to text files. The sampling for all auxiliary sensors is controlled by the camera frame rate. We store image timestamps and one measurement of each sensor per image, even though particularly the gravity measurements are available at much higher rates. The camera was configured to use autofocus and autoexposure while capturing the sequences.

To ensure a universal and representative set of camera sequences, it is important to cover many different camera positions distributed over the entire test area. As this area spans about 10,000 square meters, installing an external tracking system that measures the 6DoF pose of the phone with a precision that can be considered ground truth is very complex, if not impossible. Therefore, we limit ourselves to a set of 156 discrete camera positions, which are spread all over the area and are chosen such that placing a camera to these positions is easy to achieve.

Our test environment comprises of a large parking lot and two smaller parking lanes, which are divided into individual cells by white markings on the ground. These serve as constant markers, since they are unlikely to change in the near future. We use the crossings and end points of these markings as survey points, and measure their precise 3D positions with a total station (Trimble 3603 DR). Based on 3D-3D correspondences between additionally measured points on the buildings, and the corresponding points in our ground truth model, the survey points are finally converted into the common world coordinate system. Figure 4.6 displays these survey points as red circles.

We then divide the process of obtaining sequences with 6DoF ground truth poses into two steps. In the first step we capture sequences at known 3D camera positions and recover the corresponding 3DoF orientation in a second step. Attaching a lead weighted string of known length to the phone's

Figure 4.9: An edge model of the environment (left) together with the known ground truth camera position and the coarse camera orientation obtained from the attached sensors (center) enables recovering the ground truth orientation and therefore full 6DoF ground truth pose (right).

camera, makes it easy to precisely move the device to a known 3D position. As shown in figure 4.8, we hold the mobile phone directly over a survey point $\mathbf{s}_i$ on the ground with a string of known length h_i to be sure the camera is located at $\mathbf{t}_i$, which can simply be computed as

$$\mathbf{t}_i = \mathbf{s}_i + h_i \cdot [0,0,1]^\top. \tag{4.4}$$

We use strings at lengths of 1 and 1.8 meters, which can be considered to represent the actual heights users hold their mobile devices. An interesting property when taking sequences at a height of 1 meter, is that they contain much more occlusions of the buildings because of the cars on the parking lot. While capturing and recording sequences (some frames of two exemplary sequences are shown in figure 4.8), the camera only undergoes rotational movements and does not change its position. Since [Chit 05] found out that users prefer standing while using the screen of a mobile phone for information, we believe that our sequences have realistic kinds of camera motion for handheld AR.

4.4.3 6DoF Ground Truth Recovery

For the second step of the ground truth acquisition process, we prepared an edge model of the environment based on the ground truth model. Using the coarse camera orientation obtained from the sensor readings and the accurately known 3D position of the camera, we project the edges into the camera image, cf. figure 4.9. We then find the orientation for which the model best fits gradients in the camera image using exhaustive search in a neighborhood around the initial orientation estimate.

Finally, the recovered 3DoF camera orientation, together with the 3DoF known ground truth position, make the 6DoF ground truth pose. To account for potential errors in labeling or recovery of the rotation, the ground truth poses of all images have been manually verified by rendering a wireframe model onto the video stream. Figure 4.6 displays the recovered 6DoF ground truth pose for three exemplary images of the dataset, in green.

4.4.4 Resulting Dataset

In total, we recorded 100 sequences from different locations and heights imaging façade1, which is shown in the upper row of figure 4.8, and an additional 25 sequences of façade4, cf. figure 4.8 in the lower row. All sequences comprise over 45,000 images and the corresponding sensor information. For every frame, we recovered the 6DoF ground truth pose. Since we used parking markings as easily identifiable camera locations, it was convenient to record sequences at different times of the day and under varying weather conditions. In future the database can be easily expanded by more sequences comprising of more drastic weather changes, e.g. snow or rain, or to contain data from other devices and cameras.

The dataset is available from Metaio's research website [Meta 13] and as of January 2014, there are 16 researchers and research teams from around the globe that downloaded the dataset to use it in their research. We will use it in section 5.4.3.2 to evaluate a framework for camera localization and tracking outdoors, which makes use of the measurements obtained from GPS, compass, and inertial sensors to aid visual feature detection, description, and matching.

5 Improving Invariance and Distinctiveness of Visual Feature Descriptors

Many tasks in computer vision require the discovery of points corresponding to the same physical 3D point in different camera images. One example used in AR is finding an initial estimate of the camera pose. A common approach is to first detect local features in each image and to describe them with a descriptor that is invariant to changes in scale and orientation. The descriptor then enables comparison and therefore matching of features. This chapter introduces different approaches to improve the invariance and to increase the distinctiveness of visual feature descriptors. Most approaches make use of auxiliary sensors attached to the camera, in particular inertial sensors, and assume a static environment.

Classical desktop and kiosk video-see-through AR applications are based on tracking the pose (i.e. position and orientation) of a moving object, e.g. a cardboard box of a product, with a static camera and display. A fundamental finding when looking at real-world handheld AR applications is that here the situation is the opposite. While camera and display are in the user's hand and therefore can be freely moved, the objects that are used for tracking are very often static and have a known orientation with respect to gravity.

Figure 5.1 illustrates a variety of objects that have been used for camera pose tracking in real-world applications. Many of the shown items, such as the posters and billboards, the windows at the building façades and the TV screens, mainly consist of static and (close to) vertical surfaces. Another group of objects including navigation prints on the floor or magazines lying on the table mainly consist of horizontal surfaces. Vehicles have both (close to) vertical and (close to) horizontal surfaces. But all mentioned objects have a very useful property in common. While they can face towards arbitrary points of the compass (heading), their orientation with respect to gravity is static and known.

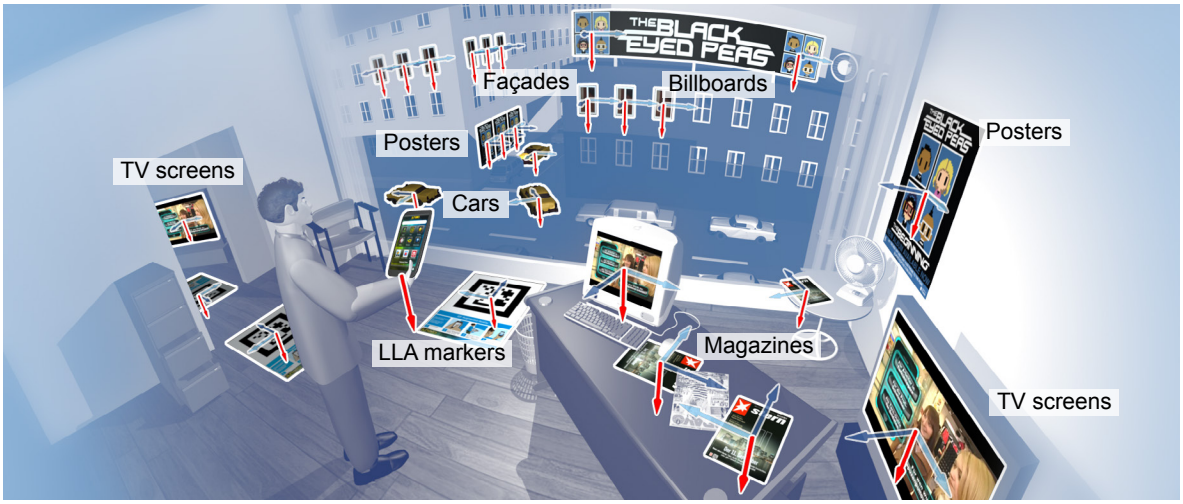


Figure 5.1: The world around us is gravity-aligned. Many real objects that handheld Augmented Reality applications use to determine the camera pose by computer vision have a static and known orientation with respect to gravity. Some examples from real-world applications include posters and billboards, windows, cars, TV screens, magazines lying on a table or printouts attached to the floor for indoor navigation purposes, e.g. LLA markers encoding longitude, latitude and altitude.

In this chapter, we will explain and evaluate how visual feature detection, description, and matching can benefit from this knowledge in combination with gravity measurements provided by inertial sensors. Thereby feature descriptors can benefit in terms of invariance and distinctiveness and we present an approach to improve on these for (close to) vertical surfaces in section 5.1. A different approach dealing with (close to) horizontal surfaces is discussed in section 5.2.

We assume that in future, AR-ready environments will tell a new client device about the Spatial Relationships Patterns [Pust 06] between different objects. Particularly, their orientation with respect to gravity will provide a client device with the information needed to decide which method to use for a particular object.

We further present an approach in section 5.3 which improves invariance without the need of inertial sensors, which therefore can be used for surfaces at any orientation including objects changing their orientation over time. We also show how this approach can additionally benefit from gravity measurements by constraining the set of features to match against depending on the orientation of the camera in section 5.3.3.

Constraining reference features to match against is particularly important in large-scale outdoor environments when the reference dataset becomes very large. Compared to the gravity-aligned objects described above, outdoors not only the orientation with respect to gravity is known, but also the position (and scale) of landmark features is static and known and can be utilized, which is exploited in section 5.4.

5.1 Gravity-Aligned Feature Descriptors

This section is based on the paper "Inertial sensor-aligned visual feature descriptors" [Kurz 11b] in which we propose to align the orientation of local feature descriptors with the gravitational force measured with inertial sensors. In contrast to standard approaches that gain a reproducible feature orientation from the intensities of neighboring pixels to remain invariant against rotation, this approach results in clearly distinguishable descriptors for congruent features in different orientations. Gravity-Aligned Feature Descriptors (GAFD) are suitable for any application relying on corresponding points in multiple images of static scenes and are particularly beneficial in the presence of differently oriented repetitive features as they are widespread in urban scenes and on man-made objects. We show with different examples that the process of feature description and matching gets both faster and results in better matches when aligning the descriptors with gravity compared to traditional techniques. In addition to the results of the initial paper mentioned above, this section contains additional evaluation results which were published in [Kurz 12a].

5.1.1 Introduction

To address the invariance to perspective distortions resulting from changes in the camera rotation, an image transformation is generally used to normalize the pixels in the region around a feature point. The descriptor is then computed based on this normalized region. It is critical that the normalized region for the same physical point in two images under different viewing angles is similar. The normalization is usually an in-plane rotation based on a feature orientation computed from the pixel intensities in the neighborhood of the feature point.

This way of orientation assignment results in problems in the presence of congruent or near-congruent features in different orientations as shown in figure 5.2 at the example of the corners of a window. The regions around the four corners that correspond to four different physical points would be ideally identical after normalization as shown in the left column resulting in indistinguishable descriptors. This was also discussed in the problem statement of this thesis in section 3.1.

Increasingly more mobile devices are equipped with inertial sensors. While mobile phones provide interfaces that make accessing the measured gravity vector corresponding to a camera image very easy, digital cameras at least store a coarse orientation in steps of 90° with digital photos using the EXIF format which might support more precise measures in the future. Because the orientation of all static objects with respect to gravity is constant, as discussed above, it is worth evaluating using this orientation as a reference in the feature description process.

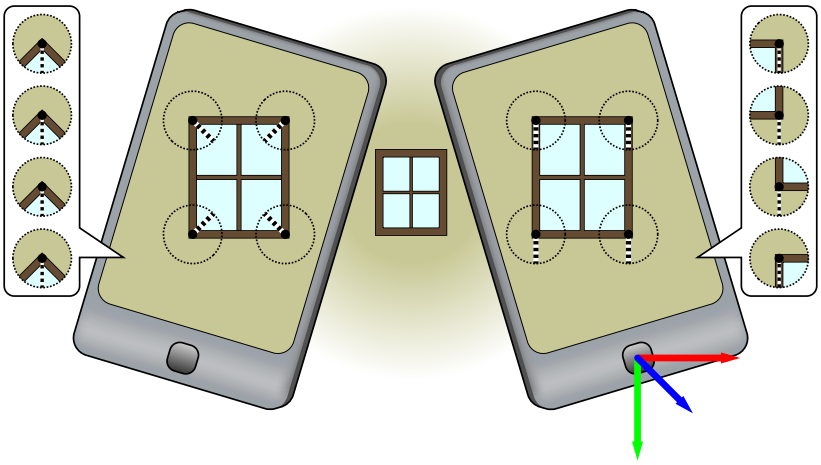


Figure 5.2: Schematic sketch of the effect of Gravity-Aligned Feature Descriptors (right) compared to regular techniques (left).

5.1.2 Proposed Method

We present a new approach for creating local visual feature descriptors suited for static and close to upright surfaces. It outperforms standard approaches as it overcomes ambiguities resulting from congruent or near-congruent features with different orientations without constraining the camera movement. The result is an improved distinctiveness and therefore improved precision-recall characteristic. In addition, we present means to speed-up both the descriptor computation and the process of matching two or more sets of features.

The proposed technique can be applied to any existing feature description method based on a normalized region around the feature point and is suitable for many applications including image classification, 6-DoF camera pose tracking, and 3D scene reconstruction.

A feature descriptor is generally built such that two features corresponding to the same physical point in different images result in close descriptors with respect to a similarity measure, thanks to invariance of the description method. While two features corresponding to two different physical points are supposed to result in distinct descriptors. Often, the descriptor computation is composed of two steps namely spatial normalization of the region around the feature and the actual computation of the descriptor as a function of the normalized region, e.g. [Lowe 04]. Other advanced methods exist, e.g. Hessian-Affine regions [Miko 05b], but in the simplest case the normalization only consists of an in-plane rotation according to the feature orientation. This orientation is defined based on pixel intensities, for instance based on the direction of the strongest gradient in a limited region around the feature. In case there are multiple dominant gradient directions the normalization and following description is carried out for all of them. The actual descriptor is finally a function of pixel intensities in the normalized region, usually based on histograms, that gives a multi-dimensional vector which will be referred to as feature descriptor.

5.1.2.1 Proposed Feature Orientation Assignment

We propose to align the orientation of local feature descriptors with the projection of the gravitational force in the coordinate system of the camera image. Figure 5.2 illustrates a window as an example of a real static object with vertical surfaces. The camera of the left mobile phone captures the window and its four corners act as feature points for which a descriptor is computed. Due to invariance to rotation, as schematically illustrated with the normalized regions of the features in the left column, a regular feature descriptor would describe these features in exactly the same way making them indistinguishable. In a real-world setup, the descriptors will not be identical but very similar and therefore virtually indistinguishable. Consequently, the probability that two features are matched and considered to correspond to the same physical point (even though they in fact correspond to two different physical points) is high for such urban scenes. Too many mismatches may result in a failure of systems using feature descriptors.

The phone on the right in figure 5.2 is equipped with a 3-axis-accelerometer, potentially in combination with a gyroscope, and provides the gravity vector expressed in the device coordinate system. The projection of this three-dimensional vector into 2D image coordinates acts as the orientation of the four feature descriptors of the corners of the window. As illustrated in the right column the normalized regions around the four features are clearly distinct resulting in distinct feature descriptors. Thus, the proposed method reduces the probability of mismatches by improving distinctiveness.

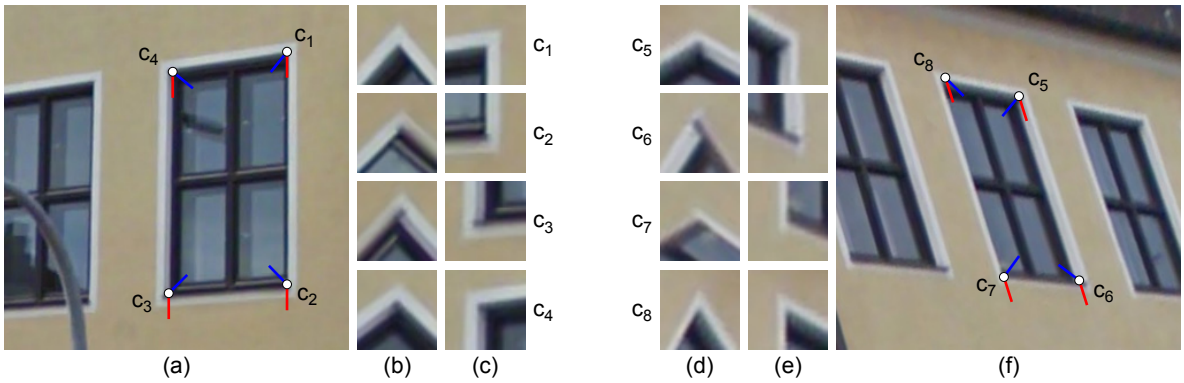


Figure 5.3: Normalized regions in two images (a,f) of the same window from different viewpoints. Normalization was performed based on image gradients (b,d) and gravity (c,e).

Figure 5.3 shows the normalized regions around window corners in two images (a,f) where the normalization was performed based on image gradients (b,d) and based on the direction of gravity (c,e). Firstly, it is apparent that the patches in column (b), which correspond to different physical points, are much more similar to each other than those in column (c), which use the normalization we propose. The figure further shows that our proposed method results in similar normalized regions around a single physical point (i.e. corner of a window), when imaged from different viewpoints, when comparing figure 5.3 (c) and figure 5.3 (e).

Note that the direction of gravity is not necessarily uniform across the camera image, as can be seen in figure 5.6 on the bottom left. Given the normalized gravity vector $\mathbf{g} = [g_x,\, g_y,\, g_z]^\top$, where $\|\mathbf{g}\| = 1$, and the (3×3) intrinsic matrix $\mathbf{K}$ of the camera, the projected 2D gravity direction $\mathbf{d} = [d_u,\, d_v,\, 0]^\top$ at a pixel $\mathbf{p} = [u,\, v,\, 1]^\top$ is computed as

$$\mathbf{d} = \mathbf{p}' - \mathbf{p}\,, \tag{5.1}$$

where $\mathbf{p}' = [u',\, v',\, 1]^\top$ can be computed from

$$[(1+g_z)u',\, (1+g_z)v',\, (1+g_z)]^\top = \mathbf{p} + \mathbf{K}\mathbf{g}, \qquad \text{and } \mathbf{K} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.2}$$

The fact that the lengths of $\mathbf{d}$ and $\mathbf{g}$ are arbitrary leads to a computationally cheap formulation of the direction as

$$\mathbf{d} \propto [g_z(p_u - u) + f_u g_x,\, g_z(p_v - v) + f_v g_y,\, 0]^\top\,, \tag{5.3}$$

where $[u_0,\, v_0]^\top$ is the principal point and f_u and f_v are the horizontal and vertical focal lengths of the camera. The orientation angle o_g of a feature is eventually computed as the arctangent of (d_v/d_u).

5.1.2.2 Approaches to Take Advantage of Gravity

The local orientation o_l of a feature computed from the intensities of neighboring pixels is usually computed such that it provides the same normalized region at any viewpoint and view direction. We propose three ways to take advantage of combining it with the global orientation o_g as the direction of gravity, cf. figure 5.4 a.

Regular Feature Descriptors With Relative Gravity Orientation In case the global orientation corresponding to the gravity measurement o_g is too coarse, we propose to use o_l as descriptor orientation as done in regular approaches and additionally store the relative global orientation: $o_{gl} = o_g - o_l$ for every feature as depicted in figure 5.4 b. Note that o_{gl} is theoretically constant and independent of the camera rotation. Similar to the sign of the Laplacian in SURF [Bay 08], we propose to use o_{gl} as a part of the descriptor and give it a higher priority. We use it to preclude the comparison of the descriptors for features whose relative orientation o_{gl} differs significantly. In fact, features with very different o_{gl} do not correspond to the same physical point. By comparing only features with a similar o_{gl}, the matching process is not only improved in terms of accuracy but also it can be speeded up significantly. In the evaluation section 5.1.3, this technique will be referred to as *regular fast*.

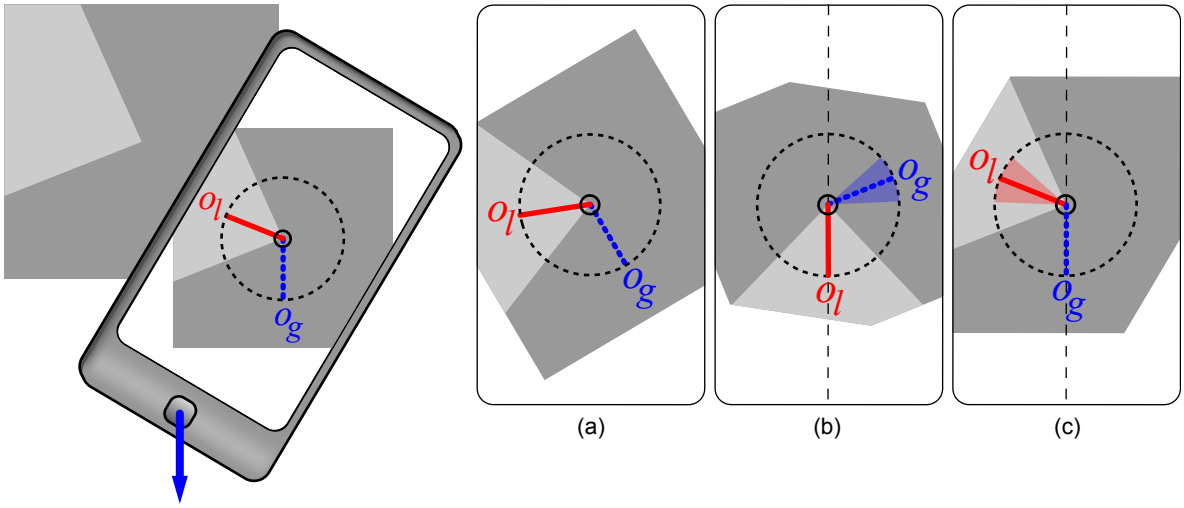


Figure 5.4: Each feature point has a local (red) o_l and global (blue) o_g orientation (a). Using the local orientation for descriptor alignment (b) leaves the relative global orientation as part of the descriptor. Alignment with the global orientation (c) allows to enrich the descriptor with the relative local orientation.

Gravity-Aligned Feature Descriptors One other possibility is to use the global orientation o_g of the feature for normalization instead of the local orientation o_l as shown in figure 5.4 c. This allows not only to improve the matching results, but also saves computational time since the computation of o_l can be skipped. This approach will be referred to as *GAFD*.

Gravity-Aligned Feature Descriptors With Relative Local Orientation If the orientation of gravity o_g can be considered accurate, we propose to use o_g as orientation for the region normalization and store o_{gl} with every feature. In case there are multiple dominant orientations o_l they are all stored relatively to gravity o_g. Here again, in the matching process, only the descriptors of those features that have at least one similar relative orientation o_{gl} need to be compared, cf. figure 5.4 c. This allows a faster matching with an improved accuracy thanks to gravity-alignment. We will call this approach *GAFD fast* in the following.

5.1.3 Evaluation Results Based on SIFT

This section reports on evaluations measuring the effect of the three approaches described above, i.e. *regular fast*, *GAFD*, and *GAFD fast*, compared to *regular* feature descriptors on the low-level matching performance of SIFT [Lowe 04] descriptors.

Figure 5.5: The four target images used in the matching evaluation.

5.1.3.1 Matching Precision for Vertical Surfaces

In order to measure the impact on the matching precision, we carried out experiments on two different mobile phones, the iPhone 3GS and the iPhone 4, providing a different level of gravity measurement accuracy. While the iPhone 3GS has a built-in 3-axis accelerometer the iPhone 4 has additionally a 3-axis gyroscope. We quantified the accuracy of the measured gravity for both devices based on 40 pictures of three vertical lines and the error distribution shows that the iPhone 4 ($\varnothing - 0.63^\circ \pm 0.82^\circ$) provides significantly better results than the iPhone 3GS ($\varnothing 2.8^\circ \pm 3.67^\circ$).

In the following experiments, we use the four planar target images that are depicted in figure 5.5.

Dartboard Its 20 radial sections make it a good example for many congruent features in different orientations.

Façade This urban scene has many repetitive structures both in different orientations and the same orientation.

Butterfly This is a natural (not man-made) scene.

Isetta This target represents man-made objects that do not have a large amount of repetitive features.

A (213×160) mm print-out of each target image has been attached onto a planar surface in an upright orientation using an electronic water level. Three sequences of eight photos each have been taken of each target. In the first sequence, the mobile phone underwent strong rotations about the viewing axis (*roll*) and moderate rotations about the other axes while in the second sequence, the

camera movement mainly consists of *pitch* rotations and in the third sequence *yaw* rotations dominate. Figure 5.7 shows a subset of the images used on the left. In addition, one picture of each target is taken from a straight perspective to act as reference image in the evaluation.

Because all photos have been taken with the phone in hand, some of them contain blur due to defocus or motion. With each picture, we store the measurement of the gravity vector at the time it was taken. All following computations are done offline on a PC.

First, we removed the effect of lens distortions from all images followed by resizing all query images to a resolution of (640×480) pixels for further processing. The four reference images have been manually rectified and resized to a resolution of (320×240) pixels which approximately matches the size the targets have in the query images.

The publicly available implementation of SIFT features in VLFeat [Veda 10] has been used to extract regular SIFT feature descriptors from all the images while a modified version, that in addition takes the 3D gravity vector as input, is used to extract gravity-aligned SIFT feature descriptors. In contrast to regular SIFT that describes a feature in multiple orientations in case there are multiple dominant gradient directions, the gravity-aligned version only describes each feature point once with the orientation of the projection of the measured gravity at the position of the feature point.

For each sequence of images the matching stage aims to find for each query feature extracted from a query image the corresponding feature in the reference image. Therefore, the nearest neighbor, i.e. the reference feature with the descriptor that has the smallest Euclidean distance to the descriptor of the query feature, is found using exhaustive search.

The ground truth to classify matches to be correct or false has been generated by manually defining the position of the four corners of the target in each query image, as described in section 4.1. Based on these correspondences, a homography is computed for each query image that maps each point on the target from the query image to the reference image. For each match, this homography is used to warp the position of the query feature to the reference image. If the Euclidean distance between the warped position and the position of the matching reference feature is below a chosen threshold of 6 pixels, the match is classified as correct. Otherwise, it is considered wrong, as illustrated in figure 5.6 in red.

As in [Miko 05b], we evaluate and compare $1 - precision$ vs. $recall$ where

$$1 - precision = \frac{\#false\ matches}{\#correct\ matches + \#false\ matches} \quad \text{and} \tag{5.4}$$

$$recall = \frac{\#correct\ matches}{\#correspondences}. \tag{5.5}$$

We measure the precision-recall characteristics of four different approaches, namely *SIFT regular* and the three approaches described in 5.1.2.2: *SIFT regular fast*, *SIFT GAFD*, and *SIFT GAFD fast*.

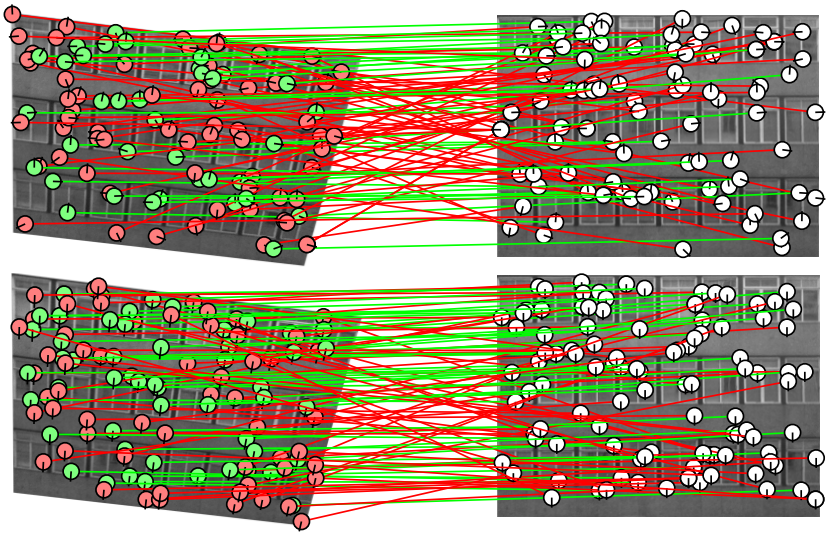


Figure 5.6: Correct (green) and false (red) feature matches between a query (left) and reference (right) image using SIFT regular (top) and SIFT GAFD (bottom) resulting in 15% more correct matches.

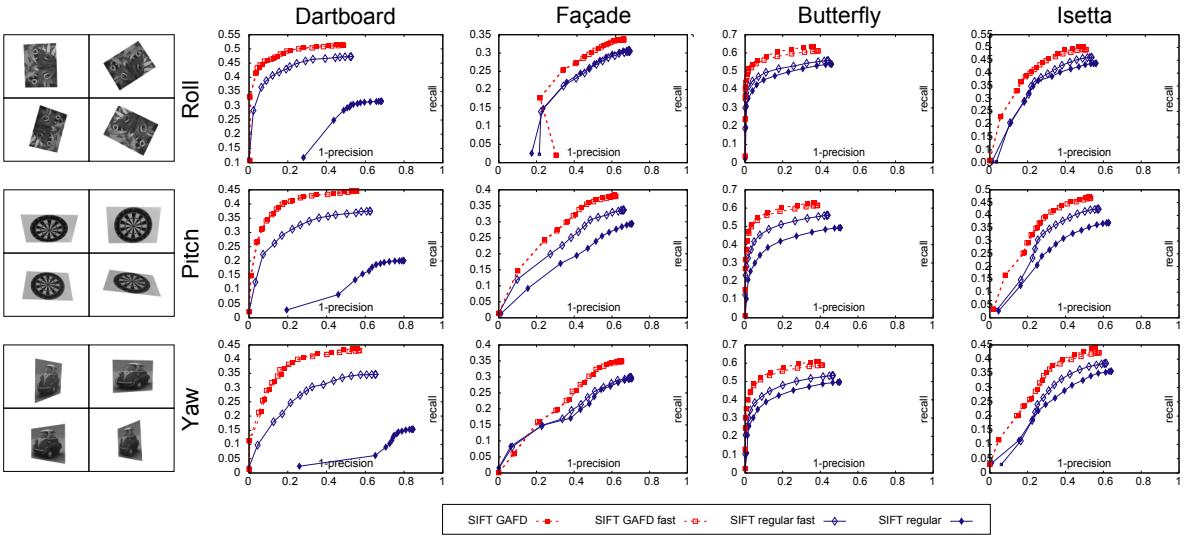


Figure 5.7: The precision-recall plots for the results on the iPhone 4 clearly show that all three proposed techniques outperform regular SIFT.

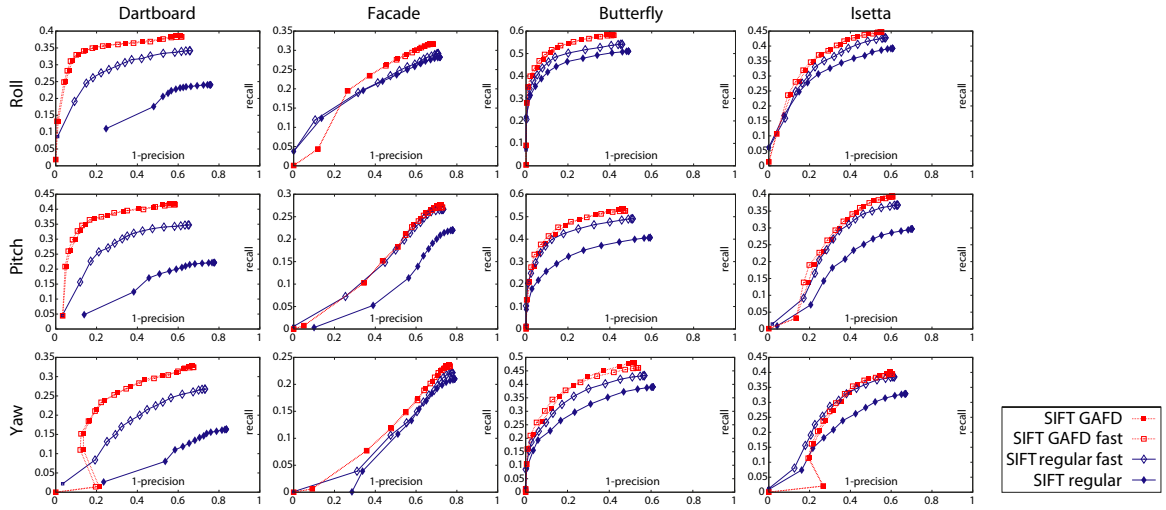


Figure 5.8: The precision-recall plots for the results on the iPhone 3GS confirm that our approaches outperform SIFT.

In the two *fast* techniques, we avoid matching features whose relative orientations o_{gl} differ by more than 0.5 radians which corresponds to $\sim 29°$. Assuming equally distributed feature orientations this saves around 84% of the descriptor comparisons needed.

The samples used to plot the precision-recall curves in figure 5.7 were gained using different subsets of all matches whose descriptors' distance is below a threshold for a set of different thresholds. The results for the iPhone 4 in figure 5.7 and iPhone 3GS in figure 5.8 show similar behavior although the sensor accuracy differs significantly.

In nearly all cases, for all targets and for all kinds of camera movement, *SIFT GAFD* performs best closely followed by *SIFT GAFD fast*. Also, constraining the nearest neighbor search for regular SIFT features (*SIFT regular fast*) increases precision but does not give as good results as *SIFT GAFD*. As expected, the impact of gravity-alignment is particularly high for the *Dartboard* target as it has many congruent features in different orientations. It might be surprising that the impact of GAFD is stronger in the pitch and yaw sequences than in the roll sequences. The reason is that the orientation assignment of SIFT is more invariant to roll rotations than pitch and yaw rotations as roll rotations do not change which physical points around the feature point are considered in the orientation assignment.

Considering both matching precision and computational time, *SIFT GAFD fast* gives the best results as the precision is similar to *SIFT GAFD* while it saves around 84% expensive descriptor comparisons.

5.1.3.2 Performance Analysis for Non-Vertical Surfaces

While the concept described above is designed to be used with vertical surfaces, this section evaluates how GAFD performs if the surfaces where features are extracted are not vertical. Therefore, the

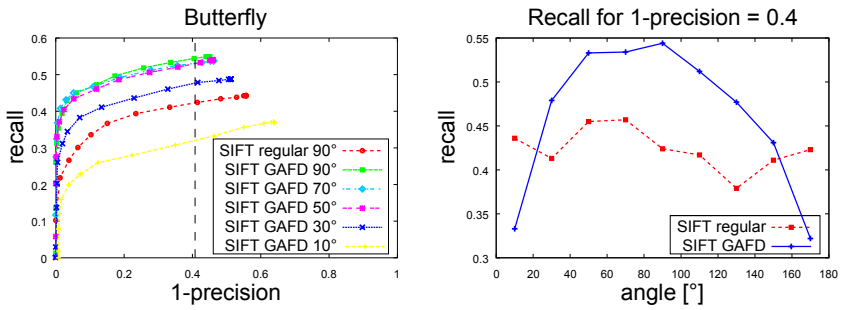


Figure 5.9: GAFD improves the matching also for non-vertical surfaces.

Butterfly target shown in figure 5.5 has been attached to a tiltable surface at different orientations that were measured with an electronic water level. For each orientation, we took three images at different roll orientations of the camera keeping the camera image plane approximately parallel to the plane of the image target.

Figure 5.9 shows on the left the precision-recall characteristic for different angles where 90° represents a vertical surface. The right plot displays the recall for a fixed (1-precision) of 0.4 where it is visible that in this configuration *SIFT GAFD* outperforms *SIFT regular* in the range of $90° \pm 60°$.

5.1.4 Evaluation Based on a Custom Descriptor: Mobile48

In the following, we will evaluate the impact of gravity-alignment on a custom feature descriptor that (as opposed to SIFT) is designed to run in real-time on off-the-shelf camera phones. First, we will evaluate how the quality of feature matches improves compared to regular descriptors followed by an evaluation that focuses on feature-based localization of a template, i.e. a planar object. As this is crucial to initialize camera pose tracking, it is an important step in handheld video-see-through AR applications. Many such applications rely on pose estimation from features on vertical static surfaces where GAFD could be applied.

5.1.4.1 Evaluation of Mobile48-GAFD-Based Feature Correspondences

For evaluation of the feature correspondences gained with Gravity-Aligned Feature Descriptors, we use the four template images shown in the upper row of figure 4.2. A print-out of each template at (160×120) mm has been attached to an untextured vertical surface in an upright orientation using a digital water-level. For each template, we capture sets of ten camera images each with an iPhone4 at a resolution of (480×360) pixels and store them along with the corresponding gravity vector measurement for each frame. As in previous evaluation, we store one set where the mobile phone mainly undergoes rotations about the viewing axis (*roll*), and second set where the camera movement mainly consists of *pitch* rotations. As illustrated in figure 5.10, *yaw* rotations dominate in the third set of camera images. Additionally, we take one fronto-parallel image of every template with the

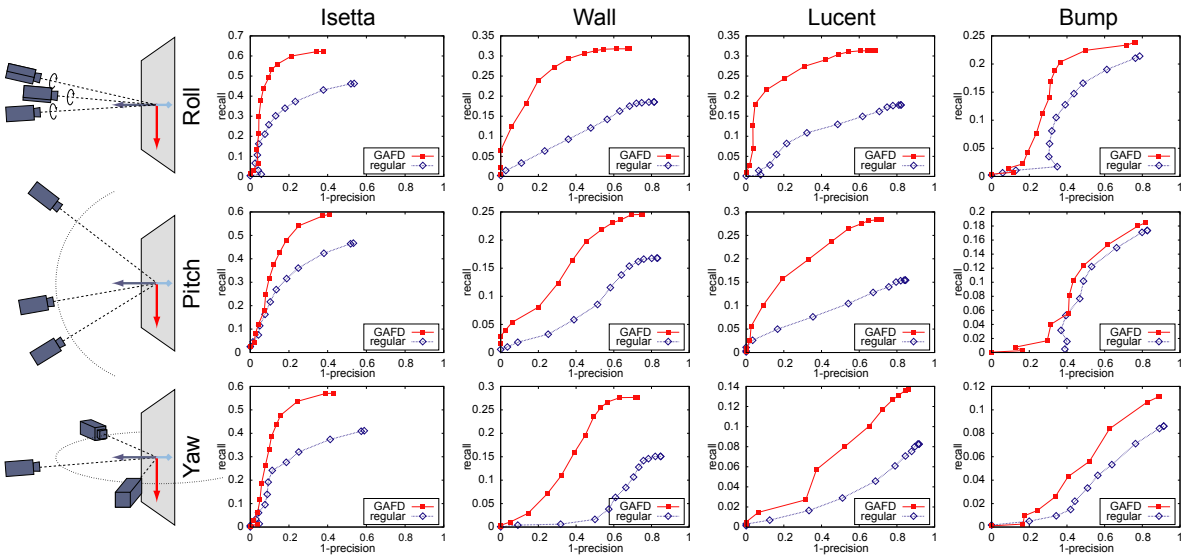


Figure 5.10: Precision-recall characteristic of GAFD and regular feature descriptors. GAFD outperforms the regular approach for all templates and camera motions.

same camera. These reference templates have a resolution of (320×240) pixels which approximately matches the average size of the template in the image sets.

In order to compute 2D-2D correspondences between points in the reference templates and corresponding points in the camera images, we first extract 200 features at different scales from all captured images using image pyramids. We compare two versions of the feature descriptor that only differ in the feature orientation assignment step. While the *regular* algorithm uses up to 6 directions of dominant gradients in a region around the feature as orientation and computes one descriptor for each orientation, the *GAFD* algorithm uses the projected gravity vector as descriptor orientation.

After quantization of the orientation angle, the descriptor uses the assigned orientation to perform an in-plane rotation of its coordinates. Finally, a 48-dimensional vector is computed based on gradient histograms which will be referred to as the feature descriptor. The similarity measure used to compare feature descriptors is the Euclidean distance. For each feature in a current camera image, a matching reference feature from the reference template is found using exhaustive nearest neighbor search. Whether a match is correct or not is determined by transforming the position of the current feature into the coordinate system of the reference template. This is done using a homography computed based on the manually defined positions of the four template corners in the current image, cf. section 4.1. If the transformed position of the current feature and the position of the matched reference feature differ by less than 5 pixels, the match is considered correct.

Again, we compute *recall* vs. $1 - precision$ for each image set to evaluate the different kinds of feature descriptors. The resulting precision-recall characteristic for the 12 different image sets is shown in figure 5.10. The samples for the plots have been created by using subsets of all matches whose

Euclidean distance of the descriptors is below a particular threshold for different thresholds. The plots show, that GAFD provides an improved precision-recall characteristic for all templates and all kinds of camera motion in the test. This confirms the results of the previous evaluations in section 5.1.3 using SIFT feature descriptors [Lowe 04] and different template images.

5.1.4.2 Evaluation of Mobile48-GAFD-Based Template Localization without Ground Truth

We showed that using GAFD improves recall-precision of SIFT and our custom descriptor Mobile48. In the following, we will focus on how it affects feature-based localization of a template, which is crucial to initialize camera pose tracking for handheld video-see-through AR applications. Many such applications rely on pose estimation from features on vertical static surfaces. Examples include posters or billboards, cars, and increasingly more important building façades for large-scale self localization in urban outdoor environments.

To ensure a fair comparison, we recorded image sequences with the measured gravity vector for each image and then run the evaluations offline on a PC. Nevertheless, the detection algorithm we use for evaluation is optimized to run on mobile devices in real-time. In an offline step, the algorithm is provided with a reference image of the template to detect. We use a fronto-parallel photo of the template captured with the same camera that took the image sequences. From this reference image, we extract up to 200 features and describe them with reference feature descriptors. For each current image from a sequence, we first detect and describe current features.

The next stage then finds the nearest neighbor (with the closest descriptor in terms of Euclidean distance) of each reference feature in the set of current features using exhaustive search. This results in a set of 2D-2D correspondences, of which we only keep those where the ratio of distances of the best and the second best match is above a particular threshold. The remaining correspondences are used by PROSAC [Chum 05] to find a homography that maps the current image to the reference image. If such homography could be found, we eventually refine it using Inverse Compositional [Bake 01] image registration. This homography, together with the intrinsic parameters of the camera, can then be used to compute the pose of the camera to enable rendering registered 3D models into the camera image.

In our evaluation, we compare two versions of the localization algorithm that only differ in the feature orientation assignment step. While the *regular* algorithm uses up to 6 directions of dominant gradients in a region around the feature as orientation and computes one descriptor for each orientation, the *GAFD* algorithm computes the orientation as the projection of the gravity vector onto the image plane, as described above.

We judge the properness of a detection and the computed homography by computing the Zero-mean Normalized Cross Correlation (ZNCC) of the reference image and the current image warped with the

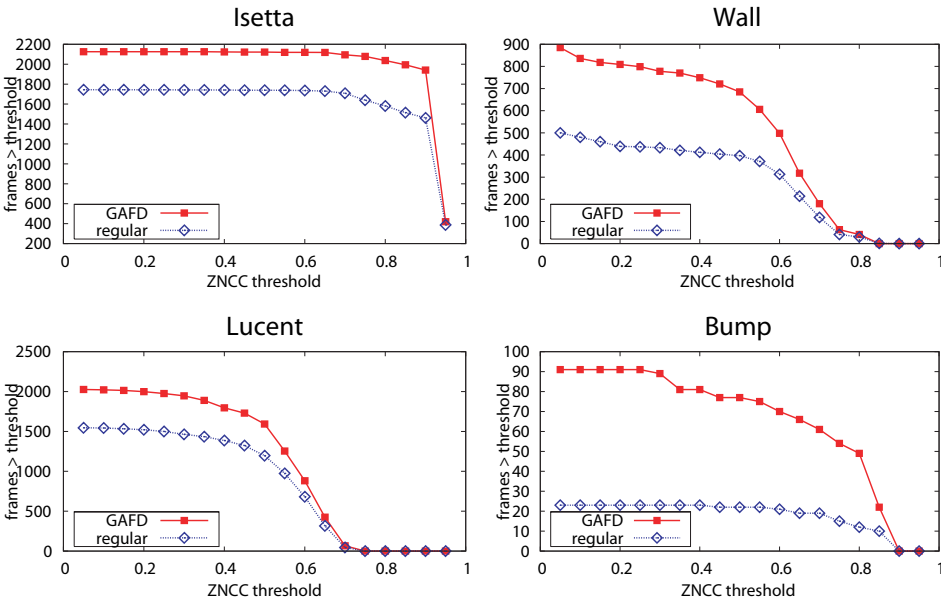


Figure 5.11: The plots show the number of images, where a localization could be performed that results in a ZNCC greater than a particular threshold as a function of the threshold used. GAFD provide better results for all templates in figure 4.2 than regular feature descriptors.

homography. Our assumption is that with an increasing ZNCC value, the accuracy of the homography increases, as explained in section 4.2. If the algorithm does not find a homography the ZNCC is set to -1.0.

Again the printouts of the four templates in the upper row of figure 4.2 are used and for each of them, we record a sequence of 480 camera frames with an iPhone4 at a resolution of (480×360) pixels and store the measured gravity for each frame. As reference image, we take one fronto-parallel image of the template with the same camera.

To reduce the effect of the randomization in PROSAC [Chum 05], we evaluated each sequence five times, resulting in 2,400 ZNCC computations in total. The results in figure 5.11 show the number of images, where a localization could be performed that results in a ZNCC greater than a particular threshold for different thresholds between 0 and 1. Even though the localization performance differs significantly among the different templates, the average ZNCC improved clearly when using GAFD for all of them. On average, over all templates and all thresholds plotted in figure 5.11, the number of correctly localized images is slightly more than doubled when using GAFD compared to regular descriptors.

As one important application of GAFD is tracking initialization in urban outdoor environments, we also carried out the same experiments explained above on image sequences of real building façades. For each sequence of 480 images, a reference image has been manually rectified and extracted from an additional image. The façades and the areas used as reference are illustrated in figure 5.12 with

a yellow frame. Again, each image has been evaluated five times to reduce randomization effects. The results are shown in figure 5.12. While the improvements have a high variance over the different sequences, our approach outperforms regular feature descriptors in all cases. The average increase in detection for these sequences of over 50% clearly shows the potential and applicability of GAFD for outdoor tracking, which we consider one of the most exciting areas in the near future of handheld AR.

5.1.4.3 Evaluation of Mobile48-GAFD-Based Template Localization with Ground Truth Poses

The template localization framework explained above was also evaluated using the template benchmark dataset explained in section 4.3. This dataset not only uses all eight template images shown in figure 4.2, more diverse motions and even changes in illumination, but most importantly comes with ground truth poses for every single of the 48,000 current camera images. Even though there are no inertial sensor measurements corresponding to the images available, we have shown in section 4.3 that synthetic gravity vectors computed from the ground truth poses provide comparable results.

Gravity-Aligned Feature Descriptors (GAFD) are designed for static and (close to) vertical surfaces. Therefore, we define the synthetic gravity vector $\mathbf{g}_{synth}$ as if the template was located on a vertical surface in an upright orientation. This means that the gravity vector corresponds to the negative y-axis ($-\mathbf{y}_w$) of the world coordinate system associated to the print-out of the template, cf. figure 4.3. Given the ground truth pose, this vector is transformed to the camera coordinate system by multiplying it with the rotational part of the pose ${}^c\mathbf{R}_w$ and is then used as gravity vector. As explained in section 4.3, we use a degradation model adding noise and a delay to the ground truth gravity vector in order to create synthesized gravity vectors with similar properties as real gravity measurements.

The results of the above mentioned template localization framework using the synthesized gravity measurements can be found in table 5.1. For every sequence, the ratio of correctly localized camera frames is given for the localization framework in two different configurations. The second column contains the ratios for a system using regular feature descriptors. The results in the third column have been retrieved using Gravity-Aligned Feature Descriptors. For nearly all sequences, we observe that GAFD enables correct template localization in more frames than regular descriptors. However, in four sequences the ratio decreases when using GAFD. We do not have an explanation for that but given that the overall performance significantly improves using GAFD, we are convinced it is useful for handheld AR.

These results are put into comparison with the result when using Gravity-Rectified Feature Descriptors, which will be explained in the next section, in table 5.3.

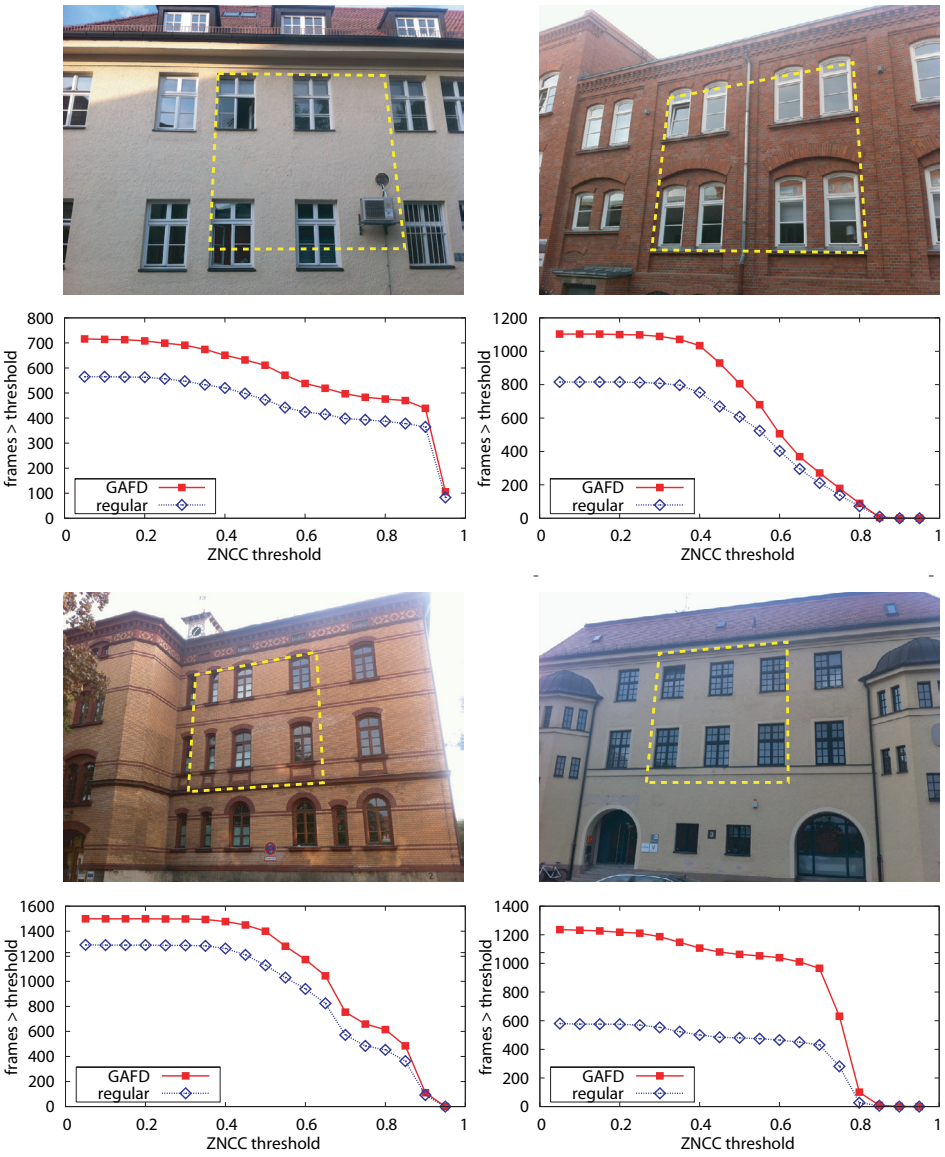


Figure 5.12: The four building façades used in the evaluation provide a good amount of repetitive features. The yellow frame indicates the individual areas used as reference template for feature-based localization. The corresponding plots show the number of images for which the algorithm could provide a transformation that results in a ZNCC greater than a particular threshold as a function of the threshold used. We clearly see that, with respect to building façades in urban outdoor environments, for any ZNCC threshold that the algorithm would use to decide about a correct localization, GAFD outperform regular feature descriptors.

Method	regular	GAFD
Isetta (Angle)	**90.25%**	83.33%
Isetta (Range)	95.42%	**96.92%**
Isetta (Fast Far)	40.67%	**54.17%**
Isetta (Fast Close)	72.50%	**79.25%**
Isetta (Lighting)	98.92%	**99.33%**
Philadelphia (Angle)	**43.00%**	30.75%
Philadelphia (Range)	73.42%	**73.75%**
Philadelphia (Fast Far)	13.17%	**14.58%**
Philadelphia (Fast Close)	39.08%	**49.00%**
Philadelphia (Lighting)	77.08%	**80.25%**
Wall (Angle)	14.33%	**26.67%**
Wall (Range)	22.42%	**34.17%**
Wall (Fast Far)	2.50%	**5.67%**
Wall (Fast Close)	5.42%	**14.58%**
Wall (Lighting)	28.08%	**52.50%**
Grass (Angle)	1.33%	**1.58%**
Grass (Range)	0.50%	**0.67%**
Grass (Fast Far)	0.08%	**0.25%**
Grass (Fast Close)	0.50%	**0.58%**
Grass (Lighting)	**0.75%**	0.58%
Lucent (Angle)	19.58%	**25.08%**
Lucent (Range)	28.42%	**33.83%**
Lucent (Fast Far)	6.50%	**7.25%**
Lucent (Fast Close)	19.50%	**26.58%**
Lucent (Lighting)	47.50%	**62.75%**
Mac Mini (Angle)	10.75%	**24.08%**
Mac Mini (Range)	16.33%	**28.17%**
Mac Mini (Fast Far)	4.67%	**5.58%**
Mac Mini (Fast Close)	18.33%	**28.42%**
Mac Mini (Lighting)	22.75%	**41.75%**
Bump Sign (Angle)	**50.75%**	49.67%
Bump Sign (Range)	45.42%	**55.33%**
Bump Sign (Fast Far)	13.08%	**23.50%**
Bump Sign (Fast Close)	17.08%	**24.83%**
Bump Sign (Lighting)	49.67%	**65.58%**
Stop (Angle)	53.25%	**70.58%**
Stop (Range)	90.83%	**97.42%**
Stop (Fast Far)	24.75%	**31.50%**
Stop (Fast Close)	36.33%	**51.08%**
Stop (Lighting)	88.92%	**92.17%**
All sequences	34.60%	**41.09%**

Table 5.1: Ratio of correctly localized frames using GAFD based on synthesized gravity measurements compared to a regular approach. Bold numbers indicate the better approach for each row.

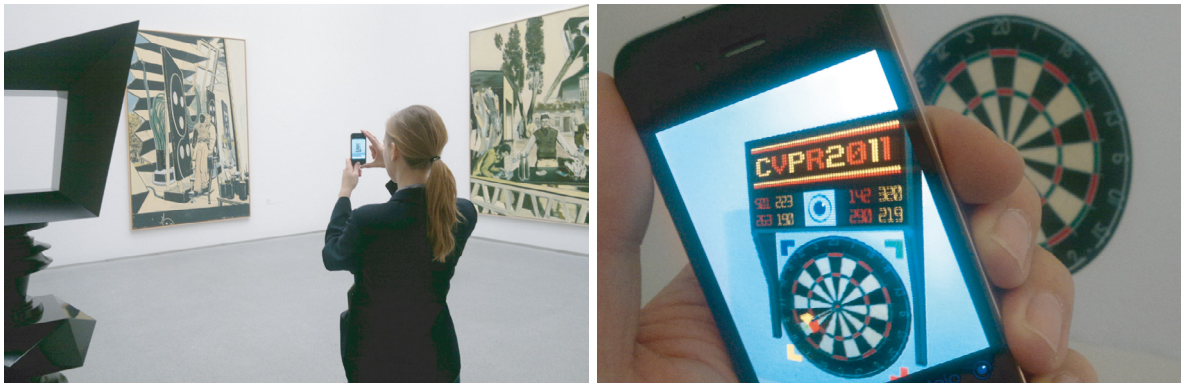


Figure 5.13: Using GAFD for object recognition (left) and Augmented Reality (right) on mobile phones.

5.1.4.4 Object Recognition on a Mobile Device

There are many possible applications for GAFD such as Augmented Reality and image classification on mobile phones (see figure 5.13). In the following, we measure the effect of GAFD on object recognition for a mobile museum guide. The user takes an image of an artwork, image recognition is performed on the mobile phone and information about the particular artwork is being displayed.

To this end, we randomly chose 100 artworks (paintings, drawings, photos and sculptures) in the Pinakothek der Moderne museum in Munich and took five pictures from different angles of each artwork. These pictures were stored along with the corresponding 3D gravity vector. For each artwork, we chose one picture to be the reference image while the other four images are used as query images.

All pictures have been resized to (320×240) pixels and we use our custom Mobile48 descriptor which performs in real-time on mobile devices. For each reference image, we extract 200 features and store them as reference features along with the ID of the artwork they belong to. From each query image, we also extract 200 features based on which we aim to detect the artwork by comparing the query features with the reference features. Since nearest neighbor search for each query feature in the set of $200 \times 100 = 20,000$ reference features is too expensive on a mobile device, we use the Best-Bin-First algorithm [Lowe 04] to find an approximate nearest neighbor instead. After finding it for each query feature, the detection result is eventually the ID that appears most frequently in the set of matched reference features.

The entire process has been carried out both with regular and GAFD feature descriptors. Table 5.2 shows the percentage of correctly recognized objects in several subsets of the dataset of different sizes. We observe the detection rate increasing on average by over 15% when aligning the feature orientation with gravity. This clearly shows an improvement over standard approaches in a real-world application.

# objects	recognition rate regular	recognition rate GAFD
20	73.75%	82.50%
40	76.88%	90.00%
60	68.33%	84.58%
80	70.94%	86.56%
100	71.25%	87.75%

Table 5.2: GAFD improves the recognition rates of a mobile guide.

5.1.5 Conclusions

The evaluation results clearly show the improvement of GAFD on the precision of feature matches, on template localization, and in an object recognition application on a mobile phone. The proposed method outperforms classical approaches without introducing any drawbacks since even the computational costs are reduced. This approach is capable of distinguishing repetitive visual features that only differ by their absolute orientation, as demanded in section 3.1.1. Of course, the field of applications for GAFD is limited to the presence of an inertial sensor and does not work for features on surfaces that are parallel to the ground plane or close to it. However, we believe that there is a variety of applications where GAFD are useful.

Figure 5.14 shows how GAFD has been successfully used for tracking 3D objects, which in this example form a miniature city. In section 5.4, we will use GAFD in combination with a more advanced approach, to enable camera localization outdoors with respect to a 3D model of the environment. Besides real-time applications on handheld devices, our proposed *regular fast* approach can also be applied to information retrieval on image databases, such as flickr, storing a coarse orientation with respect to gravity in their EXIF data.

While this section was concerned with the description of features on (close to) vertical surfaces, and how a measurement of gravity can aid their description, the following section will focus on horizontal surfaces. Examples for horizontal surfaces include the floor or (planar objects located on) tabletops. In that case, the gravity vector is perpendicular to the surface, which opens up different novel approaches to aid the description of visual features.

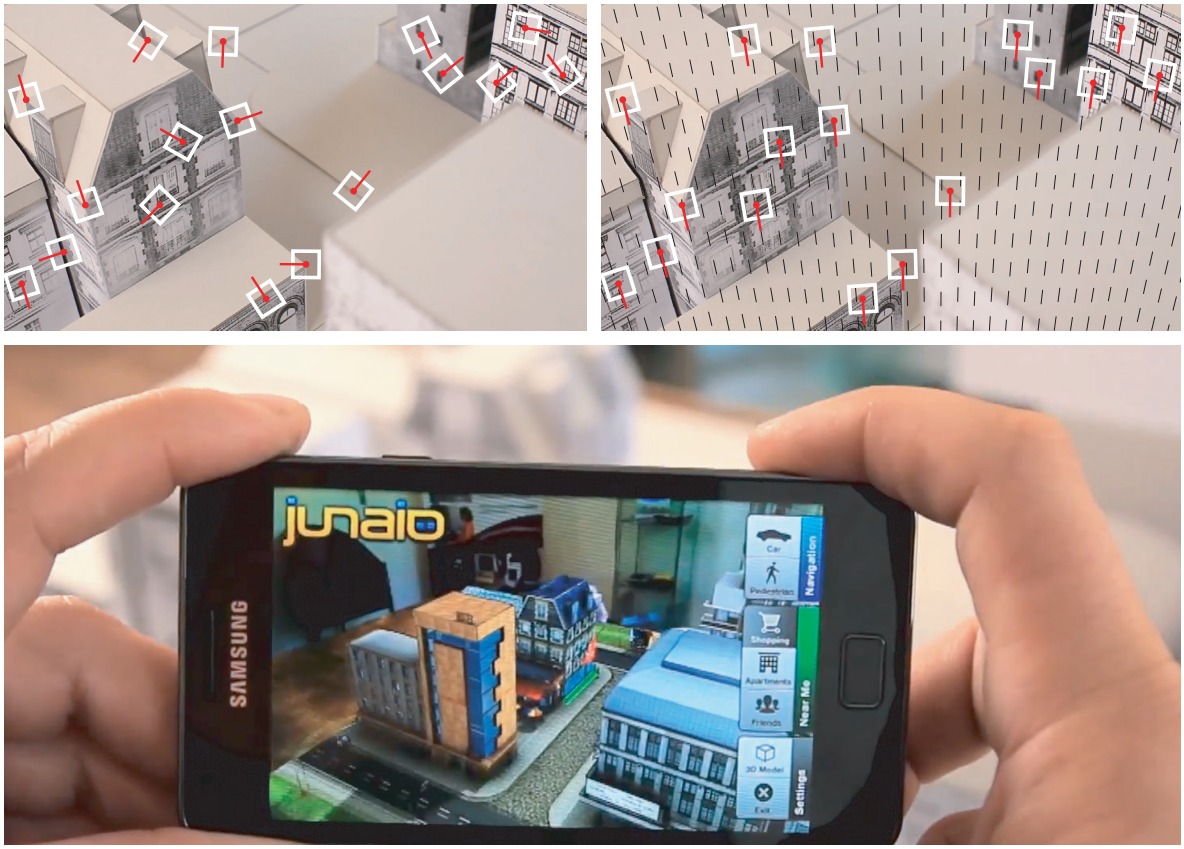


Figure 5.14: Orientation assignment using a regular approach (left) and Gravity-Aligned Feature Descriptors (right), which have also been employed for tracking 3D objects (bottom).

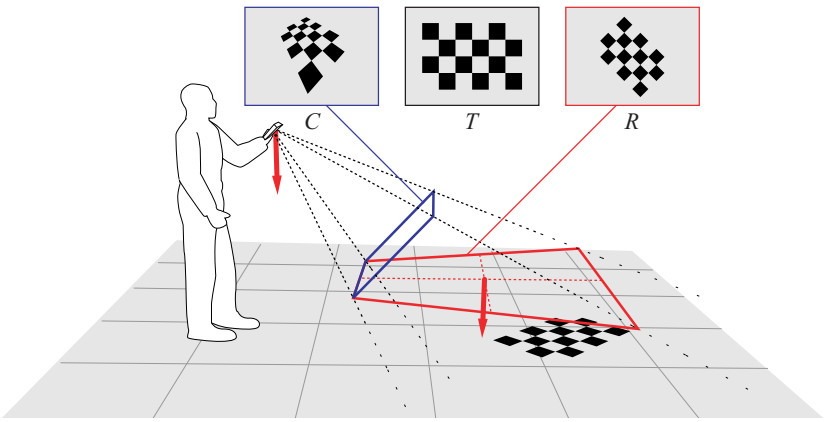


Figure 5.15: By projecting the camera image C onto a virtual image plane perpendicular to the measured gravity, we gain a synthetic gravity-rectified view R of a template T which is located on a (close to) horizontal plane.

5.2 Gravity-Rectified Feature Descriptors

This section is based on work, which has been previously published in "Gravity-Aware Handheld Augmented Reality" [Kurz 11a] and "Handheld Augmented Reality involving gravity measurements" [Kurz 12a]. It is concerned with how the description of features located on horizontal surfaces can be improved given a gravity vector, which in this case is perpendicular to the surface.

5.2.1 Introduction

As can be seen from some of the examples in figure 5.1, an important fraction of handheld AR applications is based on tracking the camera pose with respect to a planar object on a horizontal plane. Examples include games or marketing applications that track books, magazines, menus in a restaurant, prints on the floor or product packages on a table. This section describes how the description of features can be improved knowing that the object they correspond to is located on a horizontal plane and having a measure of the direction of gravity.

5.2.2 Proposed Method

We introduce a virtual image plane which is perpendicular to the measured gravity vector as illustrated in figure 5.15. Warping a camera image C onto this virtual image plane results in a gravity-rectified image R that appears as if it was taken in a camera orientation where the optical axis is aligned with the gravity vector. While the original image C shows perspective distortions with respect to the template (reference) image T, the gravity-rectified image R does not. A very similar approach is used in [Lee 11] to rectify reference templates located on horizontal surfaces offline before tracking.

In our implementation, the gravity image rectification is done with inverse warping in order to avoid having holes in the gravity-rectified image. Therefore, we compute a matrix $\mathbf{H}$ that aligns the gravity vector $\mathbf{g}$ with the optical axis of the camera $\mathbf{z} = [0,\ 0,\ 1]^\top$.

We assume the z-component of the gravity vector to be non-zero. The case $g_z = 0$ happens only when the optical axis is perpendicular to the gravity vector. Since we assume that the tracked objects are on (close to) horizontal planes, the assumption $g_z \neq 0$ is generally valid when the tracked objects are in the camera field of view. Now, let $\mathbf{g}_1$ and $\mathbf{g}_2$ be two vectors defining a plane with a normal $\mathbf{g}$.

These two vectors can be defined as follows:

$$\mathbf{g}_1 = [-g_z,\ 0,\ g_x]^\top \text{ and } \mathbf{g}_2 = \mathbf{g} \times \mathbf{g}_1. \tag{5.6}$$

The (3×3) matrix $\mathbf{H}$ defined as

$$\mathbf{H} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix}^{-1} \tag{5.7}$$

transforms the gravity vector $\mathbf{g}$ into a vector aligned with $\mathbf{z}$:

$$\begin{aligned} \mathbf{H}^{-\top}\mathbf{g} &= \left(\begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix}^{-1} \right)^{-\top} \mathbf{g} && (5.8) \\ &= \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix}^{\top} \mathbf{g} && (5.9) \\ &= [0,\ 0,\ g_z]^\top \propto \mathbf{z} && (5.10) \end{aligned}$$

Given the matrix of intrinsic parameters $\mathbf{K}$, transforming every captured image using an inverse warping with the homography $\mathbf{G}$ defined as

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix} \mathbf{K}^{-1} \tag{5.11}$$

allows rectifying perspective distortions of any plane orthogonal to the gravity vector. This means that when a tracked object is lying on a horizontal surface, its fronto-parallel reference template is transformed in the gravity-rectified image with a similarity transformation. Our implementation uses $\mathbf{G}' = \mathbf{K}[\ \mathbf{g}_1 \quad \mathbf{g}_2 \quad \mathbf{z}\ ]\mathbf{K}^{-1}$ instead, which provides the same property if the camera intrinsics do not contain any shear and pixels are quadratic, but in contrast to $\mathbf{G}$ results in an image in the same domain as the original one.

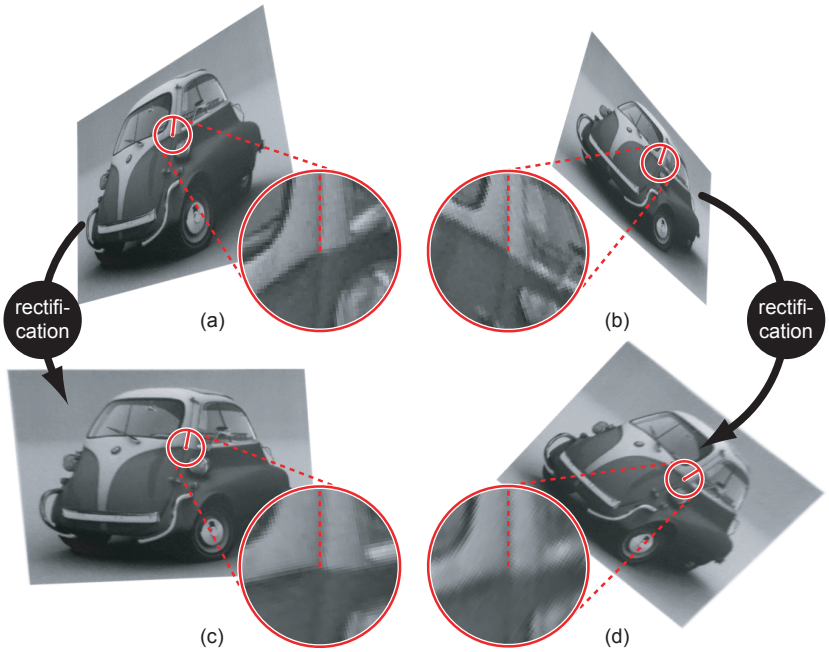


Figure 5.16: The pixel intensities in the region around features corresponding to the same physical point on a horizontal surface become more similar (c,d) after gravity-rectification of the camera image compared to the original images (a,b).

5.2.3 Evaluation of GREFD-Based Feature Correspondences

In the following, we evaluate the impact of this rectification to the recall-precision characteristic of a local feature descriptor. Inertial sensor-based rectification does not increase distinctiveness of congruent features in different orientations (in contrast to gravity-alignment as described in the previous section 5.1). The improvement of Gravity-Rectified Feature Descriptors (GREFD) over standard approaches is that, due to the rectification, the region around each feature point, which is used to compute its orientation and the actual descriptor, is more similar, even under steep viewing angles. Figures 5.16 a and 5.16 b show the region around features corresponding to the same physical point in two different views. When comparing this with the regions around the corresponding features in the rectified camera images (figures 5.16 c and 5.16 d), it is clearly visible that the regions are more similar (ZNCC: 0.7065) compared to the regions in the original images (ZNCC: 0.2612). Below, we will show that this improved invariance also results in more similar descriptors which leads to an improved recall-precision characteristic. A similar effect is achieved in [Wu 08], where the normal of a surface is reconstructed from the local geometry in a range image instead of measuring gravity.

The feature descriptor used in this evaluation equals the regular feature descriptor in the previous section. It is optimized to run in real-time on handheld devices and referred to as Mobile48. The orientation assignment is based on the dominant gradient directions. Figure 5.17 illustrates the support regions used for orientation assignment and description of the features with a white quadrangle.

Figure 5.17: Visualization of the support regions of Gravity-Rectified Feature Descriptors on a desk.

These regions correspond to squares in the gravity-rectified image. The orientation of each feature is illustrated by the red line.

As in 5.1.4.1, we use the four template images in the upper row of figure 4.2. The print-outs are first rigidly attached to a horizontal surface. For each template, we take two sets of ten images at a resolution of (480×360) pixels each with an iPod touch 4G and additionally store the gravity measurement for each image. While in the first set, the viewing axis of the camera is *close to perpendicular* to the template plane, the second sequence images the template under *steep angles*. Our hypothesis is that the impact of inertial sensor-based rectification is bigger on the latter group of sets than the first group. Given the images and the corresponding gravity vectors, we synthesize gravity-rectified images R using inverse image warping, as explained above. As image warping is computationally expensive, we use both nearest neighbor (*GREFD NN*) and bilinear (*GREFD BIL*) interpolation and evaluate the impact of the technique used. Two results of the rectification process using bilinear interpolation are shown in figure 5.16. In addition to the image sets, we take one reference image per target as a fronto-parallel view.

Analogous to section 5.1.4.1, we detect and describe 200 features in each camera image. The nearest neighbor with respect to the Euclidean distance of every current descriptor in the set of reference descriptors is determined using exhaustive search. A feature match is considered correct if the position of the reference feature differs by less than 5 pixels from the corresponding current feature after transforming it into the reference coordinate frame. This transformation is computed based on the manually defined positions of the four template corners in every current image.

Again, we evaluate *recall* vs. $1 - precision$ for each set of images. The resulting characteristics for the eight different image sets are shown in figure 5.18. The samples for the plots have been created by using subsets of all matches whose Euclidean distance of the descriptors is below a particular threshold for different thresholds.

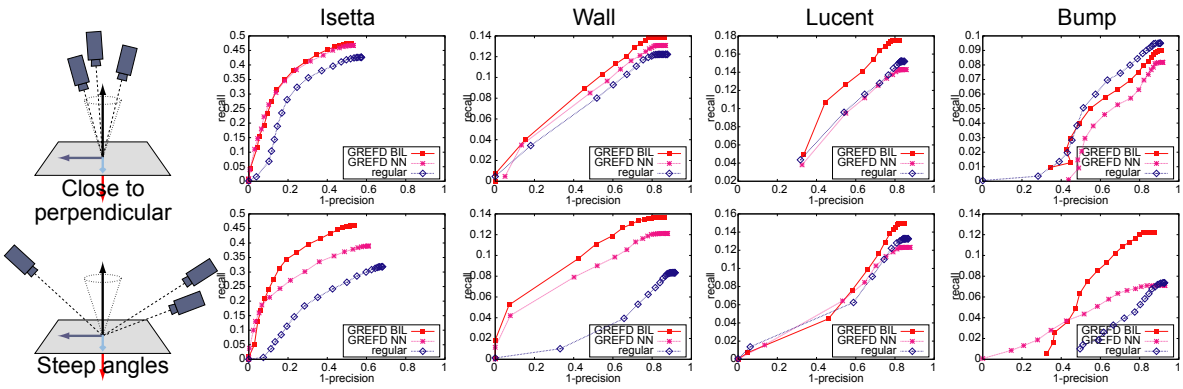


Figure 5.18: Precision-recall characteristic of feature descriptors for different target images. The upper row compares using the original image (*regular*), with gravity-rectified versions using nearest neighbor interpolation (*GREFD NN*) and bilinear interpolation (*GREFD BIL*) for images taken from close to perpendicular viewpoints. The lower row shows the same properties for images taken from steep angles. In general, gravity-rectification improves the characteristic when using nearest neighbor interpolation and even more significantly when using bilinear interpolation.

For the normally textured *Isetta* target and the highly textured *Wall*, the results meet our expectations: rectification using nearest neighbor interpolation improves the matching quality and using bilinear interpolation leads to even bigger improvements. Also it is clearly visible that the impact is bigger under steep angles than for close to perpendicular camera poses. Interestingly, the characteristic of the description of features on the repetitive *Lucent* target does not considerably change when rectifying with nearest neighbor warping. One explanation for that is that the negative impact of fine details getting lost in nearest neighbor interpolation and the positive impact of rectification cancel each other out. When using bilinear interpolation, the recall is improved for nearly all samples, but for this target, the impact is clearly stronger for close to perpendicular viewpoints compared to steep angles. The *Bump* sign, with its very low texture, shows a similar behavior for steep camera angles as the *Isetta* and *Wall* targets. However, rectification downgrades the quality independent of the interpolation method used for close to perpendicular viewpoints.

Looking at the results above, we propose an adaptive approach that chooses the procedure based on the current inertial sensor measurements. If the camera is oriented in a close to perpendicular angle, we do not use any rectification, as we could not report on significant improvements that would be worth the extra effort of warping the image. For steep angles however, using rectification clearly is desirable. We will evaluate different strategies for adaptive gravity-rectification in section 5.2.6.

It is important to keep in mind, that gravity-rectification may have side effects on the extraction of feature points. On the one hand, warping introduces new edges to the gravity-rectified image at the borders of the original camera image. However, feature point detection at these edges can be easily avoided, as their positions are known. On the other hand, it might happen, in particular under

steep angles, that relevant parts of the camera image project beyond the dimensions of the gravity-rectified image. While the test images used in this evaluation were taken such that the template is always entirely visible in the gravity-rectified camera image, the system used in the following scales the warping based on the steepness of the camera. An empirically found method uses the square-root of the z-component of the normalized gravity vector, g_z, as scale.

$$\mathbf{H}_{scaled} = [\ \mathbf{g}_1 \quad \mathbf{g}_2 \quad \sqrt{|g_z|}\mathbf{z}\ ]^{-1} \tag{5.12}$$

5.2.4 Evaluation of GREFD-Based Template Localization without Ground Truth

Figure 5.19 compares in the left column the localization results of the *regular* template localization algorithm explained in section 5.1.4.2 with a version using *GREFD* as feature descriptor. The goal is to localize print-outs of the templates in the upper row of figure 4.2, located on a horizontal surface. Results are rated by the ZNCC of the reference template and the current image warped with the estimated homography, cf. section 4.2. The number of frames where a template could be localized with a ZNCC above a particular threshold increases for all templates and all thresholds when using GREFD compared to regular descriptors.

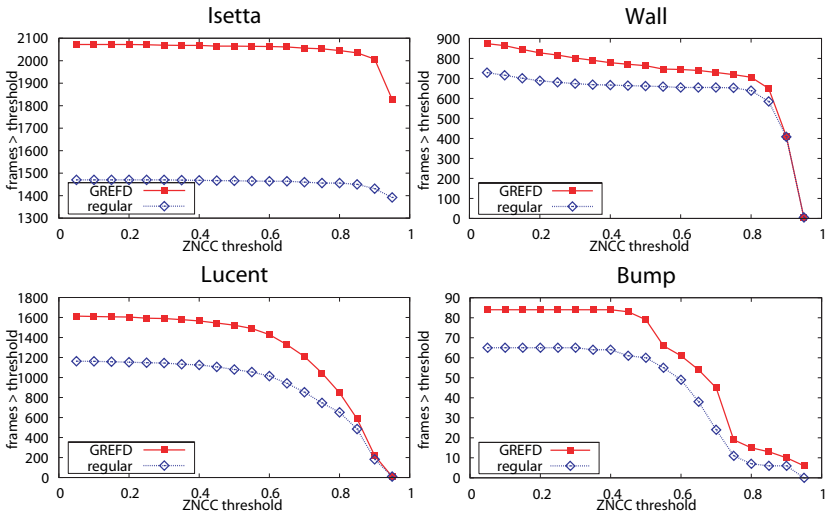


Figure 5.19: Number of frames where feature-based template localization resulted in a ZNCC above a threshold for different thresholds. When using GREFD, all ZNCC thresholds are exceeded more often than with regular feature descriptors.

5.2.5 Evaluation of GREFD-Based Template Localization with Ground Truth Poses

Using synthesized gravity vectors as explained in section 4.3, we evaluate GREFD on a dataset of 48,000 images with ground truth poses. As the purpose of this descriptor is to improve the description of features corresponding to physical points located on horizontal surfaces, we consider the template to be oriented horizontally for this evaluation. Consequently the gravity vector $\mathbf{g}_{synth}$ is computed as the negative z-axis ($-\mathbf{z}_w$) of the world coordinate system, as illustrated in figure 4.3. This vector is then again transformed into the camera coordinate system using the known ground truth pose and artificial degradation is applied resulting in a plausible synthesized gravity vector.

Table 5.3 shows for every sequence the ratio of images for which the template could be correctly localized. When comparing the results using *GREFD* with those using the regular approach, we observe for all templates except *Grass*, that gravity-rectification gives significantly better results in the *Angle* sequences, while it performs comparably or worse for the other sequences. The reason for this is, that the *Angle* sequences contain images taken under steep camera angles while in the other sequences the camera axis is mostly close to perpendicular to the template. This table further prints the results of section 5.1.4.3 using Gravity-Aligned Feature Descriptors as reference.

5.2.6 Gravity-Adaptive Image Rectification

When the camera is orientated such that its optical axis is close to perpendicular to a horizontal tracking template, gravity-rectification of the camera image is not recommended. It does not provide significant improvements and sometimes even degradation for feature description while the image warping introduces additional computational load. Therefore, we propose an adaptive approach that creates a transition between regular feature descriptors and GREFD depending on the gravity angle, i.e. the angle between the camera view axis and the gravity vector. This angle is computed as $\cos^{-1}(g_z)$. There are different options where to fade between the regular approach and GREFD. Examples include computing both kinds of feature descriptors and using weighted subsets of them. We decided to use the earliest, simplest and computationally most efficient way, namely to interpolate at the gravity vector level. An adaptive gravity vector $\mathbf{g}_{adapt}$ is then defined as

$$\mathbf{g}_{adapt}(\mathbf{g},t,s) = \frac{\alpha(\mathbf{g},t,s)\cdot\mathbf{g} + (1-\alpha(\mathbf{g},t,s))\cdot[0,\ 0,\ 1]^\top}{\|\alpha(\mathbf{g},t,s)\cdot\mathbf{g} + (1-\alpha(\mathbf{g},t,s))\cdot[0,\ 0,\ 1]^\top\|_2}. \tag{5.13}$$

Note that the above equation is not defined for $\mathbf{g} = [0,\ 0,\ -1]^\top$ and $\alpha = 0.5$. As this is the case when facing the camera straight to the top, it is not relevant for applications tracking up-facing horizontal surfaces. We chose the Sigmoid function for weighting between the real gravity vector and the z-axis, which corresponds to no image rectification at all. The Sigmoid function is defined as follows.

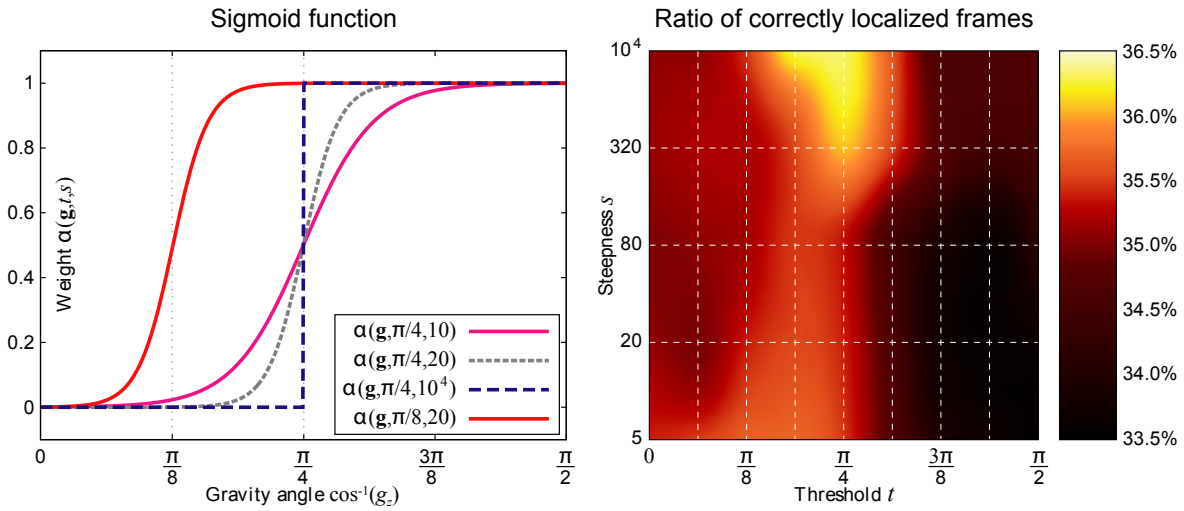


Figure 5.20: We use the Sigmoid function to adaptively fade the gravity vector before image rectification. The two parameters t and s control the threshold and the steepness of the function. The right plot shows the ratio of correctly localized frames in a dataset of 48,000 images for different parameters t and s.

$$\alpha(\mathbf{g},t,s) = 1/\left(1+e^{s(-\cos^{-1}(g_z)+t)}\right) \tag{5.14}$$

As can be seen in figure 5.20, the function can be parametrized with a threshold t and a steepness s resulting in differently shaped transitions.

Figure 5.21 shows for different frames the original image, the image using the proposed adaptive approach and the gravity-rectified image. The adaptive approach provides the same result as the original image for close to perpendicular angles and the same result as gravity-rectification for very steep angles. Depending on the parameters t and s, it results in "partly" rectified camera images when $\alpha =]0,1[$ as the case in t_1. If the steepness s is chosen very high, the transition turns into a rectangular function and the adaptive approach switches from no rectification to full rectification when the gravity angle reaches the threshold t.

We tested all possible combinations of the threshold $t = \frac{k\pi}{16}$, $k \in \{0,1,2,\ldots,8\}$ and $s \in \{5,20,80,320,10000\}$ using the ground truth test explained in section 4.3 to find the best configuration that results in the highest number of frames with a correct template localization. The results over all 40 sequences are shown in the right plot in figure 5.20. The best configuration among the samples has been found for $t = \frac{3\pi}{16}$ and $s = 10,000$. The great steepness s leads to the conclusion that a steep transition between regular feature descriptors and GREFD outperforms smooth transitions. This means, that an implementation of the optimal parameters would simply switch from regular feature descriptors to GREFD once the angle between the camera's optical axis and the gravity reaches $\frac{3\pi}{16}$ radians, which corresponds to 33.75°.

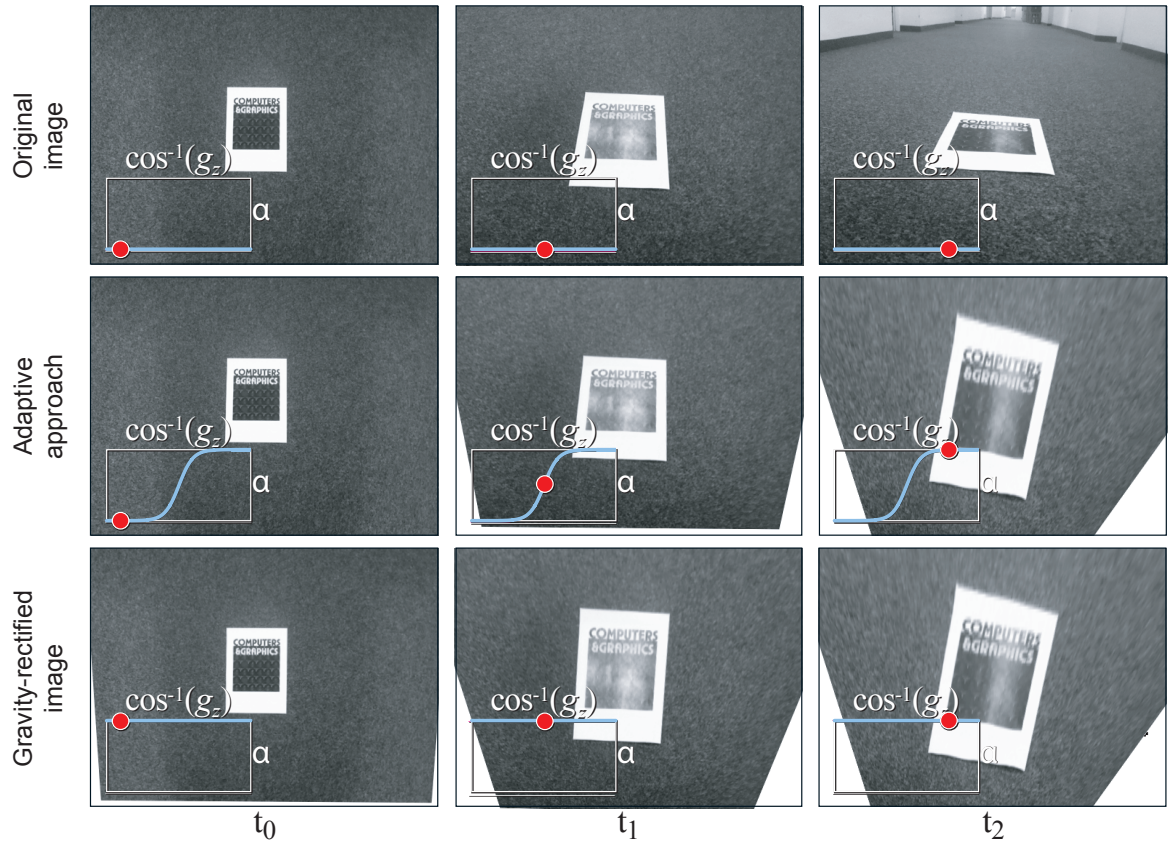


Figure 5.21: Three frames of a sequence imaging a horizontal template. The first row shows the original images while the last row consist of the gravity-rectified versions of these images. The images in the middle row were obtained using the proposed gravity-adaptive image rectification.

The ratio of correctly localized frames using these parameters is shown in table 5.3 in the last column as *aGREFD*, which stands for *adaptive GREFD*. The overall ratio increases by 1.32% compared to GREFD. We can observe that while the ratios of the *Angle* sequences only slightly change, particularly those sequences benefit from the adaptive approach that performed worse using GREFD than with the regular approach. Besides the increased number of correctly localized frames, the adaptive approach most importantly saves computational time. Skipping the image warping for close to perpendicular camera poses enables higher frame rates and therefore faster (re-)initialization which increases the usability of handheld AR applications.

5.2.7 Conclusions

In this section, we showed that gravity-rectification of the camera image before detecting and describing features can improve the precision-recall characteristic of feature descriptors. As a result, Gravity-Rectified Feature Descriptors (GREFD) provide higher success rates for template localization, partic-

Assumed template orientation **Method**	(arbitrary) regular	vertical GAFD	horizontal GREFD	horizontal aGREFD
Isetta (Angle)	90.25%	83.33%	**97.00%**	96.92%
Isetta (Range)	95.42%	**96.92%**	95.25%	95.08%
Isetta (Fast Far)	40.67%	**54.17%**	**41.92%**	40.67%
Isetta (Fast Close)	72.50%	**79.25%**	68.00%	**72.67%**
Isetta (Lighting)	98.92%	**99.33%**	**99.08%**	98.58%
Philadelphia (Angle)	43.00%	30.75%	**62.50%**	61.92%
Philadelphia (Range)	73.42%	**73.75%**	69.75%	**69.92%**
Philadelphia (Fast Far)	13.17%	**14.58%**	12.42%	**13.25%**
Philadelphia (Fast Close)	39.08%	**49.00%**	33.42%	39.00%
Philadelphia (Lighting)	77.08%	**80.25%**	71.42%	73.50%
Wall (Angle)	14.33%	**26.67%**	**20.58%**	19.75%
Wall (Range)	22.42%	**34.17%**	17.17%	20.75%
Wall (Fast Far)	2.50%	**5.67%**	**2.75%**	**2.75%**
Wall (Fast Close)	5.42%	**14.58%**	3.50%	**5.67%**
Wall (Lighting)	28.08%	**52.50%**	22.08%	**28.42%**
Grass (Angle)	1.33%	**1.58%**	0.75%	**1.83%**
Grass (Range)	0.50%	**0.67%**	0.17%	0.42%
Grass (Fast Far)	0.08%	**0.25%**	0.08%	0.08%
Grass (Fast Close)	0.50%	**0.58%**	0.33%	0.25%
Grass (Lighting)	0.75%	0.58%	0.75%	0.58%
Lucent (Angle)	19.58%	**25.08%**	24.83%	**26.08%**
Lucent (Range)	28.42%	**33.83%**	26.83%	**29.00%**
Lucent (Fast Far)	6.50%	**7.25%**	6.67%	6.42%
Lucent (Fast Close)	19.50%	**26.58%**	19.17%	19.33%
Lucent (Lighting)	47.50%	**62.75%**	44.58%	44.83%
Mac Mini (Angle)	10.75%	**24.08%**	12.25%	**13.25%**
Mac Mini (Range)	16.33%	**28.17%**	14.58%	**16.50%**
Mac Mini (Fast Far)	4.67%	**5.58%**	4.00%	4.58%
Mac Mini (Fast Close)	18.33%	**28.42%**	14.83%	17.83%
Mac Mini (Lighting)	22.75%	**41.75%**	20.17%	22.00%
Bump Sign (Angle)	50.75%	49.67%	60.67%	**60.75%**
Bump Sign (Range)	45.42%	**55.33%**	46.00%	**46.17%**
Bump Sign (Fast Far)	13.08%	**23.50%**	12.50%	**13.17%**
Bump Sign (Fast Close)	17.08%	**24.83%**	15.00%	**18.00%**
Bump Sign (Lighting)	49.67%	**65.58%**	52.17%	**53.42%**
Stop (Angle)	53.25%	**70.58%**	82.25%	**82.42%**
Stop (Range)	90.83%	**97.42%**	89.08%	**93.17%**
Stop (Fast Far)	24.75%	**31.50%**	24.33%	23.92%
Stop (Fast Close)	36.33%	**51.08%**	28.25%	34.83%
Stop (Lighting)	88.92%	**92.17%**	84.83%	87.00%
All sequences	34.60%	**41.09%**	35.05%	**36.37%**

Table 5.3: Ratio of correctly localized frames using synthesized gravity measurements for aiding feature description. Bold numbers indicate the best result for a particular template orientation and sequence, if better than the regular approach.

ularly under steep camera angles. Thereby, this method improves invariance to out-of-plane rotations as requested in section 3.2.1. However, gravity-rectification of the camera image introduces an additional image warping to the algorithm. To make sure this potentially expensive step is only performed when actually needed and beneficial, i.e. for steep camera views, we introduced a gravity-adaptive method. Our tests on 48,000 images with ground truth poses verified, that this adaptive approach outperforms gravity-rectification in each frame both in terms of computational efficiency and template localization performance.

While the presented approach is limited to be used with (close to) horizontal surfaces, the following section will introduce an approach to increase invariance of feature descriptors for (non-planar) surfaces in any orientation and without the need for gravity measurements. Since the approach in the following section 5.3 is a training-based offline method, it further does not require any expensive image warping during runtime. For horizontal surfaces, we will also compare its results with GREFD as a baseline method.

5.3 Representative Feature Descriptor Sets

In this section, we present a method which was originally proposed in "Representative Feature Descriptor Sets for Robust Handheld Camera Localization" [Kurz 12b]. This method automatically determines a set of feature descriptors that describes an object such that it can be localized under a variety of viewpoints. Based on a set of synthetically generated views, local image features are detected, described and aggregated in a database. Our proposed method evaluates matches between these database features to eventually find a set of the most representative descriptors from the database. Using this scalable offline process, the localization success rate is significantly increased without adding computational load to the runtime method. Moreover, if camera localization is performed with respect to objects at a known gravity orientation, we propose to create multiple reference descriptor sets for different angles between the camera's principal axis and the gravity vector. This approach is particularly suited for handheld devices with built-in inertial sensors and enables matching against a reference dataset only containing the information relevant for camera poses that are consistent with the measured gravity.

Comprehensive evaluations of the proposed methods using a large quantity of real camera images, a variety of objects, different cameras and different kinds of feature descriptors confirm that our approaches outperform standard feature descriptor-based methods.

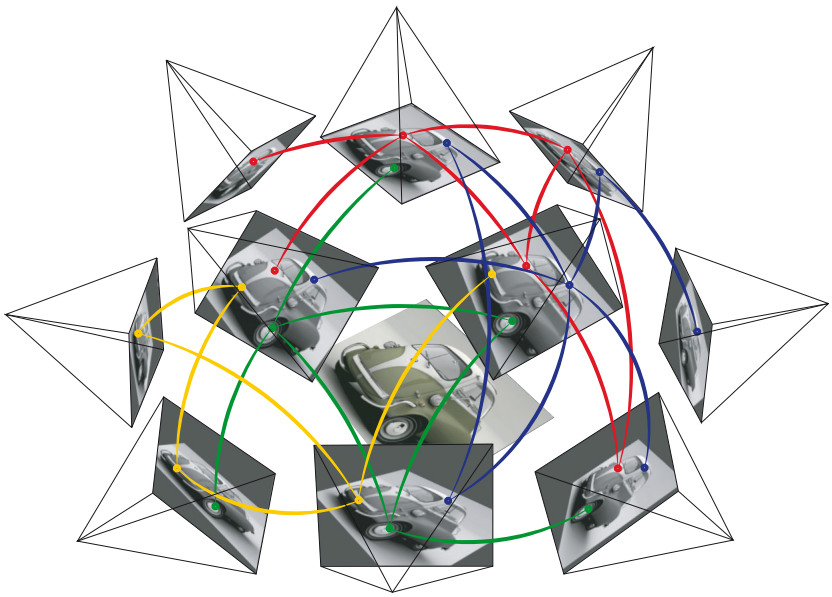


Figure 5.22: Our approach finds representative feature descriptor sets based on matches between synthetic views of an object. This is visualized at the example of a planar object, namely a photograph of an Isetta car.

5.3.1 Introduction

Our proposed method requires a model of the object we aim to localize which enables creating synthetic views of it. For a planar object a fronto-parallel image of the object is sufficient, while for general 3D objects, a textured model of these is needed. While the presented method is universally applicable, for simplicity, we will explain and evaluate the method for planar objects where synthetic views can be generated using simple image warping. In section 5.4 we will use this method with a 3D laser scan of several buildings, but without providing evaluation against a different approach to create feature descriptors for these buildings.

A common approach to localize the camera with respect to an object with a known geometry and visual appearance uses 2D-3D correspondences gained by means of local feature descriptors extracted both from the current camera image and the underlying reference model, as discusses in the previous sections. As the reference descriptors of a planar object are usually based on a fronto-parallel image, localization algorithms work best for this viewpoint and the localization rate decreases with steeper camera angles. In addition to appearance changes as a result of perspective distortions, also aliasing artifacts may lead to a decreased performance for viewpoints different from the reference view. However, in particular in consumer applications, it is important that the tracking works even under steep angles. Therefore, we propose a method that significantly improves camera localization as needed for video-see-through AR.

Based on synthetic views of the real object, we present a general approach for determining a set of representative feature descriptors which can be used with arbitrarily oriented objects. Thereafter, we extend this approach tailored for handheld AR applications where the object has a static and known orientation with respect to gravity and inertial sensors allow for measuring the gravity vector in the camera coordinate system. Our evaluation results attest our approaches lead to clearly increased localization rates for out-of-plane rotations of planar objects without increasing the amount of data or increasing the complexity during runtime.

5.3.2 Proposed Method

Given a reference image $I(u,v)$ of the object, we use image warping to create a set of synthetic views $I(\mathbf{H}_i(u,v))$, as illustrated in figure 5.23. The perspective warpings we use correspond to images taken by virtual cameras that are located on a hemisphere centered around the planar object. To ensure a uniform sampling, we use the vertices of IcoSpheres as camera locations, similarly as in [Hint 11]. Figure 5.25 displays three such IcoSpheres with different resolutions, i.e. number of virtual cameras. The camera's view vector is then defined to look at the object center and the up-vector initially is the normal of the object. Based on these two vectors, an orthonormal coordinate system is defined representing the orientation of the virtual camera. If the intrinsic parameters of the real camera that will be used are known, they are used to eventually compute the homography $\mathbf{H}_i$ corresponding to

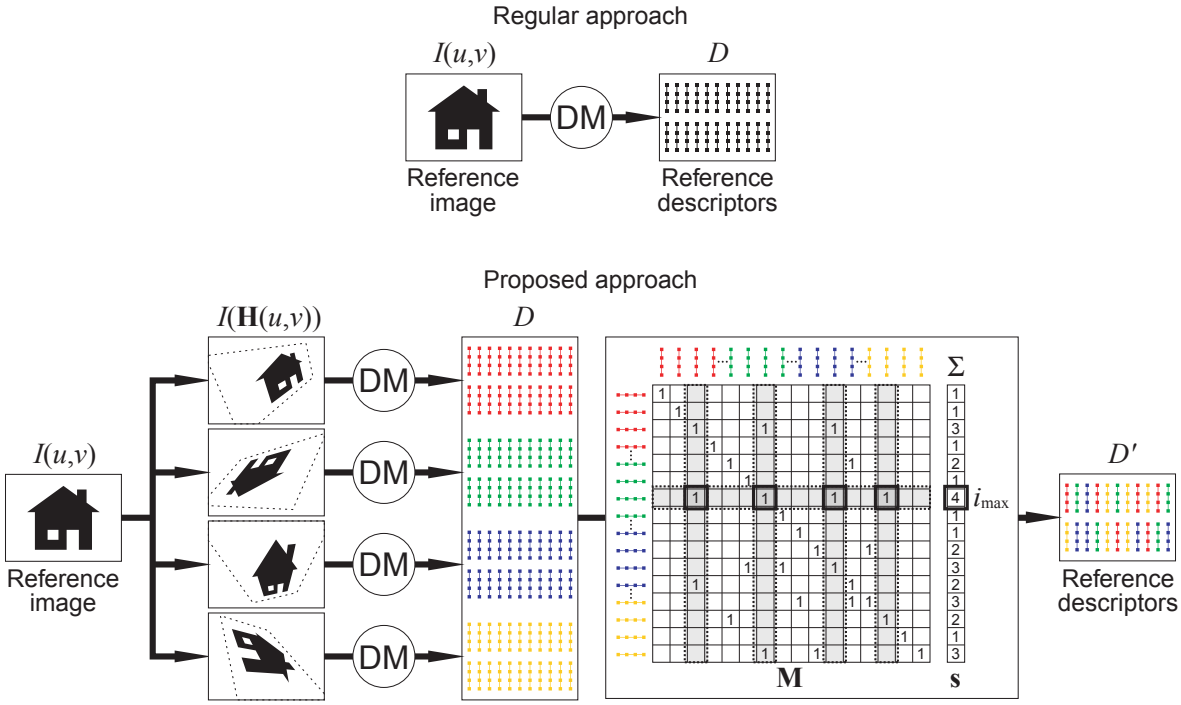


Figure 5.23: Workflow of the proposed off-line process. The regular approach (top) uses one image and creates descriptors for it. Our proposed method (bottom) first creates warpings, then descriptors for every warping, then matches them and based on that selects a subset of the descriptors.

the virtual camera pose. Otherwise, an assumption of the intrinsic parameters is used. The resulting images are shown for a photo of a peach in figure 5.24. For each such image, we detect and describe image features with a description method (DM) and aggregate all descriptors d_i together with the index of the corresponding view v_i in a set of descriptors D and view indices V.

Now, we are looking for a subset $D' \subset D$ of the descriptors in the database that provides a sufficient number of matches among all synthetic views. Therefore, we first match every descriptor $d_i \in D$ against all subsets $D_j = \{d_k | v_k = j\}$ of descriptors from every synthetic view. A match connects a descriptor with its nearest neighbor if the second nearest neighbor is sufficiently far away as in [Lowe 04]. After having matched all descriptors in the database, we discard all wrong matches, i.e. where the feature position warped back to the original reference image differs by more than $\sqrt{1.5}$ pixels. For all remaining (correct) matches, we update the feature positions as the average over all matched features, which results in a sub-pixel precise position that showed to improve localization results.

The matches are then expressed as a binary $(n \times n)$ matrix $\mathbf{M}$, where $n = |D|$ is the overall number of descriptors. The matrix entry $\mathbf{M}_{i,j}$ carries a 1 if the i-th descriptor has been matched with the j-th descriptor and otherwise a 0. Now let the vector $\mathbf{s}$ contain the sum of every row, i.e. $\mathbf{s}_i$ represents the number of matches for d_i.

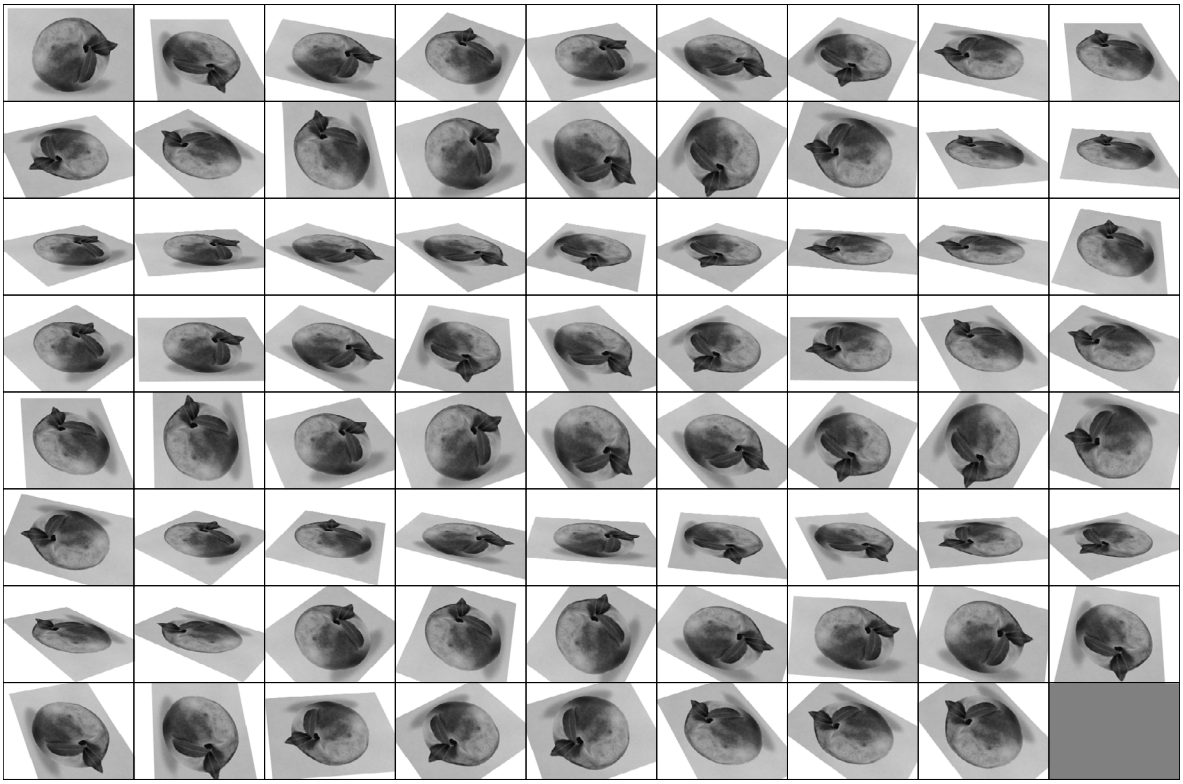


Figure 5.24: The 71 warpings our method creates for a reference image using the set of warpings IcoSphere3

The only parameter our method needs is the desired size of the final feature descriptor set. While the number of descriptors we have in our initially empty set D' is less than this number, we determine the index of the best descriptor (with the most matches) in the database as

$$i_{max} = \underset{i}{\operatorname{argmax}} \left(\mathbf{s}_i \right). \tag{5.15}$$

The i_{max}-th descriptor is then added to the final set of descriptors:

$$D' = D' \cup \{ d_{i_{max}} \}. \tag{5.16}$$

Afterwards, our proposed method sets the i_{max}-th row to zero, such that $d_{i_{max}}$ cannot be picked as best descriptor again. Now that $d_{i_{max}}$ has been added to the set of representative descriptors D', we consider those descriptors that $d_{i_{max}}$ has been matched with, as covered by D'. Therefore, we do not need any additional descriptors in D' which also match against these descriptors. By setting all columns j of $\mathbf{M}$ that have a non-zero element in the i_{max}-th row to zero, we remove these matches from $\mathbf{M}$. Thereby, our method scales well with an increasing number of synthetic views. After updating $\mathbf{s}$, the procedure is repeated until the desired number of descriptors is reached.

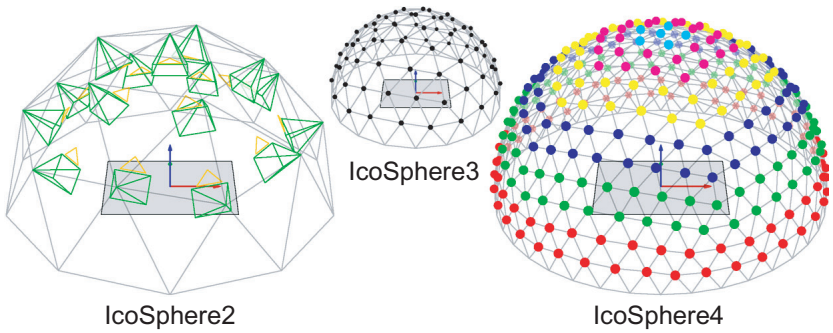


Figure 5.25: The three IcoSpheres whose vertices we use for uniformly sampling 16 (IcoSphere2), 71 (IcoSphere3) or 301 (IcoSphere4) camera positions on a unit hemisphere.

Figure 5.23 illustrates the proposed method for a planar object. The resulting representative feature descriptor set D' is finally used in the same way as D in the regular approach shown in the top of figure 5.23.

It seems natural, that when matching the feature descriptors of a live camera image against the set of reference descriptors, the best matches will involve those reference descriptors that originate from a virtual camera pose similar to the one of the real camera. To empirically confirm this correlation, we divide the camera poses on IcoSphere4 in six different bins, as shown color-coded in figure 5.25, and store with every reference descriptor the corresponding view bin. For a sequence of camera images taken at increasingly steep angles, we plot the histogram of the view bins of those reference features that were successfully matched as inliers in figure 5.26. Here the steepness of the used reference views clearly correlates with the steepness of the camera, which motivates a gravity-aware approach.

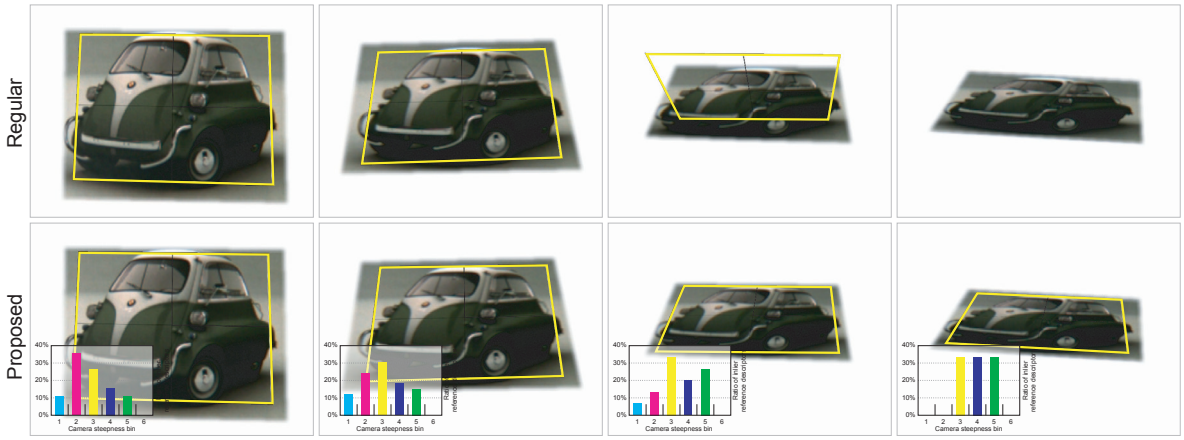


Figure 5.26: Four frames of the evaluation dataset and the localization result of the regular approach (upper row) and our proposed method (lower row). In particular for steep camera angles, our method provides better results than the regular method. The histograms visualize the distribution of the view bins of the reference features that led to the localization, which is clearly correlated with the steepness of the camera.

5.3.3 Gravity-Aware Method

As illustrated in figure 5.1 and discussed earlier, many handheld AR applications are using objects with a known orientation with respect to gravity for camera localization. Most handheld devices are equipped with inertial sensors which provide a measurement of the normalized gravity vector $\mathbf{g} = [g_x, g_y, g_z]^\top$ in the camera coordinate system. Based on this vector, the angle between the camera's principal axis and gravity can be computed as $\gamma_c = \cos^{-1}(g_z)$. As this angle is correlated with the corresponding angle γ_r of those synthetic reference views that provide the most inlier matches, cf. figure 5.26, we aim to narrow the set of reference descriptors to match against based on this angle.

We propose to create multiple representative feature descriptor sets D'_g for different camera orientations with respect to gravity, i.e. for certain ranges of γ_r. During runtime, we then only use the reference descriptor set that corresponds to the current measured camera orientation angle γ_c. Thereby, the same number of reference descriptors to match against can contain much more descriptors representing the object in an orientation similar to the one of the camera. For the camera positions sampled using IcoSphere4, cf. figure 5.25, we create six view bins which are visualized with different colors.

As in the original approach, we create feature descriptors for all virtual cameras. The matching stage is then carried out for every view bin g, by matching all descriptors belonging to the g-th bin against all descriptors from all bins. The resulting matching matrix $\mathbf{M}$ is a $(m \times n)$ matrix where m is the number of descriptors in the g-th view bin and n is the overall number of descriptors in all bins. The algorithm then proceeds as described in section 5.3.2 by iteratively finding the best descriptor, adding it to D', removing the related matches from $\mathbf{M}$, and updating $\mathbf{s}$ until the desired number of descriptors is reached.

For every live camera image, we then compute γ_c based on the measured gravity vector and only match against the reference descriptor set with the average $\overline{\gamma_r}$ closest to γ_c. We found experimentally, that our method provides better results when computing the matches of the descriptors of a view bin with all descriptors from all bins than matching only within each view bin. Thereby, a descriptor is considered good not only if it can be matched with many descriptors within the same bin, i.e. gravity angle range, but also with descriptors from different bins which helps avoiding aliasing problems at the borders between two bins.

In comparison with Gravity-Rectified Feature Descriptors (GREFD), which were described in section 5.2, this approach does not require warping the camera image during runtime, which can be expensive on handheld devices. Instead we create warpings of the reference image in an offline step. More importantly, our proposed gravity-aware method can be applied to surfaces at arbitrary but known orientations with respect to gravity, while GREFD is limited to horizontal surfaces.

5.3.4 Evaluations and Results

In order to evaluate the impact of our proposed methods, we compare the success rate of a template localization algorithm using four different feature description methods. The *regular* version detects and describes both reference and current features in the provided images without any additional processing. The *GREFD* version uses regular feature detection and description for the reference image. However, for every current camera image it uses Gravity-Rectified Feature Descriptors that were introduced in the previous section and rectify the camera image based on gravity measurements before detecting and describing features. Against these baseline versions, we compare our proposed method in two different configurations. The *proposed* version uses regular descriptors for the current camera images but employs the approach explained in section 5.3.2 with the views of IcoSphere4 to determine a set of representative feature descriptors for the reference image. Finally, our *proposed gravity* method creates six of such representative feature descriptor sets for the view bins of IcoSphere4 as discussed in section 5.3.3.

We compare two kinds of feature detectors and descriptors, namely SURF [Bay 08] using the OpenCV implementation, where the descriptor entries have been linearly transformed from the range $[-1.0, 1.0]$ to integers in $[0, 255]$, and our custom 48-dimensional feature descriptor *Mobile48*. It is also based on histograms of image gradients but in contrast to SURF optimized to perform in real-time on mobile devices. We use image pyramids to achieve scale invariance of the feature descriptors. The number of reference descriptors to match against in a single frame is fixed to 250 for all methods and tests.

Despite the feature detection and description, all versions use the same procedure for template localization. For every reference feature descriptor, the nearest neighbor among all current feature descriptors is determined using the Euclidean distance and exhaustive search. If the ratio between the distance to the nearest and the second nearest neighbor is below a threshold of 0.77, the two descriptors are matched.

All matches, which represent 2D-2D correspondences between the reference image and the planar object in the current camera image, are used to determine a homography mapping the current image to the coordinate system of the reference image using PROSAC [Chum 05]. This homography is then refined using all inlier matches. Finally the homography is used in combination with the calibrated intrinsic camera parameters to determine the camera pose with respect to the planar object.

Note that the algorithm does not perform any frame-to-frame tracking, i.e. it does not use any information from the previous frame but re-initializes in every frame. The underlying template localization framework used here is an improved version of the framework used for the results in table 5.3, which explains why the baseline methods provide better results in this evaluation.

5.3.4.1 Benchmark with Ground Truth Poses

The results of the aforementioned template localization method can be judged in different ways, where the most reliable approach is to compare them with ground truth. To this end, we employ the dataset explained in section 4.3 with synthesized gravity vectors, which are needed by the approaches *proposed gravity* and *GREFD*. For every template, there are five image sequences, namely *Angle*, *Range*, *Fast Far*, *Fast Close* and *Lighting*, cf. [Lieb 09]. While the *Angle* sequences contain a good amount of steep camera angles, the other sequences do not and are therefore summarized as *Others*.

Table 5.4 shows for the versions of the algorithm using *Mobile48*, the ratio of camera frames within every sequence, in which the planar object could be correctly localized. As can be seen, our proposed methods outperform the regular approach in all sequences. On average, the localization success rate increases by 21.96% over all sequences and by 33.74% for the steep *Angle* sequences when using the *proposed* method compared to *regular*. Note that these are relative percentages as opposed to the absolute percentages given in table 5.4. For most of the sequences, *proposed gravity* gives even better results, in particular for the *Angle* sequences, where the localization rate increases by nearly 40% compared to *regular*.

However, *GREFD* is clearly superior for the *Angle* sequences containing steep camera views. But as the impact of *GREFD* is very small and sometimes even negative for the *Others* sequences without steep camera poses, the overall performance of both *proposed* and *proposed gravity* is better than that of *GREFD*. In addition to the increased ratio, our proposed methods also increase the localization precision. While the average re-projection error of the four template corners is 4.08 pixels in the *regular* approach and 4.12 pixels when using *GREFD*, it decreases to 3.65 pixels for *proposed* and 3.58 pixels for *proposed gravity*. Note that we did not compare this method with adaptive GREFD (cf. section 5.2.6), which aims at overcoming the negative impact of gravity-rectification for images where the viewing axis is close to perpendicular to the surface.

The results of the same evaluation using SURF are also shown in table 5.4 and verify that in particular for the *Angle* sequences, localization improves significantly (+46.33%) with *proposed gravity*. Over all sequences, *proposed gravity* performs best, followed by *proposed* and *GREFD* showing that the *regular* approach has been clearly outperformed by all other approaches.

The reason why *GREFD* is better than *proposed gravity* under steep angles, is that the problem for *GREFD* is simplified with the rectification. 250 descriptors can be used to match a fronto-parallel view (the reference image) with a nearly fronto-parallel view (the rectified current camera image), while our *proposed gravity* method needs to match a steep view (current image) with one out of unlimited possible views at the same gravity angle, whereof only a couple of views have been sampled in the offline stage, with the same number of descriptors. But both proposed methods are more than four times faster than *GREFD* in our implementation, as they do not require image warping during runtime. Additionally, in contrast to *GREFD*, both our proposed methods can deal with surfaces in arbitrary orientations while *GREFD* is restricted to horizontal surfaces.

Sequence**Method** using Mobile48	regular	GREFD	proposed	proposed gravity
Grass (Angle)	8.17%	**14.58%**	13.75%	14.08%
Grass (Others)	7.35%	8.95%	11.38%	**12.41%**
Wall (Angle)	22.67%	**33.08%**	24.00%	24.67%
Wall (Others)	30.25%	30.31%	33.15%	**35.67%**
Bump (Angle)	35.67%	**46.75%**	44.83%	43.83%
Bump (Others)	28.54%	**32.33%**	31.58%	30.94%
Stop (Angle)	36.83%	**82.58%**	54.25%	63.58%
Stop (Others)	53.35%	53.54%	59.96%	**60.31%**
Isetta (Angle)	62.33%	**84.92%**	81.25%	83.75%
Isetta (Others)	58.33%	57.98%	66.10%	**66.33%**
Philadelphia (Angle)	32.42%	**49.42%**	37.17%	36.83%
Philadelphia (Others)	36.98%	35.47%	45.33%	**45.79%**
Lucent (Angle)	23.33%	**44.58%**	37.67%	39.83%
Lucent (Others)	42.87%	43.63%	**55.52%**	54.67%
Mac Mini (Angle)	27.08%	**39.92%**	39.42%	39.33%
Mac Mini (Others)	30.81%	32.08%	41.46%	**41.56%**
All sequences (Angle)	31.06%	**49.48%**	41.54%	43.24%
All sequences (Others)	36.06%	36.79%	43.05%	**43.46%**
All sequences	35.06%	39.33%	42.76%	**43.42%**
Sequence**Method** using SURF	regular	GREFD	proposed	proposed gravity
All sequences (Angle)	35.98%	**61.98%**	51.51%	52.65%
All sequences (Others)	38.57%	39.23%	**47.31%**	47.30%
All sequences	38.05%	43.78%	48.15%	**48.37%**

Table 5.4: Ratio of correctly localized frames in the localization ground truth dataset explained in section 4.3. The highest ratio in each row is printed in bold.

Figure 5.27: Selected frames with successful localizations of the handheld dataset using the four objects shown on the right.

In order to empirically verify, that *proposed gravity* can deal with any orientation as long as it is known, we repeated the template localization test with *Mobile48* under the assumption that the templates were located on a vertical surface. Therefore, the definition of the gravity vector and the binning of the views on IcoSphere4 had to be adjusted. The results confirm, that in this configuration, where *GREFD* does not work at all, *proposed gravity* gives an overall ratio of 43.41% correct frames, which is much better than the *regular* approach with which the template is only localized correctly in 35.06% of the camera images.

5.3.4.2 Benchmark with Mobile Device Data

As the targeted application of our proposed methods are mobile applications running on handheld devices, we also carried out evaluations with images and inertial sensor readings provided by a mobile device (Apple's iPad2). In these datasets, both the camera images and the gravity measurements have exactly the properties a real application running on a mobile device would use.

However, there is no ground truth information available for these datasets. Instead, we compute the Zero-Mean Normalized Cross Correlation (ZNCC) between the reference image and the current image warped with the homography that led to the determined pose and assume that this similarity corresponds to the accurateness of the pose, as described in 4.2.

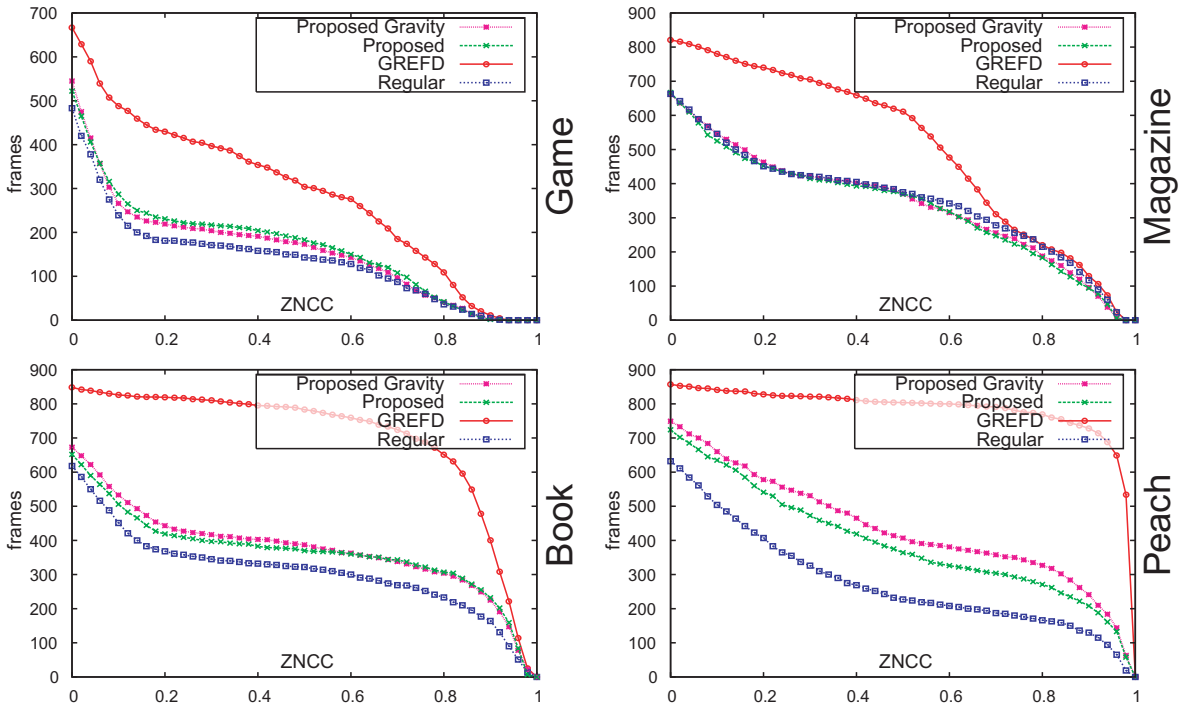


Figure 5.28: Localization results for horizontal templates under steep angles based on the datasets captured with a handheld device.

We evaluate the ZNCCs after localizing the four planar objects shown in figure 5.27 which correspond to real objects that would actually be used in AR applications. The objects were located on a horizontal surface such that they are facing up to provide comparability with GREFD. In contrast to the image sequences used in the previous section where the area around the template is masked out, these sequences do have a background cluttered with office supplies. For each object, we capture 880 images and store them with the corresponding gravity vectors while moving the camera. Some example images with the found pose are shown in figure 5.27.

The plots in figure 5.28 show the number of frames for which the resulting ZNCC was above a particular threshold as a function of this threshold. While for the *Magazine*, the regular approach is slightly better than our proposed methods, the other three objects can be localized with higher ZNCCs more often when using our *Proposed* method to determine representative feature descriptor sets. Incorporating the measured gravity further improves the results for *Book* and *Peach*.

It is obvious, that *GREFD* performs much better than any other method in this evaluation, but it is important to keep in mind that it also is more expensive due to the warping of every camera image. In our implementation, detecting and describing features from a (480×360) pixels image including grayscale conversion and creating an image pyramid takes on average 31 ms on an iPhone 4. The warping needed for GREFD adds another 99 ms making our proposed methods over four times faster than GREFD. Also, GREFD is limited to horizontal surfaces while the proposed methods are not.

5.3.5 Conclusions

We presented a method that automatically determines a set of representative feature descriptors for enabling the localization of real objects under a variety of viewpoints. Our evaluation results attest that our proposal outperforms state-of-the-art methods for feature-based template localization. It does so by improving invariance to out-of-plane rotations, which is one of the key motivations of this work discussed in section 3.2.1.

The range of possible applications of the proposed method is broad as it can be used in any application using any kind of offline-created feature descriptors. Because our method is offline, it does not introduce any computational overhead nor does it increase the amount of data describing an object which is crucial for apps retrieving descriptors via mobile networks.

While all evaluations use planar objects, the same approach can be applied to 3D objects with a given model. For outdoor tracking in urban environments, the range of synthetic views can be drastically reduced assuming that the users are located on the ground and their camera is at an approximately constant height. In the next section, we will use this method to create representative feature descriptor sets based on 3D laser scans of a set of buildings to enable outdoor camera localization.

5.4 Absolute Spatial Context for Feature Descriptors

This section is based on the paper "Absolute Spatial Context-Aware Visual Feature Descriptors for Outdoor Handheld Camera Localization" [Kurz 14]. It presents a framework that enables 6DoF camera localization in outdoor environments by providing visual feature descriptors with an Absolute Spatial Context (ASPAC). These descriptors combine visual information from the image patch around a feature with spatial information, based on a model of the environment and the readings of sensors attached to the camera, such as GPS, accelerometers, and a digital compass. The result is a more distinct description of features in the camera image, which correspond to 3D points in the environment. This is particularly helpful in urban environments containing large amounts of repetitive visual features. Based on the comprehensive test database for outdoor handheld camera localization introduced in section 4.4, we show that using our proposed framework provides both faster matching and better localization results compared to state-of-the-art methods.

5.4.1 Introduction

Figure 5.29: Accurate camera localization outdoors enables outdoor AR applications with precisely registered augmentations.

One of the most important challenges towards the everyday usage of handheld AR is precise and robust camera localization outdoors. Pose estimation, which is based only on information from sensors such as GPS, compass and inertial sensors, is currently being used in AR browsers. The precision of this is controlled by environmental conditions and is usually not enough for pixel-precise registration of overlays in the camera image, as shown in figure 5.29.

Visual localization and tracking is very well suited to provide very accurate registration, and is frequently used for camera tracking in desktop-sized environments, as discusses in the previous sections. While these applications usually assume a static indoor environment, there are different challenges

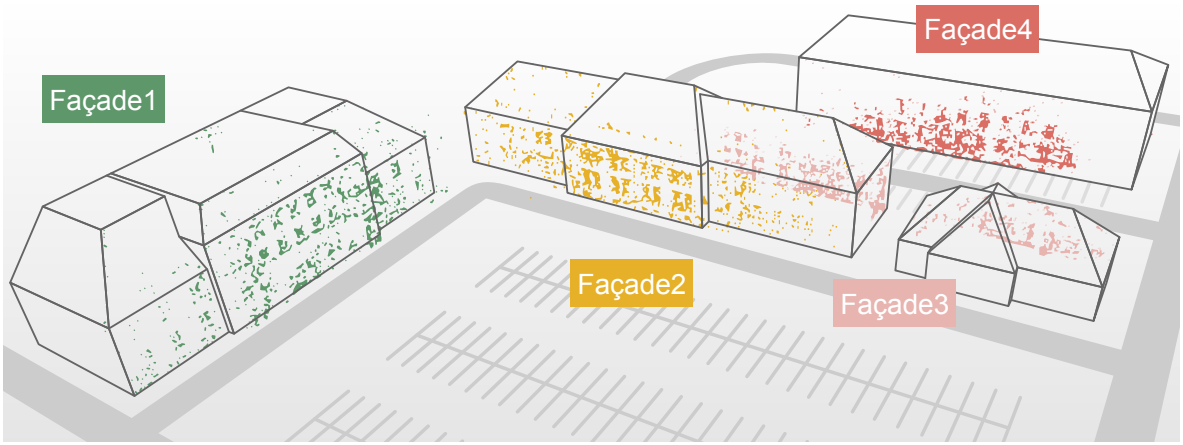


Figure 5.30: Our framework localizes a camera with respect to sparse feature map and exploits a coarse environment model together with sensor readings to aid feature detection, description and matching.

to tackle when going outdoors. Illumination and weather may be subject to change and parts of the environment, such as parked cars and pedestrians, frequently change and might occlude static parts of the environment. Another very important challenge is handling repetitive visual structures which are ubiquitous in urban and man-made environments. This was for example discovered in [Arth 12] and is elaborated in section 3.1. The outdoor dataset presented in section 4.4 covers all the above aspects, for it comprises of real camera images of a real urban environment taken at different points in time. Therefore we will use it to evaluate the method proposed in this section.

5.4.2 Proposed Method

We propose a visual 6DoF localization framework that – in addition to the camera image – employs the auxiliary sensors, which off-the-shelf smartphones are equipped with, to estimate an accurate camera pose. A GPS receiver provides a coarse absolute position, an electronic compass measures the device's heading and the direction of gravity is obtained from inertial sensors. Together with the assumption that the device is approximately 1.6 meters above the ground floor, a coarse 6DoF camera pose can be computed. While this pose usually is not accurate enough for precisely registered visual augmentations in the camera image, we describe how it can be used to support computer vision methods that enable a more accurate camera pose estimation.

Our work is based on a state-of-the-art visual localization and tracking framework, using local image features and 2D-3D point correspondences. We make contributions to advance state-of-the-art in feature detection, feature description, and feature matching for (wide-area) outdoor applications by giving features an Absolute Spatial Context, which will be explained in the following.

5.4.2.1 Required Environment Model

As this work aims to localize a camera in a known environment, we require a model that describes the environment in a way that enables determining correspondences between the model and parts of a camera image. Our method is based upon a sparse representation of the environment comprising of 3D points with associated feature descriptors that will be explained in more detail in 5.4.2.3. Such a kind of model, which we will refer to as *reference feature map*, can be obtained by means of structure from motion methods, e.g. [Arth 12], or from synthetic views of dense environment model as explained in section 5.4.3.1.

We use our custom feature descriptor Mobile48 which performs in real-time on mobile devices. This descriptor uses the direction of gravity to normalize an image patch around a feature before its description, as proposed in section 5.1. The reference feature map describes local parts of the environment in great detail, but does not contain any topological or global information.

Additionally, our method requires a coarse but dense polygonal representation of the environment's surfaces. Such models can, for example, be obtained by extruding floor plans. This representation will be referred to as *reference surface model*, and neither needs to contain any details nor does it need to be accurately registered. As it will be described in the following, it is only used to aid the process of feature detection, description and matching, but its coordinates are never used for pose estimation.

Figure 5.30 shows the reference feature maps of four building façades in different colors and the reference surface models in black. The surface models are stored as a set of 3D triangles and the reference feature maps are stored as sets of 3D points with associated ASPAC-aware feature descriptors. Since both models do not share any 3D points, they are stored separately. All models are required to be in a consistent and geo-referenced coordinate system. We assume such data can be made available for the majority of cities soon.

5.4.2.2 Environment Model-Guided Feature Detection

The detection of image features is commonly used to speed up finding correspondences in images containing the same object or scene from different views. Instead of comparing patches around every pixel, comparison and matching is only performed for salient image features. It is crucial to the whole process of describing and matching features for camera localization, that the detected features are well distributed and that many of them actually correspond to the object or environment our reference model describes.

In particular in outdoor environments, large parts of the camera image often contain objects which are not part of the model. Examples include clouds in the sky, the floor, trees, cars and pedestrians potentially occluding parts of the model, see figure 5.31 left. Therefore, we developed a method to make the feature detection process focus on the parts of the image that most likely correspond to something meaningful for localization.

Figure 5.31: Comparison of regular feature detection on the entire image (left) and the proposed environment model-guided approach (right).

As described above, we use GPS, compass and inertial sensors to compute a coarse 6DoF pose of the camera in a global coordinate system, which we will refer to as *sensor pose*. To account for inaccurate GPS, we make sure the pose is not located inside the surface model and not facing surfaces too close to the camera. To this end, we push the pose backwards along the principal axis until the closest intersection of this axis and the surface model is at least 15 meters away from the camera.

Based on this pose and the known intrinsic camera parameters, we project the reference surface model into the camera image. The resulting mask is used to only extract features in the camera image where parts of the model project into the image. Additionally, we propose to not extract any features in the lower hemisphere centered around the camera, because the parts of the model located below the horizon line are often occluded by pedestrians or cars.

Figure 5.31 compares the distribution of extracted features from two images using a regular approach (left) with our proposed method (right). In all cases, we use the FAST corner detector [Rost 06] and find a threshold that results in 300 corner features. It is apparent that our proposed method leads to a significantly higher ratio of detected features, which correspond to parts of the environment model,

than in the regular approach. As a result, the robustness against background clutter and partial occlusions of the environment is increased. In section 5.4.3.2 we will show that this also results in significantly increased localization success rates.

5.4.2.3 Absolute Spatial Context-Aware Visual Feature Description

Given a coarse reference surface model and a coarse sensor pose of the camera, we are not only able to determine which pixels of the camera image most likely contain parts of the model, but we also retrieve the coarse coordinates of the 3D point $\mathbf{P}(u,v)$ corresponding to those pixels. The position can for example be obtained by ray casting or by rendering the surface model into a position map. Additionally, the sensor pose provides every feature with an absolute orientation. In the following, we will describe how this information can be used to improve feature description by giving features in the camera image an Absolute Spatial Context (ASPAC).

As stated in section 3.1, an important challenge for visual outdoor localization in urban environments is to deal with repetitive visual features, such as the four corners of a window and multiple windows on a façade side by side and on top of each other that look exactly the same. Additionally, man-made environments tend to contain visually similar features at different physical scales. It is crucial for any visual camera localization method to distinguish these repetitive features to be able to determine an accurate camera pose.

In this work, we use the term *Absolute Spatial Context* to describe the absolute scale, the absolute position, and the absolute orientation of a feature. As opposed to the common definition of a feature's scale, position and orientation, which are defined in the (2D) coordinate system of the camera image, the Absolute Spatial Context is defined in a (3D) global world coordinate system.

Awareness of Absolute Scale makes it possible to distinguish features with similar visual appearance at different physical scales. Most state-of-the-art camera localization and tracking methods based on local image features are scale-invariant. A common way to make feature detection and description invariant to scale, which is also used in our approach, is to use image pyramids that represent a camera image at different scales. This makes it possible to detect and describe visual features of an object in a similar way no matter if it is 1 meter away from the camera or 5 meters away.

As this scale invariance happens in a projected space, i.e. the camera image, it is impossible to distinguish scale resulting from the distance of an object to the camera from the actual physical scale of an object. Invariance to scale resulting from the distance of the camera to an object is clearly desirable in many applications, and was the original motivation for scale-invariance.

However, in the presence of similar features at different physical scales, invariance to scale makes them indistinguishable which may result in mismatches and therefore is undesirable.

In the following, we will use the term *feature scale* as a scalar value describing the width and height of the squarish support region of the feature's descriptor. Given the coarse sensor pose and the coarse 3D coordinates $\mathbf{P}(u,v)$ of a visual feature located in the camera image at pixel (u,v), the distance from the optical center of the camera to the feature point $d(u,v)$ can be easily computed.

Based on this distance, the intrinsic camera parameters, and the scale of a feature in pixels s_{pix} as described above, we propose to compute an approximation of its absolute physical scale $s(u,v)$ as

$$s(u,v) = s_{\text{pix}}(u,v)\frac{d(u,v)}{f}, \tag{5.17}$$

where f is the camera's focal length in pixels. This absolute physical scale is computed and stored for every feature.

Awareness of Absolute Position can help distinguishing between repetitive features that are located at different positions, such as similar windows on a building façade. To this end, we simply store the approximate 3D position $\mathbf{P}(u,v)$ of a feature computed from the sensor pose and the reference surface model. This position is, of course, inaccurate, but we will discuss in 5.4.2.4 how it can aid and speed up the process of feature matching.

Awareness of Absolute Orientation makes similar features at different absolute orientations distinguishable, as described in section 5.1. We use Gravity-Aligned Feature Descriptors (GAFD) that take the measured direction of gravity projected into the camera image as feature orientation. The visual descriptor, which is based on a histogram of gradient orientations in the image patch, is denoted as $\mathbf{v}(u,v)$.

Additionally, we compute and store the dominant gradient direction o_{gradient} in a patch around the feature relative to the orientation of the gravity o_{gravity} as an additional part of the Absolute Spatial Context. This is in fact the same approach as pursued in section 5.1 under the name *GAFD fast*.

$$o(u,v) = \left|o_{\text{gradient}} - o_{\text{gravity}}\right|_{\text{angle}} \tag{5.18}$$

$$|\alpha|_{\text{angle}} = \begin{cases} \alpha + 2\pi, & \text{if } \alpha \leq -\pi \\ \alpha - 2\pi, & \text{if } \alpha \geq \pi \\ \alpha, & \text{else.} \end{cases} \tag{5.19}$$

Figure 5.32 plots an exemplary distribution of absolute orientations of the features located on a building façade. We clearly observe peaks at all multiples of $\pi/2$, i.e. 90 degrees, which are very common in man-made environments.

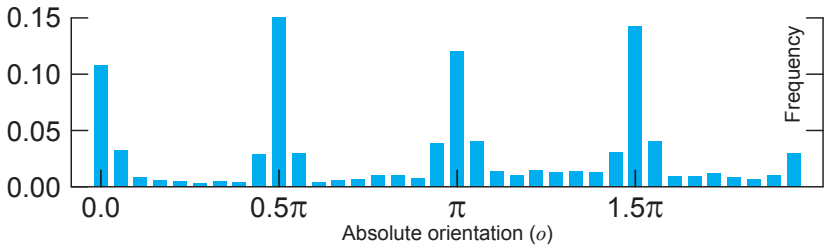


Figure 5.32: Distribution of the absolute orientation of features on a building façade comprising of mainly horizontal and vertical structures.

All the properties of a feature described above remain constant for varying camera positions and orientations. Therefore, they can be used to add distinctiveness to visually similar features, as they are very frequent in man-made environments. Additionally, they can be used to significantly speed up feature matching, which will be explained in the following.

5.4.2.4 Matching Absolute Spatial Context-Aware Features

After detecting and describing features from a camera image, the matching stage is responsible for determining correspondences between these camera features and the reference feature map. We do this by finding for every camera feature the reference feature with the lowest dissimilarity using exhaustive search. Based on the resulting 2D-3D correspondences, the 6DoF pose of the camera can finally be determined.

We propose to use the Absolute Spatial Context of visual features to speed up the matching process by precluding potential matches where the context is not consistent. Thereby, for a majority of combinations of camera and reference features, the expensive step of computing the distance of their visual descriptors can be skipped. This not only results in faster matching, but also provides more correct matches, because the Absolute Spatial Context prevents similar looking features that differ significantly in their global scale, position, or orientation, to be matched.

As described in the previous section, our feature description $\mathbf{d}$ is composed of

- the absolute position $\mathbf{P}$,
- the absolute scale s,
- the absolute dominant gradient orientation with respect to gravity o, and
- and a gravity-aligned visual feature descriptor $\mathbf{v}$.

To compute the dissimilarity of two features, our method computes intermediate distances δ_i followed by a check if these intermediate distances are below given thresholds. If this condition is fulfilled, the next intermediate distance is computed. Otherwise, the dissimilarity of the two features is set to infinity without any further computations.

The dissimilarity is defined as

$$\left\|\mathbf{d}_i - \mathbf{d}_j\right\| = \Delta_1 \tag{5.20}$$

where $\Delta_1, \Delta_2, \Delta_3$, and Δ_4 are defined as

$$\Delta_k = \begin{cases} \infty, & \text{if } \delta_k \geq t_k \\ \Delta_{k+1}, & \text{if } \delta_k < t_k. \end{cases} \quad \text{and} \tag{5.21}$$

$$\Delta_5 = \delta_5. \tag{5.22}$$

As intermediate distances we use the distance on the x-y plane

$$\delta_1(\mathbf{d}_i, \mathbf{d}_j) = \left| [1,1,0]^\top (\mathbf{P}_i - \mathbf{P}_j) \right|, \tag{5.23}$$

the distance along the z axis (i.e. vertical)

$$\delta_2(\mathbf{d}_i, \mathbf{d}_j) = \left| [0,0,1]^\top (\mathbf{P}_i - \mathbf{P}_j) \right|, \tag{5.24}$$

the ratio of absolute scale

$$\delta_3(\mathbf{d}_i, \mathbf{d}_j) = \max(s_i, s_j) / \min(s_i, s_j), \tag{5.25}$$

the difference in absolute orientation

$$\delta_4(\mathbf{d}_i, \mathbf{d}_j) = \left| \left| (o_i - o_j) \right|_{\text{angle}} \right|, \tag{5.26}$$

and the visual descriptor distance

$$\delta_5(\mathbf{d}_i, \mathbf{d}_j) = \left\| (\mathbf{v}_i - \mathbf{v}_j) \right\|. \tag{5.27}$$

Note, that we treat the spatial distance along the z-axis, i.e. the altitude, differently than the distance along the other two axes. This is based on the assumption that the altitude is the most reliable part of the determined 3D position of a camera feature, because it is less heavily affected by an inaccurate compass heading or GPS position, as is also shown in figure 5.33.

The benefit of the proposed method to project camera features into the environment over projecting the reference feature map into the coordinate system of the camera, is twofold. Firstly, depth is preserved and avoids matching camera features against reference features that are occluded or at the

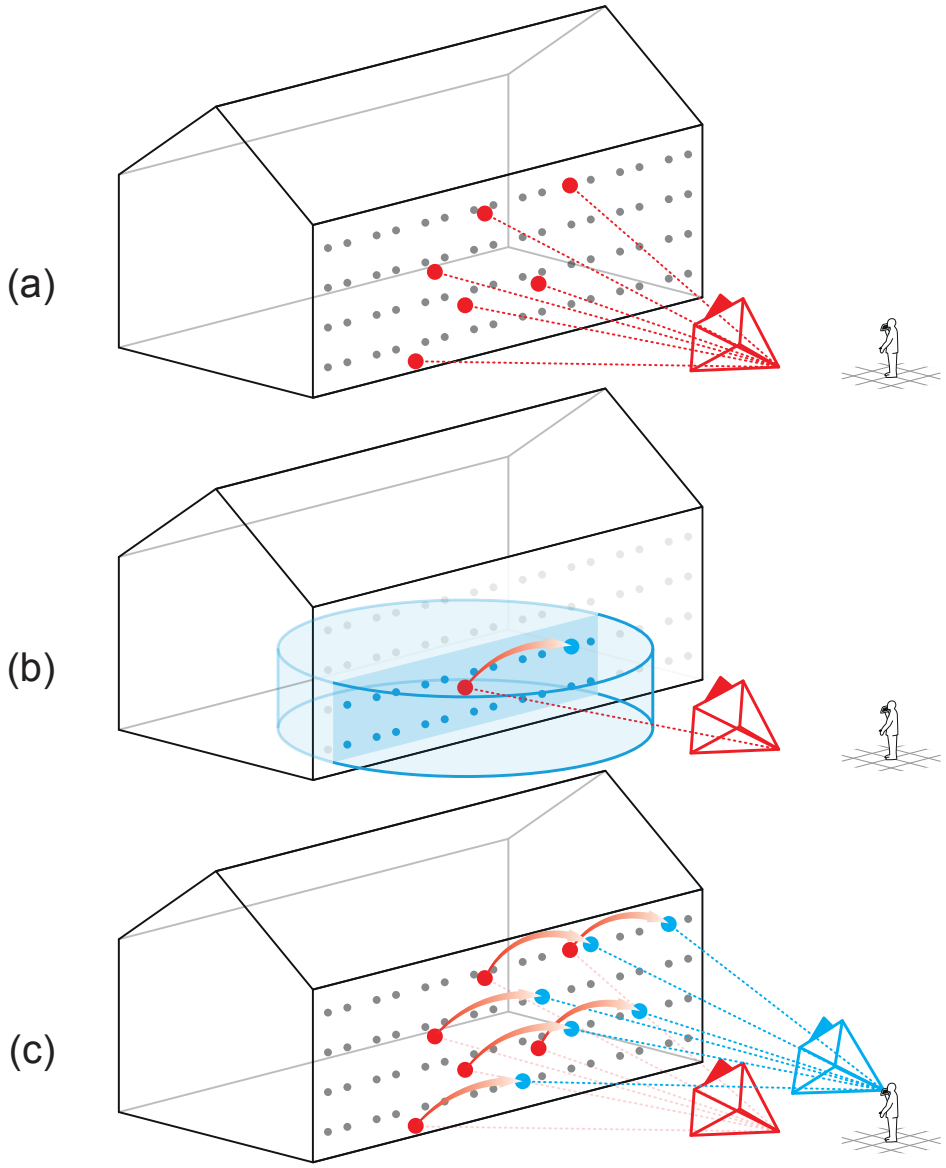


Figure 5.33: Determining the coarse absolute position of camera features (as part of their Absolute Spatial Context) based on the sensor pose (a) enables constraining feature matching to reference features that are located within a cylinder centered at the coarse position (b). Finally, the accurate camera pose can be determined based on a set of correct matches between 2D camera features and 3D reference features (c).

backside of a building. Secondly, the transformation into a different coordinate system and the computation of feature properties in this coordinate system, which is performed in every frame during localization, is in our case only done for hundreds of camera features, instead of tens of thousands of reference features.

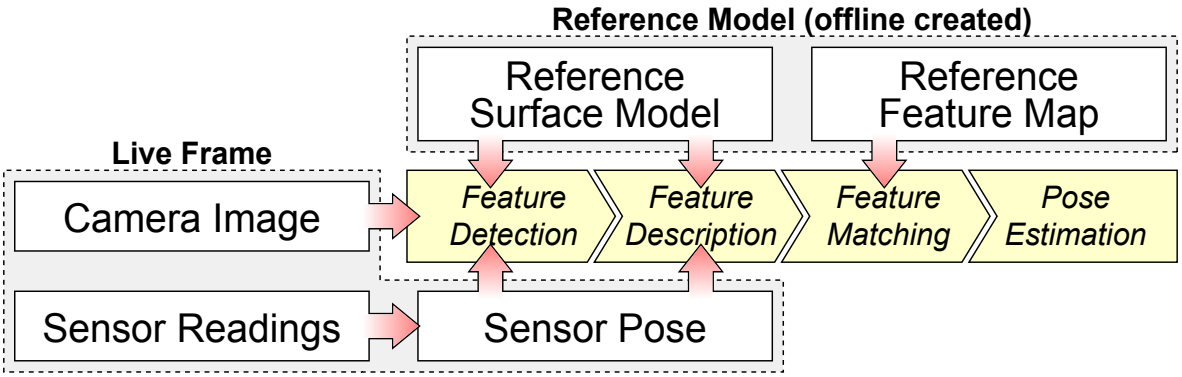


Figure 5.34: Flowchart of the proposed camera localization framework for outdoor environments.

5.4.2.5 6DoF Localization Framework

A flow diagram of our proposed framework for 6DoF camera localization on mobile devices is shown in figure 5.34. It combines the above steps to establish correspondences between features in the camera image and the reference model with a pose estimation functionality. Every live frame consists of a camera image and a set of sensor readings measured at a time close to when the image was taken. Based on these sensor readings, we first compute a coarse sensor pose, which together with the reference surface model is then used in the feature detection stage, as described in section 5.4.2.2.

Afterwards, the features are described as specified in section 5.4.2.3, which again requires the sensor pose and the surface model to determine the Absolute Spatial Context (ASPAC) of the features. Eventually the features of a live camera image are matched against those of the reference feature map, according to the method explained in section 5.4.2.4. The resulting correspondences serve as a basis for the pose estimation step, which performs PROSAC [Chum 05] followed by pose optimization based on all inlier matches.

In an outdoor handheld AR application, this localization step would be followed by frame-to-frame tracking. Our work, however, focuses on camera localization (i.e. initialization) only.

5.4.3 Evaluation and Results

In the following, we evaluate our proposed localization framework, described in section 5.4.2, using the outdoor 6DoF ground truth dataset, which was explained in section 4.4.

5.4.3.1 Ground Truth Localization Test

In order to make use of the detailed ground truth model of the environment, we first need to convert it into the model representation required by our method as described in section 5.4.2.1. We define the four different building façades, shown color-coded in figure 5.30, as objects we are interested in for localization while the rest of the environment – mainly consisting of the floor, cars and trees – is not relevant. For each of these objects, we create a localization reference model as follows.

By rendering the ground truth model from different virtual viewpoints, we gain a set of synthetic photo realistic views of the environment, where for every pixel the corresponding 3D position is known. We then detect features with known 3D coordinates from these views and describe them as elaborated in 5.4.2.3. Note, that instead of using a coarse pose computed from GPS, compass and inertial sensors, we use the precisely known pose of the virtual camera used to render the view to provide the Absolute Spatial Context. Finally, we determine out of the descriptors from all the views, a representative feature descriptor set comprising of 2,000 features per object, as explained in section 5.3. This set of descriptors and features is then used as a reference feature map. The reference surface models have been manually created for this test and are shown in figure 5.30. We run all tests offline on a PC, but use the same localization framework that runs in real-time on mobile devices.

As we are interested in localization (or initialization) only, and not tracking, we treat every single frame individually. The image and the sensor readings (GPS position, compass heading, and gravity vector) are read from files and provided to the system. We then perform the whole localization pipeline (as explained in 5.4.2.5) on this data as if it was live data. Eventually, the framework either returns a determined camera pose or replies that it did not succeed to localize the camera.

In video-see-through Augmented Reality applications, it is most important that the visualization (rendered with the estimated pose) appears correctly registered with the camera image. Therefore, we use the mean re-projection error of a set of k 3D vertices located on the reference model as error measure. The 2D position of a 3D vertex $\mathbf{V}$ projected into the camera image using the estimated pose $\mathbf{v}_{est} = [u_{est}, v_{est}, 1]^\top$ and using the ground truth pose $\mathbf{v}_{gt} = [u_{gt}, v_{gt}, 1]^\top$ can be computed from

$$[w_{est}u_{est}, w_{est}v_{est}, w_{est}]^\top = \mathbf{K}[\mathbf{R}_{est}\mathbf{t}_{est}] \begin{bmatrix} \mathbf{V} \\ 1 \end{bmatrix} \quad \text{and} \tag{5.28}$$

$$[w_{gt}u_{gt}, w_{gt}v_{gt}, w_{gt}]^\top = \mathbf{K}[\mathbf{R}_{gt}\mathbf{t}_{gt}] \begin{bmatrix} \mathbf{V} \\ 1 \end{bmatrix}. \tag{5.29}$$

Thereby $\mathbf{K}$ denotes the camera intrinsic matrix, and $[\mathbf{R}_{est}\mathbf{t}_{est}]$ and $[\mathbf{R}_{gt}\mathbf{t}_{gt}]$ are the estimated pose and ground truth pose respectively. Based on the projected positions ($\mathbf{v}_{est,i}$ and $\mathbf{v}_{gt,i}$) of all vertices $\mathbf{V}_i$, the mean re-projection error e is finally computed as

$$e = \frac{1}{k} \sum_{i=1}^{k} \|\mathbf{v}_{est,i} - \mathbf{v}_{gt,i}\|. \tag{5.30}$$

In the following evaluations, we require the mean re-projection error to be less than a threshold of 4 pixels for the pose to be considered correct.

We run the 6DoF camera localization framework described above on all frames of the outdoor ground truth dataset explained in section 4.4 in three different configurations to evaluate our proposed method against a naïve baseline method as well as an approach similar to what has been recently proposed in the literature. The following modes are used for evaluation:

Naïve A naïve approach, where feature detection is performed on the entire image, feature description uses GAFD, and the matching only compares the visual descriptors.

Orientation Similar to the naïve approach but with orientation-aware feature matching that only compares features with a similar heading analog to what is proposed in [Arth 12].

Proposed Our proposed method, where feature detection, feature description, and feature matching make use of the ASPAC provided by the sensor values.

To evaluate how well the individual approaches scale with an increasing reference model, we evaluate all sequences in all configurations with two different reference models:

Only Only uses the reference model of the building façade, which is imaged in the current sequence.

All All four reference models shown in figure 5.30 are combined to a large reference model which is used for localization in all sequences.

We chose the following thresholds in our evaluation:

$t_1 = 10{,}000$ mm – Spatial distance on the x-y plane.

$t_2 = 2{,}000$ mm – Spatial distance along the z axis.

$t_3 = 1.3$ – Ratio of absolute scale.

$t_4 = 120^\circ$ – Difference in absolute orientation.

The reason for t_4 being large, is that the scene mainly consist of windows, and the corners of these windows, which provide a majority of features, usually have at least two orthogonal dominant gradient directions in their neighborhood making the absolute orientation an unreliable parameter in this environment. The vertical distance threshold t_2 was chosen as two meters, to ensure discrimination between the windows of different building stories, which are usually about 3 meters high.

In the *Orientation* approach, we use a threshold of $\pm 30^\circ$ as in the original paper.

Sequences\Method	Naïve	Orientation	Proposed
Façade1 (Only)	30.87%	22.91%	50.03%
Façade4 (Only)	4.98%	3.66%	9.00%
Total (Only)	25.54%	18.95%	**41.58%**
Façade1 (All)	16.20%	20.99%	49.98%
Façade4 (All)	2.80%	3.01%	9.00%
Total (All)	13.44%	17.29%	**41.54%**

Table 5.5: Ratio of correctly localized frames in the outdoor ground truth dataset.

5.4.3.2 Localization Test Results

First of all, we evaluate for all configurations the ratio of the frames in our ground truth dataset for which the localization framework determines a correct pose. The results are given in table 5.5, and show that the *Orientation* approach performs better than the *Naïve* approach on the large reference model (All). When dealing with only one façade (Only), the *Naïve* provides better results than *Orientation*. In this case, the *Orientation* approach seems to preclude correct matches due to inaccurate compass heading values.

Our *Proposed* method clearly outperforms all other methods in terms of correctly localized frames. It also is apparent that our *Proposed* method scales very well with an increasing reference model. Scaling the number of reference features by a factor of four (Only $\rightarrow$ All) has a minimal effect, while the ratio of correctly localized frames drops significantly for the *Naïve* approach.

The absolute numbers given above might appear low compared to the results of other papers (e.g. [Arth 12][Vent 12]). It is important to keep in mind that the dataset we use is realistic, and therefore, particularly hard compared to tests in the literature. All reference feature maps are based on the panoramic images from the laser scanner while the test sequences were taken with a mobile phone at different days and weather conditions. Additionally, the scene – particularly façade4 – contains a significant percentage of repetitive visual features, which are mainly windows that additionally reflect the sky, resulting in frequent changes in their appearance. Another challenge is that the majority of our sequences contain cars partially occluding the building façades.

Our proposed method deals well with repetitive visual features, but still has problems with significant changes in illumination. For some of the sequences in the dataset, not a single frame was localized correctly with any method simply because the illumination is too different from that in the reference model. Here, further research on algorithms to compute the visual descriptor in a fashion invariant to illumination is needed.

Distance\Measurement	Computed δ_5	Correct
(a) $\Delta_1 \leftarrow \delta_5$ (Naïve)	100.00%	30.87%
(b) $\Delta_1 \leftarrow \delta_5$ + EMGFD	100.00%	43.56%
(c) $\Delta_2 \leftarrow \delta_5$	43.14%	46.92%
(d) $\Delta_3 \leftarrow \delta_5$	12.89%	47.91%
(e) $\Delta_4 \leftarrow \delta_5$	2.86%	**50.39**%
(f) $\Delta_5 \leftarrow \delta_5$ (Proposed)	**2.16**%	50.03%

Table 5.6: Impact of the individual proposed steps and intermediate distances on localization cost and quality in *Façade1* (Only).

5.4.3.3 Impact of the Individual Steps

To evaluate the impact of the steps involved in our proposed method and the intermediate distances computed in the matching stage, we repeated the experiment above in more different configurations. Beginning from the *Naïve* approach, every row in table 5.6 adds one more of the steps that we proposed before finally computing the visual descriptor distance. Starting from environment model-guided feature detection (denoted by *EMGFD*), we added the constraint on the distance on the x-y plane, the distance along the z axis, the ratio of absolute scale, and the difference in absolute orientation. We observe that the first four steps of our proposed method (b,c,d,e) result in a continuously increased ratio of correctly determined poses, while the number of expensive comparisons of visual feature descriptors (δ_5) needed decreases monotonically. The configuration (e) localizes over 1.6 times as many frames correctly as the *Naïve* approach and requires less than 3% of the visual descriptor comparisons.

Adding the constraint on the absolute feature orientation (f), i.e. our *Proposed* method, results even less visual descriptors being compared (factor 0.76) but also slightly decreases the ratio of correctly localized frames (factor 0.99). Therefore, depending on the application, it can make sense to omit this step because the absolute feature orientation already contributed to the distinctiveness of the (gravity-aligned) visual descriptors.

5.4.4 Conclusions

This section presented a framework for visual camera localization that utilizes the sensors modern mobile phones are equipped with to provide local visual feature descriptors with an Absolute Spatial Context (ASPAC). This novel feature description method overcomes visual repetitiveness in urban environments (in large-scale as well as building-scale), which is one of the most pressing problems for visual camera localization, as discussed in section 3.1. Using a comprehensive outdoor ground truth dataset, we showed that our proposed method clearly outperforms a naïve approach using only the direction of the gravity and visual information, and an approach similar to a recently published work [Arth 12] that additionally uses the heading orientation to constrain feature matching.

Our proposed method to detect, describe and match features shows that the auxiliary sensors of mobile devices can help to not only get better localization results, but to also speed up matching by precluding the comparison of visual descriptors of features with a largely different Absolute Spatial Context. Additionally, the proposed scheme can be very well applied to feature matching in hardware, which will make matching against large databases of reference feature maps virtually free of cost in the future.

6 Sensor-Aided Handheld Augmented Reality

Different approaches to aid visual feature descriptors based on the readings of sensors attached to the camera were explained and evaluated in the previous chapter. While feature descriptors play an important role in most handheld Augmented Reality applications, there are more parts in the AR pipeline which can benefit from such sensor readings being available.

This chapter is based on parts of the paper "Gravity-Aware Handheld Augmented Reality" [Kurz 11a]. A commonly used Augmented Reality pipeline for applications dealing with a planar object and using dense frame-to-frame tracking is shown in figure 6.1. Initially, the camera pose needs to be determined with no prior knowledge from a previous frame (A). This is usually done based on point features and descriptors as described in the previous chapter. Once this initialization succeeded, a frame-to-frame tracking method can be used to continuously determine the camera pose in an incremental fashion given the pose in the previous frame. This is commonly done with a dense template tracking approach (B) and we will describe in section section 6.1 how gravity measurements can aid this process if the planar object is oriented horizontally.

If frame-to-frame tracking fails, initialization is again performed (A). But if it is successful, the determined pose is finally used to render virtual 3D objects spatially registered overlaid onto the camera image (C), which ultimately is the goal of video-see-through Augmented Reality. In section 6.3, we show how the rendering of virtual 3D objects can benefit from the measured direction of gravity in handheld Augmented Reality applications. After rendering, the pipeline proceeds with frame-to-frame tracking of the next frame (B).

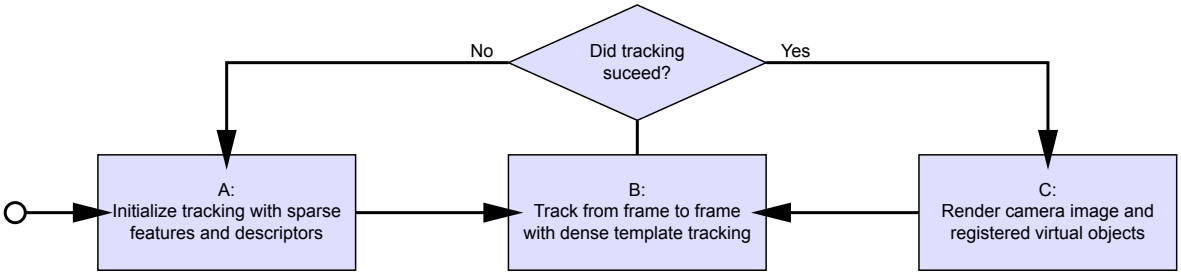


Figure 6.1: High level chart of a common Augmented Reality application flow.

6.1 Inertial Sensor-Aided Template Tracking

For arbitrary camera poses, the transformation an object undergoes when imaging it is a 3 DoF rotation, followed by a 3 DoF translation and eventually a projection onto the image plane. For planar objects, this perspective transformation can be fully described with a homography, which is a (3×3) matrix. This section deals with algorithms that can be used to track the position of a planar object in an image by estimating a homography which registers the current camera image with a template image of the planar object. When the template (i.e. planar object) is known to be located on a horizontal surface (e.g. a magazine on a table, cf. figure 5.1), we propose to rectify the camera image based on the measured gravity vector prior to aligning the camera image with the template. The principle of gravity-rectification of camera images and the underlying equations are discussed in section 5.2.2.

Figure 6.4 a depicts a camera image and its gravity-rectified version in figure 6.4 b. As can be seen, after rectification, the transformation between the template and the rectified image can be described with a similarity transform supporting 3 DoF translation and in-plane rotation only. Any image registration algorithm for similarity transforms can be applied to find this transformation. Having the similarity transformation, we propose to multiply it with $\mathbf{G}^{-1}$, which is the inverse of the transformation $\mathbf{G}$ used to rectify the camera image. This results in the correct perspective transformation for registering the current image with the template, cf. figure 6.4 c. The motivation behind this procedure is, that tracking using similarity transforms is computationally less expensive and more stable than perspective tracking. In section 6.2 we will also show that it requires significantly less memory for particular algorithms, which is beneficial particularly on mobile devices.

6.1.1 Improving Gradient Descent Image Registration

A widely used class of image registration methods is based on gradient descent algorithms. These image registration methods are based on non-linear optimization algorithms that iteratively find a local minimum of a cost function which describes the difference between the template and the current image. These algorithms iteratively update the cost function and solve a system of equations to find the optimal parameters of some transformation that is applied to one of the two images.

Note that the implementation of a gradient descent image registration method supporting only similarity transformations is very inefficient because it does not enable linear parameterization. Instead, we use methods that support affine transformations $\mathbf{H}_{affine}$, which additionally include shears and non-uniform scaling, but can be linearly parameterized with 6 parameters, cf. equation 6.1.

$$\mathbf{H}_{affine} = \begin{bmatrix} p_0 & p_1 & p_2 \\ p_3 & p_4 & p_5 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{H}_{perspective} = \begin{bmatrix} p_0 & p_1 & p_2 \\ p_3 & p_4 & p_5 \\ p_6 & p_7 & 1 \end{bmatrix} \tag{6.1}$$

Assuming perfectly gravity-rectified images, gradient descent-based image registration methods can improve in two ways. First, the optimization algorithm is much faster when using a gravity-rectified image as input and an affine transformation needs to be estimated instead of a perspective one. When the cost function needs to be updated at each iteration, an image warping using the transformation parameters is required. Since affine warpings do not require divisions, the cost function update is then much faster than when using generic perspective warpings based on standard homographies, which require divisions. Also, since affine transformations have 6 parameters to be estimated, solving the system of equations inside the optimization is much faster compared to perspective registrations $\mathbf{H}_{perspective}$ which require 8 parameters to be estimated, cf. equation 6.1.

Second, in addition to the improvements in terms of computational speed, the optimization algorithm converges more often when we take the knowledge that a gravity-rectified image is given as input into account. Using an *affine* version of the optimization algorithms gives a better convergence rate compared to the standard *perspective* versions. We validated this using the same Matlab tool and the same reference template of size (100×100) pixel as the ones used in [Bake 04]. This Matlab tool is available on the website of the Robotics Institute of the Carnegie Mellon University [Bake 14].

6.1.2 Evaluation Results on Synthetic Data

In order to compute the convergence rate, we warp the reference template 9,500 times using different random (but known) affine transformations. The affine transforms are computed by adding a Gaussian noise to the coordinates of 3 control points on the border of the square template (two corners of the template and the center of the other two corners). The standard deviation of the Gaussian noise is increased from 1 pixel to 10 pixels with steps of 0.5 pixels. We performed 500 random affine warpings for each standard deviation, which defines the amplitude of the affine warping.

On this data, we tried different registration methods namely the Forward Additive (FA) [Luca 81], the Forward Compositional (FC) [Shum 00], the Inverse Compositional (IC) [Bake 04] and the Efficient Second-Order Minimization (ESM) [Benh 04]. The affine and perspective version of each algorithm was used to register the template after each warping. In order to compute the convergence rate, we consider that an algorithm converged, if the spatial root-mean-square of the 3 control points

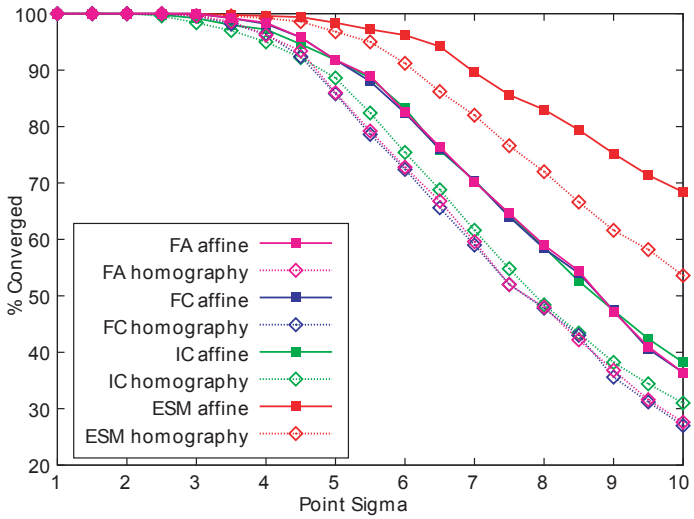


Figure 6.2: Comparison of the convergence rate of an affine template tracker with a perspective homography tracker on affine image transformations. As can be seen, the affine version converges more often than the one using a perspective homography for all tested methods.

is under 1 pixel after 15 iterations. For more information about the used benchmarking approach and the computation of the convergence rate, the reader can refer to [Bake 04]. Figure 6.2 plots the convergence rate for the different tested algorithms as a function of the amplitude of the warping.

It is possible to see that we get a higher convergence rate using the affine transform version of all tested algorithms and for all warping amplitudes than when using the version estimating a full perspective homography. As the amplitude of the warping increases, the rate of convergence of the methods that take into account the fact that the input image is gravity-rectified decreases slower than the rate of convergence of the methods that do not take the rectification into account. Similar results were obtained with all tested reference templates.

This means that performing a perfect gravity-rectification of the input image and using the affine version of the image registration improves not only the computational efficiency but also the convergence rate of their gradient descent algorithms. Note that these perfect conditions correspond to the template being perfectly horizontal and the obtained gravity vector being perfectly accurate and precise.

6.1.3 Evaluation Results on Real Smartphone Data

We evaluate the proposed method on real camera sequences with real inertial sensor readings using the tracking system explained in section 5.2.4 for initialization and the Inverse Compositional (IC) image registration method to track a template from frame to frame. If the IC tracker fails, i.e. the ZNCC between the reference image and the warped current image is below 0.7, the localization algorithm is used again to reinitialize.

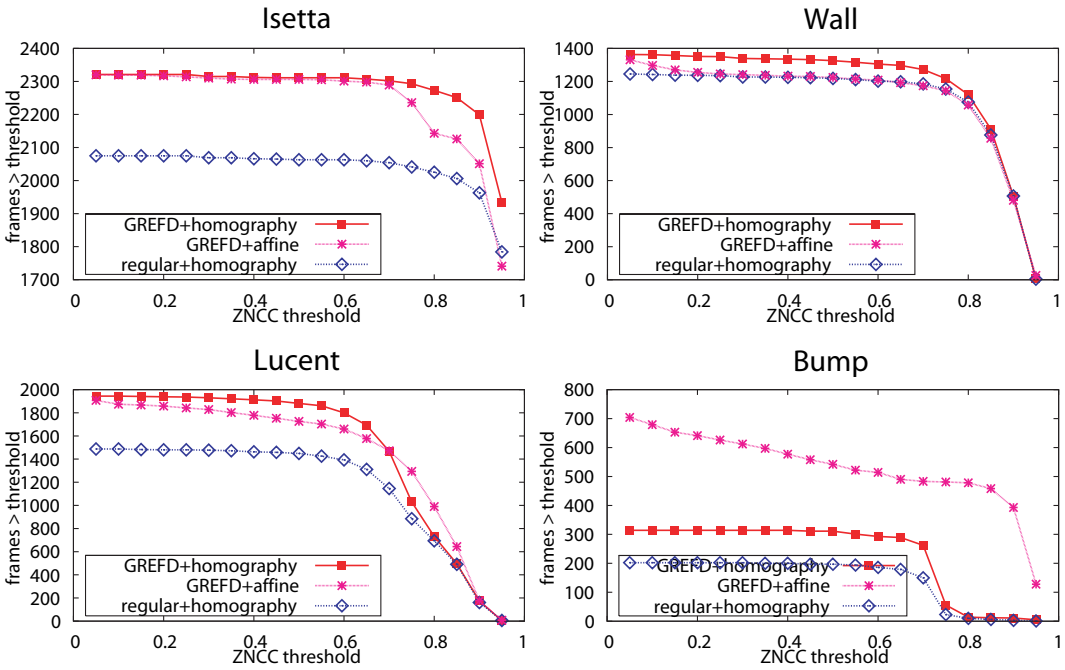


Figure 6.3: Number of frames where a transform was found resulting in a ZNCC greater than a threshold for a regular template localization and tracking system and a modified version using gravity-rectified images. In all tests, gravity-rectification increases the number of successful frames.

Below, we will compare three different tracking systems. The regular system uses regular feature descriptors for initialization and IC template tracking that estimates a full homography (*regular+homography*). We compare this tracker with two versions using GREFD (cf. section 5.2). While the first system uses an affine IC template tracker after GREFD-based initialization (*GREFD+affine*), the second uses an unchanged IC template tracker that finds a homography (*GREFD+homography*). The objective is to track (160×120) mm print-outs of the four template images shown in the upper row of figure 4.2 that are located on a horizontal table. To ensure a fair comparison, we captured two images sequences and the corresponding gravity vectors for each template image, and ran the tracking systems offline on a PC.

The resulting performance of the three different tracking systems is compared in figure 6.3. The plots show the number of frames where the ZNCC was above a particular threshold as a function of the threshold. In all cases, the two trackers using gravity-rectified feature descriptors (GREFD) clearly outperform the regular tracking system. The question, if affine tracking of the gravity-rectified camera image improves tracking compared to a regular homography-based tracker does not have an universally valid answer. While for the *Isetta* and the *Wall* template, *GREFD+homography* performs clearly better than *GREFD+affine*, it is not clear which system is better for the *Lucent* sequence. On the other hand, the *Bump* sequence clearly performs better with the affine transform approach.

6.1.4 Conclusions

The fact, that most real applications use template images that fall into a category with *Isetta* and *Wall*, led to the conclusion, that in practice we use *GREFD+homography*. While in theory affine tracking of a perfectly gravity-rectified image would outperform regular homography-based tracking as explained above, the gravity measures provided by current phones are in practice still too inaccurate and noisy. Furthermore, the readings of the inertial sensors are not perfectly synchronized with the camera images. Together with slightly inaccurate camera intrinsics, this not only results in a jittery but also a not perfectly rectangular appearance of the template in the gravity-rectified camera images.

Yet, our proposed method enables improving the invariance of image registration methods and template trackers to out-of-plane rotations, which is one of the objectives of this thesis, stated in section 3.2.1.

6.2 Enabling 6 DoF Edge-Based Tracking

Template tracking, as described above, works well in a static and well-textured environment. However, in real-world applications, parts of the template may be occluded in the camera image, the illumination may have changed non-uniformly compared to when the reference image was taken, or the template is hardly textured. In these case, edge-based approaches have shown to perform well.

This section studies how the concept explained above for gradient descent methods can be used with an edge-based method. We investigate the applicability of the edge-based object detection algorithm by Steger [Steg 02] for handheld AR applications. The original algorithm handles similarity transforms only, i.e. translation, scaling, and in-plane rotation. We present a simplified version that runs on mobile devices in real-time and show how an inertial sensor-based gravity-rectified camera image in combination with this approach enables estimating a full 6 DoF camera pose.

6.2.1 Original Approach

Given a reference image, we create synthetic views of the template under discrete samplings of all possible rotations and scalings. The step length used is 4 degrees for rotation and there are 20 supported scales uniformly distributed in the range $[0.6, 1.4]$. We use six pyramid levels with a scale factor of 1.5 between the individual levels to run the detection and tracking in a hierarchical coarse-to-fine manner. Then, the direction vector for each synthetic view at each pyramid level and for all rotations and scales is computed, as explained in the original paper [Steg 02]. Therefore, we first compute the Sobel filter and then use a threshold that removes all edges with a magnitude of less than 0.07 for image intensities of a normalized image. These computations take about 14 seconds on an iPhone4 and need to be done only once offline.

In what follows, we will first describe the procedure carried out to find the transformation of the template in each current camera image without incorporating gravity measurements. We will then show how a gravity-aware approach is able to overcome the limitations of the standard approach.

For each current camera image, we first create an image pyramid with downscaled versions of the image and then compute the direction vector for each pyramid level in the same manner as for the reference template. If we do not have an approximate scale, position and orientation of the template in the current image, e.g. from the last camera image, we start by finding the maximum similarity (as in [Steg 02]) in the transformation space on the highest pyramid level using exhaustive search.

Given this coarse transformation, we then track the highest similarity down the pyramid by computing the similarity measures with the neighborhood in transformation space for each pyramid level. When tracking multiple frames where the current transformation of the template can be considered similar to the one in the last current image, we skip the exhaustive search on the highest pyramid level and use the transformation from the last frame as a starting point instead.

The explained algorithm requires the precomputed direction vectors of all the possible warpings of the template image to be quickly accessible. The only solution to this on mobile devices is to store them in main memory. Depending on the size of the template image used and the sampling resolution of the transformation space, the direction vectors easily require 200 MB of main memory. As the memory consumption grows exponentially with the supported degrees of freedom, it is not feasible to support perspective transformations, which would increase the dimensions of the precomputed transformation space from 2 (i.e. scale and in-plane rotation) to 4 (i.e. scale and all 3 rotation axes). Assuming 60 steps per DoF this would result in $60^4 = 12,960,000$ instead of $60^2 = 3,600$ precomputed direction vectors. This will not only increase computational complexity but more critically exceed the memory available on current mobile devices, making 6 DoF tracking impossible with this kind of algorithms.

6.2.2 Proposed Approach Based on Gravity-Rectification

We propose to enable 6 DoF pose estimation from a horizontal template by using the 4 DoF edge-based detection algorithm and considering the measured gravity vector. Given a current camera image, taken under an arbitrary view angle that is not perpendicular to the template, the edge-based tracker is not able to find the correct transformation, as it supports similarity transforms only, cf. figure 6.4 a. While the illustrated transformation is the one providing the highest similarity in the class of similarity transforms, it is clearly wrong. We propose to first rectify the camera image based on the gravity as described in section 5.2. We then proceed as described above on the gravity-rectified camera image which will result in a similarity transform between the template image and the rectified camera image, see figure 6.4 b. By applying the inverse of the transformation used to rectify the camera image to the four corners found in the rectified image, we gain the position of the corners in the original image, as displayed in figure 6.4 c. These, together with the intrinsic camera parameters, can then be used to

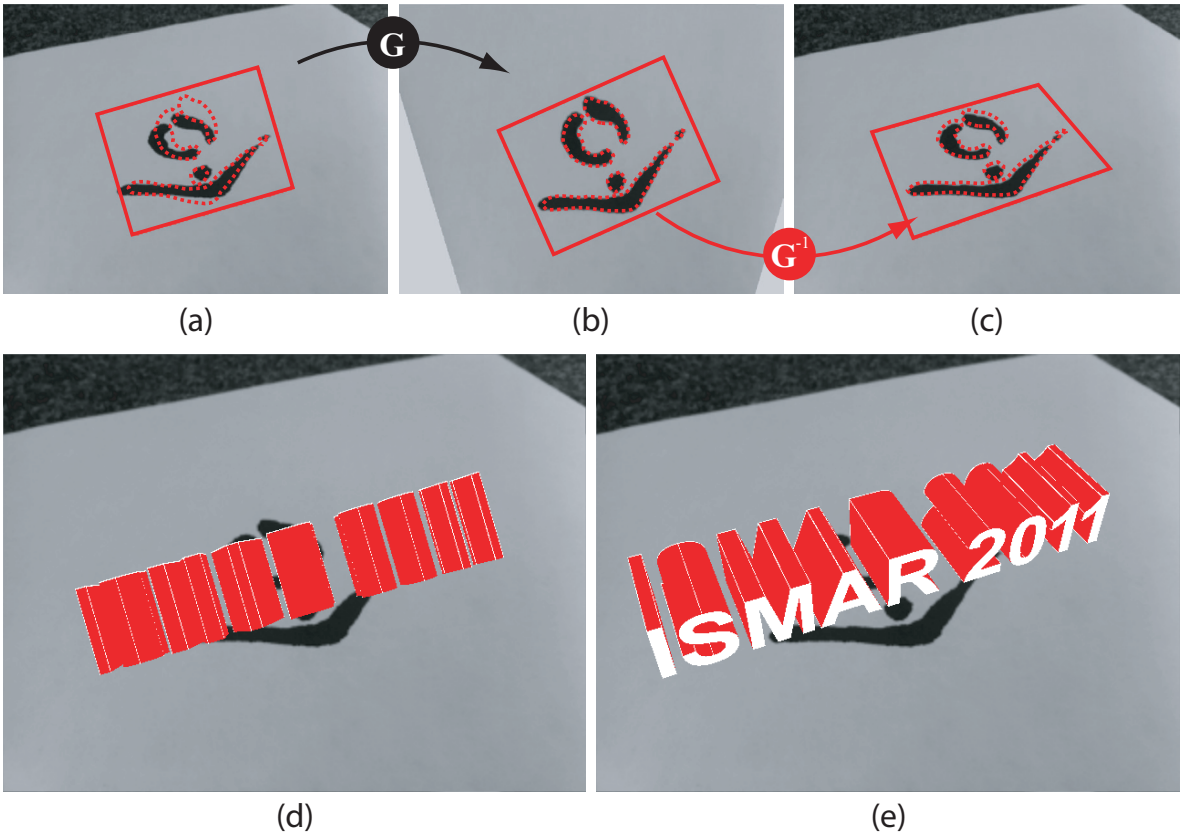


Figure 6.4: As soon as the camera's principal axis is not perpendicular to the tracked target, similarity tracking cannot estimate the transformation the template underwent in the camera image correctly (a,d). Using the proposed inertial sensor-based rectification of horizontal targets (b), a similarity tracker is able to estimate the transformation which results in a proper pose (c,e).

compute a full 6 DoF camera pose which can be used to display virtual 3D objects spatially registered with the template as shown in figure 6.4 e. In figure 6.4 d, we see that similarity tracking with no rectification results in incorrect augmentation.

6.2.3 Conclusions

As discussed in section 3.2.1, invariance to out-of-plane rotations is an important property of approaches to camera localization and tracking which is often challenging and in case of the presented edge-based tracking even not feasible on mobile devices. Our proposal to use inertial sensor-based gravity-rectification of the camera image to enable 6DoF pose estimation, overcomes this limitation when dealing with horizontal planar objects. Figure 6.5 shows this algorithm running on an iPhone 4 in real-time. A colorful logo is correctly tracked, even though it is partially occluded by a ballpen.

Figure 6.5: Gravity rectification-enabled 6DoF edge-based tracking running on a mobile phone.

6.3 Inertial Sensor-Enabled Gravity-Aware Augmentations

Above, we presented different ways of aiding computer vision algorithms that are used for camera pose estimation. The benefit of an improved camera pose for handheld AR becomes visible, when augmenting the camera image with virtual 3D objects. The more accurate and stable the estimated pose is, the better and more realistic the augmentation becomes.

In this section, we present an approach to use knowledge of the direction of gravity to directly improve augmentations in handheld AR. We show that realism and plausibility of augmented 3D models can be increased by adding simple physics to the model that are based on the gravity vector. The gravitational force influences the visual appearance of objects in the real world. Therefore, it makes sense to simulate this effect also in the virtual part of an Augmented or Mixed Reality scene.

In classical video-see-through AR applications, there exist two coordinate systems where one is associated to the tracked object O and the other one is attached to the camera C. The aim of visual camera pose tracking is to find the transformation from the object coordinate system to the camera coordinate system to enable rendering virtual objects attached to the real object. Therefore, this transformation can be assumed to always be estimated. In gravity-aware AR a third coordinate system comes into play: the world coordinate system W. As shown in figure 6.6, the gravity vector $\mathbf{g}_w$ is static and known in world coordinates. What is needed to enable gravity-aware augmentations registered with a real object, is the gravity vector in the object coordinate system O, which we denote with $\mathbf{g}_o$.

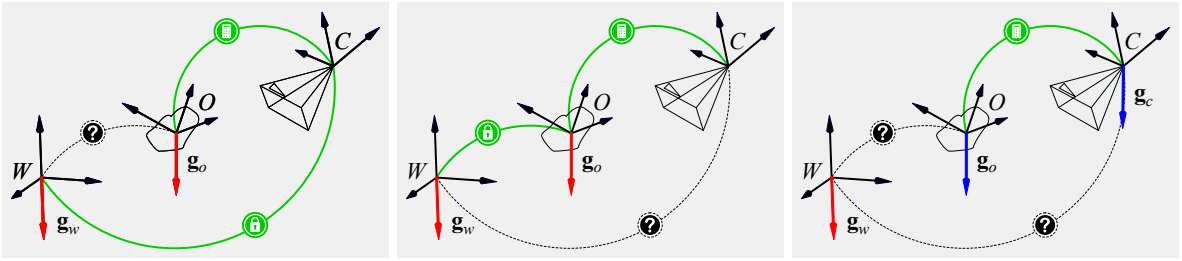


Figure 6.6: Coordinate systems and the transformations between them for a setup with a static camera (a), applications where a moving camera tracks a static object (b), and a setup where both the camera and the object may move unconstrained (c).

If the camera is static, as for instance the case for most kiosk AR applications, and shown in figure 6.6 a, the gravity vector can be transformed to camera coordinates ($\mathbf{g}_c$) given the static and known camera transformation. Now if visual localization and tracking estimates the transformation between the camera and the object, the gravity vector can further be transformed to the object coordinate system resulting in the desired $\mathbf{g}_o$. In handheld AR applications that track a static object with a known orientation, the gravity is constant with respect to the coordinate system of the object, see figure 6.6 b. It can be easily be transformed to the object coordinate system once and then be used as $\mathbf{g}_o$. In these two configurations, gravity-aware augmentations can be implemented without any inertial sensors.

However, in handheld AR applications that track a real object which may freely move and rotate, neither of the above mentioned approaches works, see figure 6.6 c. No transformation is given that would allow for transforming $\mathbf{g}_w$ into the object coordinate system O to obtain the needed $\mathbf{g}_o$. But in this case, the gravity in the camera coordinate system $\mathbf{g}_c$, as measured with inertial sensors, can provide the required information. We propose to transform this vector to the object coordinate system using the transformation estimated in visual tracking and use it to improve the visual appearance of the virtual augmentation attached to a freely moving object. The pose used to render gravity-aligned parts in an augmentation combines the translation given from the visual tracking and the rotation from the gravity measurement. This kind of transformation was also used in [Oda 10] to control an AR game on an inertial sensor-equipped HMD.

Figure 6.7 compares the augmentation of a static 3D object representing a candle on a muffin with a gravity-aware version where the fire is aligned with the direction of gravity. As long as the plate, which is the real object being tracked in this case, is located on a horizontal plane as shown in figure 6.7 a, the static augmentation appears plausible. However, when it is rotated as in figure 6.7 b, the static augmentation looks unrealistic which completely destroys the illusion of Augmented Reality. Figure 6.7 c, however, provides a realistic appearance thanks to inertial sensor-enabled gravity-aware augmentations. While this example aligns the articulated part contrary to the direction of gravity, most other example applications, such as a tire swing mounted to a tree, would align the articulated parts downwards aligned with gravity.

Figure 6.7: A rigid 3D model of a candle looks plausible while in a vertical orientation (a). When tilted, the fire of the rigid model rotates with the candle resulting in an unrealistic appearance (b). Aligning the fire with gravity gives a believable impression (c).

We believe, that besides robust and fast tracking, high quality and plausible 3D augmentations are critical for future (handheld) Augmented Reality applications.

7 Conclusions and Future Work

This thesis proposed different methods to advance individual components of a handheld video-see-through AR pipeline, where a particular focus lies on improving the invariance and distinctiveness of visual feature descriptors. Based on different test data and evaluation methods, we could show and quantify the achieved enhancements. We also made a test dataset available to the research community hoping it will stimulate and support further research.

While each of the methods explained in chapter 5 addresses different particular problems, they all follow an overall approach. In the following, we formulate and discuss this overall approach and look at it from different perspectives. This chapter will further summarize and discuss the achievements made in this thesis and finally take a look into future research directions related to this work.

7.1 Overall Approach

The overall approach pursued in this thesis is to make use of the auxiliary sensors built into mobile handheld devices to aid handheld AR. This has been done in two ways.

Firstly, some of our approaches aim at solving problems which could be solved entirely image-based, but with the help of auxiliary sensors, they can be solved more efficiently and potentially more accurately. This is for example the case for Gravity-Rectified Feature Descriptors (cf. section 5.2) and will be further discussed in section 7.1.1.

Secondly, we proposed approaches to solve problems, which cannot be solved only based on a single image, such as distinguishing visually identical features that only differ by their absolute position, absolute orientation, or absolute scale. We will elaborate on this aspect in section 7.1.2. Some of the developed approaches fall into both groups, as we will see in the following.

All approaches described in this thesis are general in the sense that they do not refer to a particular method for features detection, feature description, or template tracking. They can be applied to any state-of-the-art method and also future methods will be able to benefit from the contributions of this thesis.

7.1.1 Shifting Responsibility From Vision to Inertial Sensors

The main motivation for shifting as much responsibility as possible from computer vision (i.e. the pixels in a camera image) to inertial sensor measurements is that these are more reliable. Particularly the measured direction of gravity is not influenced by projective foreshortening, illumination changes, partial occlusions, repetitive structures or any other visual artifacts that can be present in camera images, such as noise, blur, or oversaturation. While the process of feature descriptor orientation assignment is highly influenced by these parameters when done based on the camera image, our proposed method to align them with the measured direction of gravity instead, cf. section 5.1, is not affected at all. Furthermore, it is significantly less computationally expensive than accessing pixels, computing gradients, etc., as required for image-based approaches.

Besides the fact that inertial sensor measurements are more reliable, they can additionally help to simplify problems, such that vision methods can be more efficient in terms of runtime and partially also memory usage. This is particularly the case for all approaches based on gravity-rectification of the camera image for planar horizontal objects. It enables using simple feature detectors and description methods (GREFD, cf. section 5.2) instead of more expensive approaches that detect affine regions and use them to normalize image patches to account for the effects of projective foreshortening. For template tracking this means, that a similarity tracker can be used instead of a perspective one without loss of supported degrees of freedom. As shown in section 6.1, such trackers are not only faster than perspective ones but they also converge more often. For the edge-based template tracking approach explained in section 6.2, gravity-rectification of the camera image is an enabler for running them on mobile devices. The reason is, that this method requires to hold the direction vector of sampled views within the supported transformation space in main memory. This is obviously less demanding for the required 2 DoF after gravity-rectification than 4 DoF without using inertial sensors.

The algorithm proposed in section 5.3 uses a measurement of the gravity vector to choose between different sets of reference descriptors based on the pitch orientation of the camera. While this could potentially also be done based on the camera image, e.g. using machine learning approaches or keyframe matching, using inertial sensors instead is much easier, more efficient, and very robust, as is also the case for the other approaches discussed above.

In the future work section (7.4), we will discuss further possible implementations of the concept of shifting responsibility from vision to auxiliary sensors.

7.1.2 Decoupling Invariance in Image Space From Invariance in World Space

Most computer vision methods do not distinguish between motion of the camera and motion of the world (or object) they deal with, simply because they cannot. Analogously, the common approaches to visual feature detection and description do not distinguish between invariance in the coordinate system of the camera (image) and invariance in the coordinate system of the world. Rotation-invariant descriptors override the effect of changes in orientation of both the camera and the world. Translation of the camera or the world can result in changes in scale in the camera image, which cannot be distinguished from actual changes in physical scale in the world. Translation in 3D space can further result in translation in the camera image, which common approaches are invariant to thanks to salient feature detectors. Again, it does not make any difference to these approaches if the world moved relative to the camera or vice versa.

Under the assumption of a (predominantly) static world (or environment, or real object), invariance to camera motion is clearly required, while invariance to motion of the world is not needed. On the contrary, it is even not desirable because it may lead to confusion of features that are interpreted as having moved but in fact are different physical features in the world. Given only camera images, it is impossible to tell the difference between motion of the camera and motion of the world, as stated above. Given auxiliary sensors, which can measure the *absolute* orientation (and position) of a camera, however, we are able to distinguish between the two. And assuming a static world, we discussed several approaches in chapter 5 that aim at decoupling invariance in image space from invariance in world space. This approach of making visual feature detection and descriptors invariant to changes in position, orientation, and scale, in the camera (image) coordinate system while disabling invariance in the world coordinate system, results in an improved distinctiveness, as postulated in section 3.1. It enables detecting features only at *regions of interest in the world*, while providing full invariance to the position in the camera image coordinate system in section 5.4.2.2. It further makes features distinctive which are impossible to distinguish with vision only such as the four corners of a window (cf. section 5.1) or the windows on a façade, cf. section 5.4. This approach further allows to distinguish similar looking features that only differ in their physical scale, as shown in figure 3.1.

A further example of approaches to problems that cannot be solved entirely image-based are sensor aided (e.g. gravity-aware) augmentations for handheld AR, as discussed in section 6.3. Here the assumption on the world (or in this case object) is different as we allow unconstrained motion of both the camera and the object. In this case, the absolute orientation of gravity, which shall affect the visual appearance of virtual objects attached to the real object, can only be reliably obtained by means of sensors in addition to the camera – in this case inertial sensors.

We will discuss more examples where auxiliary sensors enable solving problems, which are impossible to be solved using monocular image information only, in the future work section 7.4. There we will see how the concept of decoupling invariance in relative coordinate systems from invariance in absolute coordinate systems can be further applied to different domains.

7.2 Achievements

In the course of this thesis, we accomplished to improve the invariance of visual feature descriptors and template trackers, and to improve distinctiveness of feature descriptors. Quantitative evaluations confirmed a positive impact of these improvements on real data representing real-world problems. Overall we could improve the naturalness of handheld AR, which will be elaborated below. Applications of the proposed approaches in the wild and coverage in popular media, as well as the underlying peer-reviewed publications and patent applications, as disclosed in section 1.3, emphasize the relevance and importance of the achievements made in this thesis both from an academic standpoint and for applications solving real-world tasks.

7.2.1 Impact of Improved Invariance and Distinctiveness on Real-World Tasks

All approaches developed in this work have been systematically evaluated on realistic data. This is important to quantify the effect of conceptual improvements in terms of invariance and distinctiveness on real-world problems. Using our custom feature descriptor Mobile48, we showed that Gravity-Aligned Feature Descriptors, as proposed in section 5.1, enable the correct localization of vertical planar objects for 18.76% more images in a dataset of 48,000 images compared to a regular approach, cf. table 5.3. For horizontal planar objects imaged under steep angles, which is particularly challenging, our proposed Gravity-Rectified Feature Descriptors (section 5.2) increase the ratio of correctly localized images by 72.26% in an implementation based on the commonly used SURF descriptor, see table 5.4.

The approach to compute Representative Feature Descriptor Sets to model an object, cf. section 5.3, increases the number of images for which a planar object could be correctly localized in an image by 21.96% for arbitrarily moving planar objects and 23.84% for the gravity-aware approach, which can handle objects at an arbitrary but static orientation with respect to gravity, cf. table 5.4.

The biggest improvements compared to a vision-only approach could be achieved by computing the Absolute Spatial Context for feature descriptors in outdoor environments (section 5.4). Here particularly the improved distinctiveness allowed for the correct estimation of the camera pose for more than 3 times as many images as the baseline method (i.e. an improvement of 209.07%, see table 5.5). Thereby the expensive comparison of the visual descriptor only needs to be performed for 2.16% of the combinations of reference features and the features in a current camera image, which additionally provides a significant speed-up, see table 5.6.

These numbers are based on a comprehensive evaluation dataset for outdoor camera localization comprising of over 45,000 real camera images, which we presented in section 4.4, and which has been made available to the public. Note that all numbers given above refer to *relative* percentages of improvement.

The improvements to template tracking based on gravity-rectification of the camera image as explained in section 6.1.1 could only be shown on synthetic data. As it turned out, the accuracy and precision of the inertial sensors used in the evaluation was not sufficient for the proposed method to improve. We also cannot give any quantitative data with respect to the edge-based template tracker in section 6.2 simply because the algorithm does not work at all on a mobile device without our proposed modification. In this case, auxiliary sensors do not lead to improvements but they enable approaches which would not be feasible without such sensors.

7.2.2 Improved Naturalness of Handheld AR Experiences

On a higher level, the approaches proposed and discussed in this thesis aim at providing an improved and more natural experience to the users of handheld Augmented Reality applications. Faster and more robust initialization of camera tracking methods result in a more natural experience to a user, simply because the application just works and does not require moving the camera to a certain viewpoint to initialize. Improvements in tracking both in terms of accuracy and speed also immediately result in better experiences because the ideal AR user interface has no delay and perfectly registered augmentations.

Finally, we also presented means to increase the visual plausibility of virtual augmentations in section 6.3. We believe that all these steps are fundamental on the way to make Augmented Reality ready for the mass market.

7.2.3 Successful Applications in the Wild

Most of the approaches discussed in this thesis have been implemented and made available as features of the Metaio SDK [Meta 14b], which is an Augmented Reality SDK for iOS, Android, and the Windows platform. Thereby, technologies such as Gravity-Aligned Feature Descriptors or Gravity-Rectified Feature Descriptors are accessible to over 80,000 active developers worldwide and are being used in a wide range of real-world applications.

Making gravity-aware AR available to the market, also gained attention of popular technology news websites, such as Engadget[1] and TechCrunch[2], who reported on novel approaches, which are part of this thesis.

In particular Gravity-Aligned Feature Descriptors are frequently used and were a key component of Metaio's tracking framework, which we employed to win the 2011 ISMAR Tracking Competition[3] in Basel, Switzerland.

[1] http://www.engadget.com/2012/02/03/next-gen-augmented-reality-from-arm, accessed February 2014
[2] http://techcrunch.com/2011/10/27/metaio-adds-gravity-to-their-augmented-reality-platform, accessed February 2014
[3] http://ismar2011.vgtc.org/images/ismar2011_trackingcompetition.pdf, accessed February 2014

7.3 Discussion, Limitations, and Drawbacks

One obvious limitation of most approaches discussed here, is that they require auxiliary sensor hardware in addition to a camera. This adds both to the purchasing cost of a handheld device and to its power consumption. Furthermore it excludes using devices without the required sensors. In fact, except for the energy consumption, this limitation is negligible, because virtually all off-the-shelf handheld devices are equipped with the sensors we need, thanks to their various uses beyond the scope of this thesis and Augmented Reality.

The fact that the required sensors are available, however, does not necessarily mean that they provide measurements consistent with our expectation. For example, they may provide completely wrong values when their driver crashed, or they provide measures in a coordinate system different from what is expected. It can also happen, that the accuracy or precision of a sensor built in a certain device is much worse than what our methods require and we implicitly expect. These situations not only put the improvements we could achieve thanks to auxiliary sensors at risk. They may lead to a complete failure of a localization and tracking system and therefore to much worse results than a vision-only approach would provide. This is a clear drawback of our proposed methods that rely on auxiliary sensors, when used outside controlled lab environments. We will discuss in section 7.4 how such situations can be dealt with in future work.

It is debatable if retrieving more information from additional sensors because an image does not provide sufficient information is a reasonable approach given that we ignore lots of information contained in the images we deal with. In fact, all approaches discussed throughout this work, convert color images to grayscale before processing them, which means we ignore available information on the hue and saturation of features in an image. In terms of distinctiveness, this means, that we accept not being able to distinguish, say, a yellow car from a cyan car, while the information to doing so actually is available, and there are approaches making use of it, e.g. [Sand 10]. The first reason to use grayscale images instead of color images is that processing them is usually faster, which is crucial for real-time applications. Secondly, it depends on the application if a descriptor should describe the color of a feature, or its physical scale, or if it should be invariant to these properties. The approaches pursued in this thesis are problem-driven in the context of handheld Augmented Reality, where for example being able to distinguish the four corners of a window is more important than being able to distinguish windows with different colors. However, incorporating color information clearly remains a topic future work could look into.

For every piece of information indicative of an *absolute* value, e.g. absolute orientation (with respect to gravity), absolute (physical) scale, or possibly absolute wavelength, there are two approaches to use them to improve the distinctiveness of the description of a feature. One option is to use it for normalization of the descriptor, as for example done with absolute orientation in GAFD (section 5.1), and with absolute scale in [Smit 12]. The other approach is to store the absolute information as part of the descriptor, as for example done in the *regular fast* approach in section 5.1, and with absolute

scale and absolute position in our approach to make feature descriptors aware of their Absolute Spatial Context (ASPAC) in section 5.4. An interesting point for discussion is which of these approaches is superior under which circumstances, or if even one of them always outperforms the other.

While we cannot expect an all-encompassing answer to that in this thesis, the results of section 5.1, where we compare both approaches at the example of absolute orientation, tell that in this case normalization outperforms adding the information as part of the descriptor. However, if the absolute information is known to be inaccurate, as the case for the estimated absolute scale based on GPS in section 5.4, using this information for normalization is probably not a promising idea. Using it as part of the descriptor instead, as done in this work, gives the opportunity to define a threshold for matching, which can be adapted to the expected accuracy. Furthermore, the normalization approach only seems reasonable if the used dimension has a fixed and assessable range. Absolute orientation in the range $[-\pi, \pi]$ is a good example where it works. Normalizing image patches based on their absolute position in the geographic coordinate system, e.g. describing them relative to the North Pole, intuitively does not sound like a good idea.

In the following section, we will discuss future work including the application of the two approaches to incorporate absolute measurements into the feature description process to different domains and dimensions.

7.4 Future Work

In the future, we aim to further pursue the general research directions of this thesis. Both the distinctiveness and invariance of means to describe features and objects in our world need to be further increased given additional contextual information retrieved from sensors. This is important for enabling the vision of the Augmented City and the Internet of Things, where each real object has a virtual representation and digital information associated to it. Instead of being used in single-purpose apps, as currently mostly the case, Augmented Reality will become a unified user interface which provides the user with relevant information on their surroundings and all the things in their field of vision. There are more challenges involved in the scaling of AR, which we will discuss below. Furthermore, the naturalness of AR experiences becomes increasingly important when deployed ubiquitously. While we made first steps in this direction, future work will not only have to deal with an increased realism of visualizations, but also, very importantly, with natural and intuitive user interfaces for AR.

7.4.1 Further Improvements by Exploiting the Full Hardware Capabilities and Online Calibration

In this work, we do not take advantage of the full capabilities current and future handheld devices offer. In the following we discuss how fully exploiting them could further improve the results of our proposed methods.

Throughout this work, we associate to each camera image a single gravity measurement and – depending on the approach – additionally a single GPS-based location and a single compass-based measurement of device heading. These measurements are queried from the respective APIs immediately after each other and we assume they all correspond to the time when the image was captured. This is not necessarily the case. During slow motion of the handheld device, missing synchronization between the sensor readings does not have a major impact. However, during rapid device motion, accurate synchronization could further improve the benefits of our proposed methods. This requires timestamps for all measurements that ideally come from the same clock source. The gravity vector corresponding to a given camera image with a known timestamp could then be determined from a set of gravity measurements with timestamps using interpolation or by combining all measurements during exposure time of the image in a meaningful way.

Such synchronization would result in more consistent input data to our methods, i.e. an image and auxiliary sensor readings. For example Gravity-Aligned Feature Descriptors could benefit from better synchronization because the descriptors would be more accurately aligned with gravity – particularly during rapid camera motion. All approaches based on gravity-rectification of camera images would benefit from a more accurately rectified image and our proposed method to provide camera features with an Absolute Spatial Context would be based on a more accurate sensor pose which fits better to the camera image and therefore results in better descriptions of the camera features.

This would be one approach to take advantage of the considerably higher sample rate of inertial sensors compared to the frame rate of cameras in handheld devices. There are also approaches to use high-frequent inertial sensor readings to compensate for motion blur, e.g. [Josh 10], which could be beneficial as a pre-processing step to our methods.

Besides inaccurate inertial sensor readings that are not synchronized with the camera images, another reason why working with gravity-rectified camera images does not always improve results on real data, might be the interpolation involved in warping the gravity-rectified image. None of the methods discussed in this work makes use of the maximum image resolution available but we usually use images with (480×360) pixels. The impact of artifacts resulting from image warping could be reduced when using higher resolution camera images for rectification and potentially scale them down after rectification for further processing and analysis.

7.4.2 Further Increasing Distinctiveness and Invariance

The proposed concepts of GAFD, GREFD, and Representative Feature Descriptor Sets are essentially concerned with the distinctiveness and invariance of image patches to (3D) rotations. Therefore they are general enough to not only be used with the methods in our evaluations. Instead, they can be applied to improve any approach to matching features that is based on an image patch around a feature, including binary descriptors and feature classifiers.

On the other hand, gravity-alignment of image patches results in properties of the normalized patch, which could be explicitly exploited by novel feature descriptor layouts. Because the pixels along the gravity axis originating from the center of the patch are much more invariant to out-of-plane rotations than those further away from the axis, anisotropic descriptor layouts may be beneficial here.

The proposed improvements on feature description and template tracking furthermore can be used in applications beyond the scope of AR and real-time pose estimation. Example use cases include offline 3D scene reconstruction and information retrieval on large image databases. Most images in such databases carry EXIF information indicative of the coarse orientation of the camera with respect to gravity. This could be exploited with our proposed *regular fast* feature description method, which stores the global orientation as part of the descriptor to improve information retrieval results.

Also global image descriptors, which become more important with growing databases of objects to deal with, can benefit from the contributions of this thesis. As these global image descriptors are often based on local feature descriptors, they can use our proposed methods right away, as for example done in [Gui 13], which uses GAFD-based image descriptors for landmark recognition on smartphones.

The concept of decoupling invariance in relative domains from invariance in absolute domains, as explained in section 7.1.2, can be generalized and applied to more domains in the future. Assuming a constant environment with constant illumination, feature descriptor methods need to be invariant to changes in brightness of certain features in the camera image as a result of auto exposure and other parameters of the camera. But it is not necessary, and potentially even self-defeating, to be invariant to changes of physical luminance in the scene. Ambient light sensors, which are built into most handheld devices and are indicative of absolute luminance, may allow for such approaches that are capable of distinguishing similar looking features with different luminance. On the contrary, it is crucial to develop methods for visual classification and localization that are particularly invariant to illumination changes for outdoor applications.

The cameras built into mobile devices also evolve and passive and active stereo cameras, e.g. [Prim 14], will provide the depth of pixels. Thereby information on accurate absolute (physical) scale will be provided, which novel feature description methods can take advantage of to increase both invariance and distinctiveness. It is also likely, that light field cameras will be built into future handheld devices enabling to re-focus an image after it was taken. Increasing invariance to defocus is only one possible approach to exploit the opportunities such cameras provide for future work. In general, devices will include more and more sensors which allow for richer descriptions of features in the environment and therefore allow to distinguish more and more features and objects.

The approach based on synthetic views of an object to localize, presented in section 5.3, could be further improved by modeling artifacts such as noise and blur to increase the representativeness and therefore invariance of the resulting descriptor sets with respect to these artifacts. Given that there will be increasingly more images available of any real object on Earth, the approach could use different actual images of the same object, comprising real artifacts, instead of using simulations thereof.

7.4.3 Further Increasing Scale

Dealing with a very large number of real objects and places imposes several challenges for AR applications and requires distributed (e.g. client-server-based) approaches to camera localization and tracking. Firstly because the database of real objects will become too big to be stored on a mobile device. And, even more importantly, this database is frequently subject to updates, whenever new products become available or new issues of print media are published. A variety of research topics in the area of data representation and information retrieval arise on the way to making this perform in real-time for a user base which will also scale in size.

With an increasing number of users, usually also the diversity of devices used will increase. Different devices include different sensors of different quality. While all approaches described and evaluated in this thesis were tested on devices that are equipped with the needed sensors, it is an important field of future work to assess availability and precision of auxiliary sensors on mobile devices and to develop approaches for camera localization and tracking that adapt to the device's capabilities. As a simple example, such approach would use Absolute Spatial Context-Aware Feature Descriptors in case GPS, compass, and inertial sensors are available. It would use Gravity-Aligned Feature Descriptors if only inertial sensors are available and fall back to regular feature descriptors in case the device in use does not include inertial sensors or they are too inaccurate to be employed.

Besides the availability of sensors, also their calibration becomes more challenging with an increased diversity. For all approaches discussed in this work, it is crucial to know parameters such as the focal length of the camera and the transformations between the coordinate systems of the individual sensors. While it is a suitable first step to calibrate these parameters offline and assume they are static and approximately the same for all devices of the same type, it is clearly desirable to re-calibrate them online for more accurate estimates. Also camera intrinsic parameters will not remain static anymore in future, given that there are already smartphones with optical zoom lenses. Therefore, online calibration of such device parameters, as for example done in [Drum 02], will play an important role. Furthermore online estimation of the biases of the individual sensors will be crucial in future systems.

A fundamentally important aid for any computer vision-based approach to camera localization in large-scale is a positioning system providing the coarse position of the camera. While GPS provides a somewhat unsatisfying accuracy outdoors and no reliable position indoors, the GALILEO positioning system [Euro 14], claims to provide significantly higher accuracy outdoors in the future. Additionally, Bluetooth low energy proximity sensing will provide positioning information indoors at many places, soon. Both an increased accuracy of positioning outdoors and the emergence of ubiquitous positioning indoors open up new possibilities to aid accurate visual 6DoF camera localization.

Usage of Augmented Reality technology will not only scale in terms of number of users, but also each user will use it much longer per day once it became an everyday user interface. Here, energy efficiency becomes an important topic for future work. There will be increasingly more Augmented Reality and computer vision hardware IP in mobile devices. For example the approach to feature

matching explained in section 5.4.2.4 is particularly well-suited to be implemented in hardware. Hardware implementations of compute-intensive parts in the Augmented Reality pipeline, such as the AR Engine [Meta 14a], will reduce the power consumption significantly to finally enable the vision of ubiquitous Augmented Reality – *Always On, Always Augmented*.

7.4.4 Further Increasing Realism, Coherency, and Naturalness

In many use cases of Augmented Reality, realism of the visualization of virtual objects is crucial. If a user wants to know if certain furniture fits in their apartment, or if certain glasses fit in their face, Augmented Reality technology can only be useful if it *reliably* provides a *realistic* preview. Most fundamentally, scaling and registration between any virtual object and the real environment need to be consistent. But future work should also further look into the field of coherent augmentations, beyond simulating imaging artifacts [Klei 10]. A reconstruction of the real environment, and particularly the real light sources, enables rendering virtual objects in a way that is visually coherent with the remaining real environment in terms of illumination, e.g. [Grub 12]. Sensing or obtaining additional information on the environment, e.g. wind speed and temperature, could be employed to additionally increase realism and coherency of virtual objects in future AR systems.

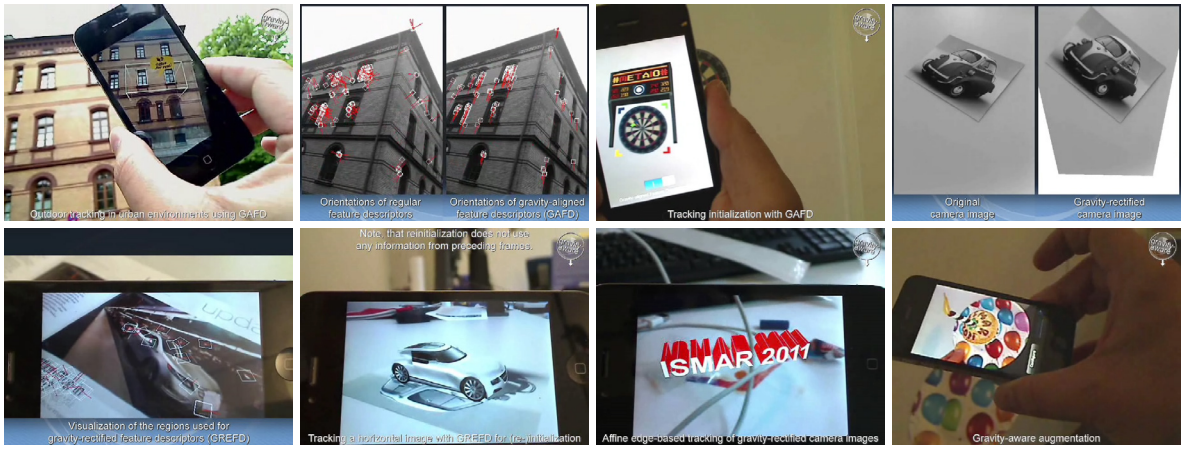
The form factor of smartphones and tablet PCs is a major hurdle towards the everyday usage of AR. All approaches discussed in this work can not only be applied for handheld AR but also for wearable AR, e.g. based on light-weight head-mounted displays (HMDs) which are currently becoming more and more common and affordable. These devices render permanently holding up a device, such that the camera captures the object or environment of interest, unnecessary and they contain all hardware and sensors needed. The fact that head motion is heavily constrained, as opposed to the motion of a handheld device, may also inspire new methods for localization and tracking of HMDs.

Wearable computers, particularly HMDs, call for another interesting and challenging area of future research. While in handheld AR, the majority of user interfaces is based on touch screens, HMDs demand different means to interact with both real and virtual objects in AR applications. There is a whole body of approaches to natural user interfaces (NUI) and all of them have their individual strengths and weaknesses. A truly natural user interface for future Augmented Reality systems might be based on the output of different complementary interaction techniques, such as hand-based interaction, gaze tracking, voice control, and brain-computer-interfaces, and – of course – sensor-aided visual camera localization and tracking!

A Appendix: Supplemental Videos

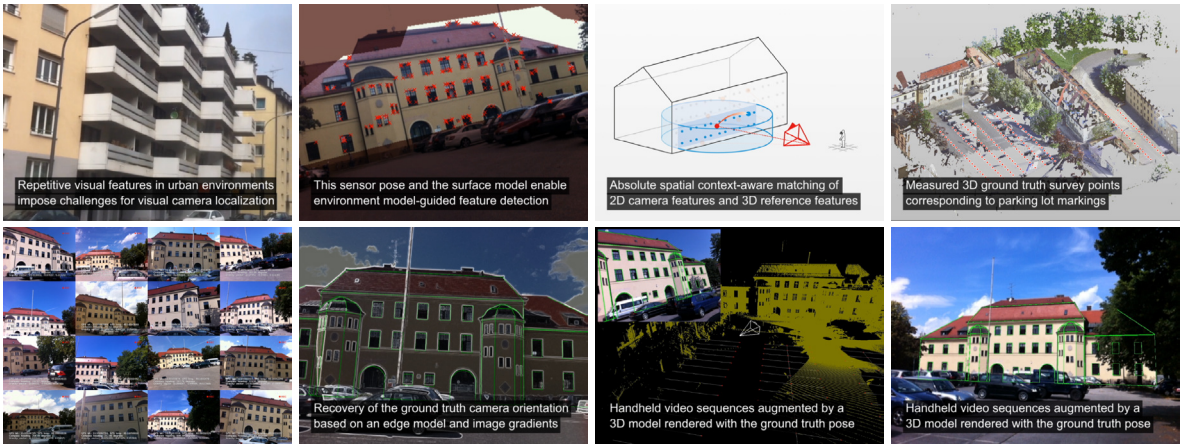
Many of the topics discussed in this thesis can be well explained with the aid of videos. The printed version of this thesis includes a CD with two AVI video files. This appendix briefly describes these videos that visualize a majority of the contributions of this work in an intuitively accessible way.

`Video_ISMAR2011.avi`



This video shows qualitative results for Gravity-Aligned Feature Descriptors (cf. section 5.1) as well as Gravity-Rectified Feature Descriptors (cf. section 5.2) when used to initialize tracking of planar objects. The viewer can further see the effect of gravity-rectification of camera images. Based on this, edge-based template tracking enables 6DoF camera pose tracking on a mobile phone, as proposed in section 6.2 and also visible in the video. Sensor-aided augmentations, as explained in section 6.3, are also shown in the video at the example of a candle with a gravity-aware flame.

Video_VISAPP2014.avi



The second video shows the approach to camera localization outdoors based on environment model-guided feature detection and providing feature descriptors with an Absolute Spatial Context, as described in section 5.4. It further explains the procedure to create the outdoor ground truth dataset introduced in section 4.4. Some of the camera sequences of this dataset along with the corresponding ground truth camera poses and the environment model are also shown.

Bibliography

[Appl 14a] Apple Inc. “Core Location Framework Reference”. https://developer.apple.com/library/ios/documentation/CoreLocation/Reference/CoreLocation_Framework, February 2014.

[Appl 14b] Apple Inc. “Core Motion Framework Reference”. https://developer.apple.com/library/ios/documentation/CoreMotion/Reference/CoreMotion_Reference, February 2014.

[Arth 12] C. Arth, A. Mulloni, and D. Schmalstieg. “Exploiting Sensors on Mobile Phones to Improve Wide-Area Localization”. In: *Proc. Int. Conference on Pattern Recognition (ICPR)*, 2012.

[Azum 97] R. T. Azuma *et al.* “A survey of augmented reality”. *Presence*, Vol. 6, No. 4, pp. 355–385, 1997.

[Bake 01] S. Baker and I. Matthews. “Equivalence and efficiency of image alignment algorithms”. In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[Bake 04] S. Baker and I. Matthews. “Lucas-Kanade 20 Years On: A Unifying Framework”. *Int. Journal of Computer Vision (IJCV)*, Vol. 56, No. 3, pp. 221–255, 2004.

[Bake 14] S. Baker. “Lucas-Kanade 20 Years On”. http://www.ri.cmu.edu/research_project_detail.html?project_id=515, February 2014.

[Bay 08] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. “Speeded-Up Robust Features (SURF)”. *Computer Vision Image Understanding*, Vol. 110, No. 3, pp. 346–359, 2008.

[Bazi 08] J. C. Bazin, I. Kweon, C. Demonceaux, and P. Vasseur. “Improvement of feature matching in catadioptric images using gyroscope data”. In: *Proc. Int. Conference on Pattern Recognition (ICPR)*, 2008.

[Beis 97] J. S. Beis and D. G. Lowe. “Shape indexing using approximate nearest-neighbour search in high-dimensional spaces”. In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 1997.

[Benh 04] S. Benhimane and E. Malis. “Real-time image-based tracking of planes using Efficient Second-order Minimization”. In: *Proc. Int. Conference on Intelligent Robots and Systems (IROS)*, 2004.

[Bles 09] G. Bleser and D. Stricker. "Advanced tracking through efficient image processing and visual-inertial sensor fusion". *Computers and Graphics (C&G)*, Vol. 33, No. 1, pp. 59–72, 2009.

[Box 58] G. E. P. Box and M. E. Muller. "A Note on the Generation of Random Normal Deviates". *The Annals of Mathematical Statistics*, Vol. 29, No. 2, pp. 610–611, 1958.

[Brow 05] M. Brown, R. Szeliski, and S. Winder. "Multi-image matching using multi-scale oriented patches". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 510–517, IEEE, 2005.

[Cai 11] H. Cai, K. Mikolajczyk, and J. Matas. "Learning linear discriminant projections for dimensionality reduction of image descriptors". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 33, No. 2, pp. 338–352, 2011.

[Calo 12] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. "BRIEF: Computing a Local Binary Descriptor Very Fast". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 34, No. 7, pp. 1281–1298, 2012.

[Chen 09] D. Chen, S. S. Tsai, V. Chandrasekhar, G. Takacs, J. Singh, and B. Girod. "Robust Image Retrieval using Multiview Scalable Vocabulary Trees". In: *Proc. VCIP*, 2009.

[Chen 11] D. M. Chen, G. Baatz, K. Koser, S. S. Tsai, R. Vedantham, T. Pylvanainen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, *et al.* "City-scale landmark identification on mobile devices". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Chit 05] L. Chittaro and S. Burigat. "Augmenting audio messages with visual directions in mobile guides: an evaluation of three approaches". In: *Proc. Mobile HCI*, 2005.

[Chum 05] O. Chum and J. Matas. "Matching with PROSAC - Progressive Sample Consensus". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[Criv 14] A. Crivellaro and V. Lepetit. "Robust 3D Tracking with Descriptor Fields". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[Dala 05] N. Dalal and B. Triggs. "Histograms of Oriented Gradients for Human Detection". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 886–893, 2005.

[Davi 07] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. "MonoSLAM: Real-time single camera SLAM". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 29, No. 6, pp. 1052–1067, 2007.

[Deme 95] D. F. Dementhon and L. S. Davis. "Model-based object pose in 25 lines of code". *Int. Journal of Computer Vision (IJCV)*, Vol. 15, No. 1-2, pp. 123–141, 1995.

[Drum 02] T. Drummond and R. Cipolla. "Real-time visual tracking of complex structures". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 24, No. 7, pp. 932–946, Jul 2002.

[Euro 14] European GNSS Agency. "Galileo Is The European Global Satellite-Based Navigation System". http://www.gsa.europa.eu/galileo-0, February 2014.

[Fein 97] S. Feiner, B. Macintyre, and T. Höllerer. "A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment". In: *Personal and Ubiquitous Computing*, 1997.

[Fisc 81] M. A. Fischler and R. C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". *Communications of the ACM*, Vol. 24, No. 6, pp. 381–395, June 1981.

[Frit 10] M. Fritz, K. Saenko, and T. Darrell. "Size Matters: Metric Visual Search Constraints from Monocular Metadata". In: *NIPS*, 2010.

[Gaug 11a] S. Gauglitz, T. Höllerer, and M. Turk. "Evaluation of Interest Point Detectors and Feature Descriptors for Visual Tracking". *Int. Journal of Computer Vision (IJCV)*, Vol. 94, pp. 335–360, 2011.

[Gaug 11b] S. Gauglitz, M. Turk, and T. Höllerer. "Improving Keypoint Orientation Assignment". In: *Proc. British Machine Vision Conference (BMVC)*, 2011.

[Gion 99] A. Gionis, P. Indyk, R. Motwani, *et al.* "Similarity search in high dimensions via hashing". In: *VLDB*, pp. 518–529, 1999.

[Grub 10] L. Gruber, S. Gauglitz, J. Ventura, S. Zollmann, M. Huber, M. Schlegel, G. Klinker, D. Schmalstieg, and T. Höllerer. "The City of Sights: Design, Construction, and Measurement of an Augmented Reality Stage Set". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2010.

[Grub 12] L. Gruber, T. Richter-Trummer, and D. Schmalstieg. "Real-Time Photometric Registration from Arbitrary Geometry". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2012.

[Gui 13] Z. Gui, Y. Wang, Y. Liu, and J. Chen. "Mobile Visual Recognition on Smartphones". *Journal of Sensors*, Vol. 2013, 2013.

[Harr 88] C. Harris and M. Stephens. "A combined corner and edge detector". In: *Proc. of Fourth Alvey Vision Conference*, 1988.

[Hart 04] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second Ed., 2004.

[Heik 09] M. Heikkilä, M. Pietikäinen, and C. Schmid. "Description of interest regions with local binary patterns". *Pattern Recognition*, Vol. 42, No. 3, pp. 425–436, 2009.

[Hein 12] J. Heinly, E. Dunn, and J.-M. Frahm. "Comparative Evaluation of Binary Features". In: *Proc. European Conference on Computer Vision (ECCV)*, 2012.

[Heym 07] S. Heymann, K. Müller, A. Smolic, B. Fröhlich, and T. Wiegand. "SIFT Implementation and Optimization for General-Purpose GPU". In: *WSCG*, 2007.

[Hint 11] S. Hinterstoisser, V. Lepetit, S. Benhimane, P. Fua, and N. Navab. "Learning Real-Time Perspective Patch Rectification". *Int. Journal of Computer Vision (IJCV)*, Vol. 91, No. 1, pp. 107–130, 2011.

[Hwan 09] M. Hwangbo, J.-S. Kim, and T. Kanade. "Inertial-aided KLT feature tracking for a moving camera". In: *Proc. Int. Conference on Intelligent Robots and Systems (IROS)*, 2009.

[Irsc 09] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. "From structure-from-motion point clouds to fast location recognition". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[Joll 05] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2005.

[Josh 10] N. Joshi, S. B. Kang, C. L. Zitnick, and R. Szeliski. "Image Deblurring Using Inertial Measurement Sensors". *ACM Trans. Graph.*, Vol. 29, No. 4, pp. 30:1–30:9, July 2010.

[Klei 04] G. Klein and T. Drummond. "Tightly integrated sensor fusion for robust visual tracking". *Image and Vision Computing*, Vol. 22, No. 10, pp. 769–776, September 2004.

[Klei 06] G. Klein. *Visual Tracking for Augmented Reality, PhD Thesis*. University of Cambridge, 2006.

[Klei 07] G. Klein and D. Murray. "Parallel tracking and mapping for small AR workspaces". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

[Klei 09] G. Klein and D. Murray. "Parallel Tracking and Mapping on a Camera Phone". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2009.

[Klei 10] G. Klein and D. W. Murray. "Simulating Low-Cost Cameras for Augmented Reality Compositing". *Trans. Visualization and Computer Graphics (TVCG)*, Vol. 16, No. 3, pp. 369–380, 2010.

[Knei 11] L. Kneip, D. Scaramuzza, and R. Siegwart. "A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Kurz 11a] D. Kurz and S. Benhimane. "Gravity-Aware Handheld Augmented Reality". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[Kurz 11b] D. Kurz and S. Benhimane. "Inertial sensor-aligned visual feature descriptors". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[Kurz 11c] D. Kurz, S. Lieberknecht, and S. Benhimane. "Benchmarking Inertial Sensor-aided Localization and Tracking Methods". In: *The 2nd Int. Workshop on AR/MR Registration, Tracking and Benchmarking*, 2011.

[Kurz 12a] D. Kurz and S. Benhimane. "Handheld Augmented Reality Involving Gravity Measurements". In: *Computers and Graphics (C&G)*, 2012.

[Kurz 12b] D. Kurz, T. Olszamowski, and S. Benhimane. "Representative Feature Descriptor Sets For Robust Handheld Camera Localization". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2012.

[Kurz 13] D. Kurz, P. G. Meier, A. Plopski, and G. Klinker. "An Outdoor Ground Truth Evaluation Dataset for Sensor-Aided Visual Handheld Camera Localization". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2013.

[Kurz 14] D. Kurz, P. G. Meier, A. Plopski, and G. Klinker. "Absolute Spatial Context-Aware Visual Feature Descriptors for Outdoor Handheld Camera Localization". In: *Proc. Int. Conference on Computer Vision Theory and Applications (VISAPP)*, 2014.

[Lee 11] W. Lee, Y. Park, V. Lepetit, and W. Woo. "Video-Based In Situ Tagging on Mobile Phones". *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 21, No. 10, pp. 1487–1496, 2011.

[Lepe 06] V. Lepetit and P. Fua. "Keypoint Recognition Using Randomized Trees". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 28, No. 9, pp. 1465–1479, 2006.

[Lepe 09] V. Lepetit, F. Moreno-Noguer, and P. Fua. "Epnp: An accurate o (n) solution to the pnp problem". *Int. Journal of Computer Vision (IJCV)*, Vol. 81, No. 2, pp. 155–166, 2009.

[Leut 11] S. Leutenegger, M. Chli, and R. Siegwart. "BRISK: Binary Robust Invariant Scalable Keypoints". In: *Proc. Int. Conference on Computer Vision (ICCV)*, 2011.

[Li 04] B. Li, L. Heng, G. H. Lee, and M. Pollefeys. "A 4-Point Algorithm for Relative Pose Estimation of a Calibrated Camera with a Known Relative Rotation Angle". In: *Proc. Int. Conference on Intelligent Robots and Systems (IROS)*, 2004.

[Li 10] Y. Li, N. Snavely, and D. P. Huttenlocher. "Location recognition using prioritized feature matching". In: *Proc. European Conference on Computer Vision (ECCV)*, 2010.

[Lieb 09] S. Lieberknecht, S. Benhimane, P. Meier, and N. Navab. "A Dataset and Evaluation Methodology for Template-based Tracking Algorithms". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2009.

[Lowe 04] D. G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". *Int. Journal of Computer Vision (IJCV)*, Vol. 60, No. 2, pp. 91–110, 2004.

[Luca 81] B. Lucas and T. Kanade. "An iterative image registration technique with application to stereo vision". In: *Int. Joint Conf. on Artificial Intelligence*, 1981.

[Luo 09] O. G. Luo Juan. "A Comparison of SIFT, PCA-SIFT and SURF". *Int. Journal of Image Processing (IJIP)*, Vol. 3, No. 4, pp. 143–152, 2009.

[Mair 10] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger. "Adaptive and Generic Corner Detection Based on the Accelerated Segment Test". In: *Proc. European Conference on Computer Vision (ECCV)*, 2010.

[Marq 63] D. W. Marquardt. "An algorithm for least-squares estimation of nonlinear parameters". *SIAM Journal of the Society for Industrial & Applied Mathematics*, Vol. 11, No. 2, pp. 431–441, 1963.

[Meta 09] Metaio. "A Dataset and Evaluation Methodology for Template-based Tracking Algorithms". http://www.metaio.com/research/a-dataset-and-evaluation-methodology-for-template-based-tracking-algorithms, November 2009.

[Meta 13] Metaio. "An Outdoor Ground Truth Evaluation Dataset for Sensor-Aided Visual Handheld Camera Localization". http://www.metaio.com/research/an-outdoor-ground-truth-evaluation-dataset-for-sensor-aided-visual-handheld-camera-localization, October 2013.

[Meta 14a] Metaio. "The AR Engine". http://www.metaio.com/products/ar-engine, January 2014.

[Meta 14b] Metaio. "Augmented Reality Software Development Kit". http://www.metaio.com/products/sdk, January 2014.

[Meta 14c] Metaio. "Junaio Augmented Reality Browser". http://www.junaio.com, January 2014.

[Miko 04] K. Mikolajczyk and C. Schmid. "Scale & Affine Invariant Interest Point Detectors". *Int. Journal of Computer Vision (IJCV)*, Vol. 60, pp. 63–86, 2004.

[Miko 05a] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. "A Comparison of Affine Region Detectors". *Int. Journal of Computer Vision (IJCV)*, Vol. 65, pp. 43–72, 2005.

[Miko 05b] K. Mikolajczyk and C. Schmid. "A Performance Evaluation of Local Descriptors". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 27, No. 10, pp. 1615–1630, Oct. 2005.

[Mora 80] H. P. Moravec. *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*. PhD thesis, Stanford University, Stanford, CA, USA, 1980. AAI8024717.

[More 09] J.-M. Morel and G. Yu. "ASIFT: A new framework for fully affine invariant image comparison". *SIAM Journal on Imaging Sciences*, Vol. 2, No. 2, pp. 438–469, 2009.

[Newc 11] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. "DTAM: Dense tracking and mapping in real-time". In: *Proc. Int. Conference on Computer Vision (ICCV)*, pp. 2320–2327, IEEE, 2011.

[Nutz 11] G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart. "Fusion of IMU and vision for absolute scale estimation in monocular SLAM". *Journal of Intelligent & Robotic Systems*, Vol. 61, No. 1-4, pp. 287–299, 2011.

[Oda 10] O. Oda and S. Feiner. "Rolling and shooting: two augmented reality games". In: *Proc. ACM Int. Conf. on Human Factors in Computing Systems (Extended Abstracts)*, 2010.

[Open 14] OpenStreetMap. "OpenStreetMap". http://www.openstreetmap.com, February 2014.

[Ozuy 07] M. Özuysal, P. Fua, and V. Lepetit. "Fast keypoint recognition in ten lines of code". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[Pini 07] P. Piniés, T. Lupton, S. Sukkarieh, and J. D. Tardós. "Inertial aiding of inverse depth SLAM using a monocular camera". In: *Proc. Int. Conference on Robotics and Automation*, 2007.

[Prim 14] PrimeSense. "Capri - 3D Sensors - PrimeSense". http://www.primesense.com/solutions/3d-sensor, February 2014.

[Pust 06] D. Pustka, M. Huber, M. Bauer, and G. Klinker. "Spatial relationship patterns: elements of reusable tracking and calibration systems". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2006.

[Reit 07] G. Reitmayr and T. W. Drummond. "Initialisation for Visual Tracking in Urban Environments". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

[Rosi 99] P. L. Rosin. "Measuring Corner Properties". *Computer Vision Image Understanding*, Vol. 73, No. 2, pp. 291–307, Feb. 1999.

[Rost 05] E. Rosten and T. Drummond. "Fusing points and lines for high performance tracking". In: *Proc. Int. Conference on Computer Vision (ICCV)*, 2005.

[Rost 06] E. Rosten and T. Drummond. "Machine learning for high-speed corner detection". In: *Proc. European Conference on Computer Vision (ECCV)*, May 2006.

[Rost 10] E. Rosten, R. Porter, and T. Drummond. "FASTER and better: A machine learning approach to corner detection". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 32, pp. 105–119, 2010.

[Rubl 11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. "ORB: An Efficient Alternative to SIFT or SURF". In: *Proc. Int. Conference on Computer Vision (ICCV)*, 2011.

[Sand 10] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. "Evaluating Color Descriptors for Object and Scene Recognition". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 32, No. 9, pp. 1582–1596, 2010.

[Schm 97] C. Schmid and R. Mohr. "Local grayvalue invariants for image retrieval". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 19, No. 5, pp. 530–535, May 1997.

[Shi 94] J. Shi and C. Tomasi. "Good features to track". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994.

[Shum 00] H. Shum and R. Szeliski. "Construction of panoramic image mosaics with global and local alignment". *Int. Journal of Computer Vision (IJCV)*, Vol. 36, No. 2, pp. 101–130, 2000.

[Smit 12] E. R. Smith, R. J. Radke, and C. V. Stewart. "Physical Scale Keypoints: Matching and Registration for Combined Intensity/Range Images". *Int. Journal of Computer Vision (IJCV)*, Vol. 97, No. 1, pp. 2–17, March 2012.

[Smit 95] S. M. Smith and J. M. Brady. "SUSAN - A New Approach to Low Level Image Processing". *Int. Journal of Computer Vision (IJCV)*, Vol. 23, pp. 45–78, 1995.

[Snav 08] N. Snavely, S. M. Seitz, and R. Szeliski. "Modeling the World from Internet Photo Collections". *Int. Journal of Computer Vision (IJCV)*, Vol. 80, No. 2, pp. 189–210, Nov. 2008.

[Steg 02] C. Steger. "Occlusion, Clutter, and Illumination Invariant Object Recognition". *Int. Archives of Photogrammetry and Remote Sensing*, Vol. 34, No. 3, pp. 345–350, 2002.

[Stre 08] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. "On benchmarking camera calibration and multi-view stereo for high resolution imagery". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[Stre 12] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua. "LDAHash: Improved matching with smaller descriptors". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 34, No. 1, pp. 66–78, 2012.

[Stur 12] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. "A Benchmark for the Evaluation of RGB-D SLAM Systems". In: *Proc. Int. Conference on Intelligent Robots and Systems (IROS)*, 2012.

[Tayl 09] S. Taylor, E. Rosten, and T. Drummond. "Robust feature matching in 2.3μs". In: *IEEE CVPR Workshop on Feature Detectors and Descriptors*, 2009.

[Tayl 11] S. Taylor and T. Drummond. "Binary Histogrammed Intensity Patches for Efficient and Robust Matching". *Int. Journal of Computer Vision (IJCV)*, Vol. 94, No. 2, pp. 241–265, 2011.

[Tuyt 08] T. Tuytelaars and K. Mikolajczyk. "Local Invariant Feature Detectors: A Survey". *Found. Trends. Comput. Graph. Vis.*, Vol. 3, No. 3, pp. 177–280, July 2008.

[Vale 12] R. E. G. Valenzuela, W. R. Schwartz, and H. Pedrini. "Dimensionality Reduction Through PCA over SIFT and SURF Descriptors". In: *Proc. Conference on Cybernetic Intelligent Systems (CIS)*, 2012.

[Veda 10] A. Vedaldi and B. Fulkerson. "VLFeat: An open and portable library of computer vision algorithms". http://www.vlfeat.org, October 2010.

[Vent 12] J. Ventura and T. Höllerer. "Wide-area scene mapping for mobile visual tracking". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2012.

[Wagn 03] D. Wagner and D. Schmalstieg. "First Steps Towards Handheld Augmented Reality". In: *Proc. Int. Symposium on Wearable Computers*, 2003.

[Wagn 05] D. Wagner, T. Pintaric, F. Ledermann, and D. Schmalstieg. "Towards massively multi-user augmented reality on handheld devices". In: *Pervasive Computing*, pp. 208–219, 2005.

[Wagn 08] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. "Pose tracking from natural features on mobile phones". In: *Proc. Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2008.

[Wagn 09] D. Wagner and D. Schmalstieg. "History and Future of Tracking for Mobile Phone Augmented Reality". In: *Proc. Int. Symposium on Ubiquitous Virtual Reality*, pp. 7–10, 2009.

[Wind 07] S. A. Winder and M. Brown. "Learning local image descriptors". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[Wu 08] C. Wu, B. Clipp, X. Li, J.-M. Frahm, and M. Pollefeys. "3D Model Matching with Viewpoint-Invariant Patches (VIP)". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[Wulf 07] O. Wulf, A. Nuchter, J. Hertzberg, and B. Wagner. "Ground truth evaluation of large urban 6D SLAM". In: *Proc. Int. Conference on Intelligent Robots and Systems (IROS)*, 2007.

[Yan 04] R. S. Yan Ke. "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors". In: *Proc. Int. Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[You 99] S. You, U. Neumann, and R. Azuma. "Hybrid inertial and vision tracking for augmented reality registration". In: *Proc. Int. Conference on Virtual Reality (VR)*, 1999.

[Zhan 00] Z. Zhang. "A Flexible New Technique for Camera Calibration". *Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 22, pp. 1330–1334, November 2000.