Dominik Gärner, BSc

Implementing a geospatial augmented reality application for 3D model texturing

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Geomatics Science

submitted to

Graz University of Technology

Supervisor

DI Dr.techn. Clemens Arth

Institute for Computer Graphics and Vision

Univ. Prof. DI Dr. techn. Dieter Schmalstieg

Graz, December 2015

Abstract

Camera images with corresponding poses can be used to provide textures for geospatial 3D models, e.g. buildings from a city model. The sensors of a smartphone deliver all necessary information (image, position, orientation) to texture existing, plain 3D models on the fly. Consequently a smartphone user easily produces a huge amount of texture data, which has to be filtered and analyzed for accuracy and quality. In an exemplary application of a mobile Augmented Reality browser, which is to be developed, the concepts of geospatial AR systems are to be explored. The development with open source libraries as well as discussing the challenges of combining a GNSS based AR application with geospatial data are to be in focus of this thesis.

Keywords: Augmented Reality, Structure from Motion, SLAM, Texture Mapping, Open Scene Graph, Point Cloud Library, 3D Reconstruction

Zusammenfassung

Zur Texturierung für Georeferenzierten 3D-Modelle, wie beispielsweise Gebäude von Stadtmodellen, sind digitale Kamerabilder mit zusätzlicher Positions- und Orientierungsinformation notwendig. Gängige Smartphones bieten sowohl die notwendigen Sensoren (Bild, Position, Orientierung) als auch die dafür notwendige Rechenleistung um 3D-Modelle in Echtzeit zu texturieren. Mit der Entwicklung eines mobilen Augmented Reality Browser, werden die Konzepte der raumbezogenen AR-Systeme durchleuchtet. Schwerpunkt dieser Arbeit liegt dabei sowohl in der Entwicklung eines mobile AR viewer mit Open-Source-Bibliotheken als auch die Untersuchung einer kombinierten GNSS-basierte AR-Anwendung mit Geodaten und dessen Herausforderungen.

Keywords: Augmented Reality, Structure from Motion, SLAM, Textureing, OpenSceneGraph, Point Cloud Library, 3D Rekonstruktion

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

....

Datum Unterschrift

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis.

....

Date Signature

Acknowledgements

I would like to express my gratitude to my supervisor Clemens Arth, for his support and patience, and Christoph Klug, for his help and contribution to this work. And also thanks to you, dear reader, for your interest in this master thesis.

Contents

Chapter 1

Introduction

1.1 Augmented Reality

Augmented Reality (AR) in general is the extension of a perceived reality with computer generated content. Many AR systems realize this concept by visualizing additional digital information or virtual objects in photos or videos. In the context of computer science, Azuma [13] defined that an Augmented Reality system must have the following three characteristics:

- Virtual information is as 2D or 3D graphical content superimposed on the real world
- The virtual objects are spatially registered in the 3D space
- The augmentation takes place interactively and in real time

Additionally, and especially within the scope of this thesis, we extend the characteristics of an AR system with the following:

- A geospatial AR system displays and processes geographical referenced data
- The majority of geospatial AR applications have a mobile characteristic and utilize mobile devices such as smartphones and head-mounted displays (HDM)

Another definition of Augmented Reality is used by Milgram et al. [38] by defining a reality-virtuality continuum. The whole spectrum of mixed reality (MR) reaches from the very extreme of a fully virtual reality (VR) to the other extreme of the real environment - the reality we live in. In a VR, a user is immersed in a completely synthetic, computer generated, world. In an AR, the real visual environment, often captured with a digital camera, functions as the basis and is augmented by the virtual. A close connection of AR lies in augmented virtuality (AV). Closer to virtuality than reality is the border to AR often blurred. The following Figure 1.1 illustrates the range of Mixed Reality and delimites AR to the other terms.

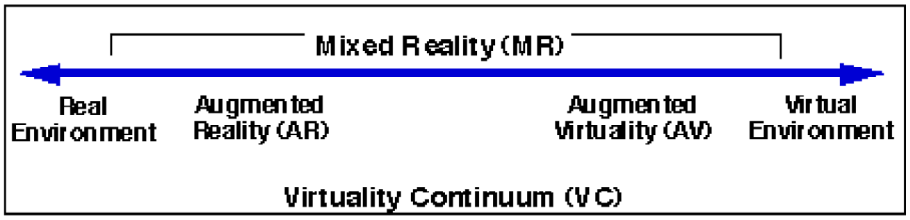


Figure 1.1: Simplified representation of a "virtuality continuum", where real environments are shown at one end of the continuum, and virtual environments at the opposite extremum. By Milgram et al. [38]

Current Trends in AR

In recent years, virtual and augmented reality systems and applications became very popular and drew public attention. Beginning with virtual reality (VR) some of the most noteworthy events of recent years are the publication of Oculus Rift in 2012 and the acquisition of Oculus VR by Facebook two years later. Oculus Rift (see Fig. 1.2(a)) is a VR head-mounted display with a initially 3 DoF, later on a 6 DoF head tracking. Quickly other hardware producers followed the VR trend by announcing HTC Vive (from HTC and Valve Corporation, Fig. 1.2(b)), Samsung Gear VR (by Samsung Electronics) or PlayStation VR (by Sony Computer Entertainment).

By neglecting the real surrounding, VR targets different markets and application scenarios. Augmented Reality interfaces clearly have advantages over conventional user interfaces by allowing a more natural interaction with virtual objects within the real world. AR came into public perception with the presentation of Google Glass (Fig. 1.2(c)) since 2012 which also formed the category of smart glasses or optical head-mounted displays (OHMD). Other well-known corporations showing their interest in AR, such as Microsoft by demonstrating their OHMD project named HoloLens (Fig. 1.2(d)). More interesting AR devices and applications are to be expected with the acquisitions of the German AR company Metaio by Apple Inc. and Qualcomm's Vuforia by Parametric Technology Corporation (PTC). Metaio and Vuforia where one of the major providers of AR applications through their software development kits (SDK). Magic Leap, known for their attention drawing marketing campaigns and financial support by Google, recently announces the future release of their own AR SDK platform [10].

(a) (b)

(c) (d)

Figure 1.2: (a): Oculus Rift Development Kit 2, (b): HTC Vive, (c): Google Glass, (d): Microsoft HoloLens

1.2 Motivation and Problem Statement

The main focus of this thesis lies on geospatial AR applications, as many upcoming developments are targeting this field. The reasons therefore are the rising demands in mobility in the private and industry sectors, the accompanying chances in the sectors of in-field (outdoor) workforce and the new possibilities of a more intuitive human-computer interaction (HCI) via AR. A mobile and geospatial Augmented Reality system meets all of these requirements.

The classic surveying discipline is traditionally associated with high costs and a very specific expertise. Recent breakthroughs in development and production brought cheap and powerful hardware in form of smartphones in widespread availability. Ubiquitous sensors and the continuous further development of algorithms in computer vision make data capturing and processing easy. The disciplines of geodesy and computer sciences therefore benefit from each other and the border between these two is often blurred. One logical next step for mobile geospatial information systems (GIS) is to combine traditional geospatial data and systems with the new technologies of Computer Vision and Augmented Reality.

Pose Tracking

The wide dissemination of mobile low-cost devices opens many opportunities, but also addresses various challenges. Many mobile AR systems fully rely on the information received from the devices sensors. The lack of quality of the built in low-cost sensors demands new ways of dealing with the given components and combining them with sophisticated algorithms. The key components for this kind of applications are the position and orientation, also referred as pose, sensed from the device. Tracking the pose of an object is often referred as six-degree-of-freedom (6 DoF) tracking, whereas the six parameters are: position vector $\mathbf{t} = [x, y, z]$ and orientation (roll, pitch and yaw angles) in 3D space. An AR application relies on the pose tracking in order to register a user's (or device's) pose in the real world to its digital content. By combining the available sensors in a sensor fusion approach, drawbacks and inaccuracies of single components can be compensated.

1.3 Contribution

In this thesis, we pursue the approach of refining sensor based (GNSS + IMU) camera poses by an additional SfM based reconstruction approach. Our intentions are to build an mobile AR system for smartphones and tablets, which uses the device's camera video stream + camera poses for a live texture mapping of urban 3D building models. We are using therefore the open source 3D rendering library OpenSceneGraph to create a C++ application and demonstrate the concept of a geospatial AR application on a Windows 8 tablet. The geo-referenced 3D building models are to be visualized in the camera's video stream, accordingly registrated to the user's position and orientation (pose), captured by the device's GNSS and IMU sensors. Over time captured images and poses can be globally adjusted and therefore can improve the results of a live texture mapping algorithm. Figure 1.3 provides a coneptual overview of the implemented AR system, with the attempt of further improvements of texture mapping results through improved camera poses. More details are to be discussed in the following sections.

1.4 Outline

In the following Chapter 2, we will present related work in the fields of Augmented Reality, Computer Vision and Geographical Information Systems (GIS). Chapter 3 addresses the theoretical background of all in this thesis relevant methods and

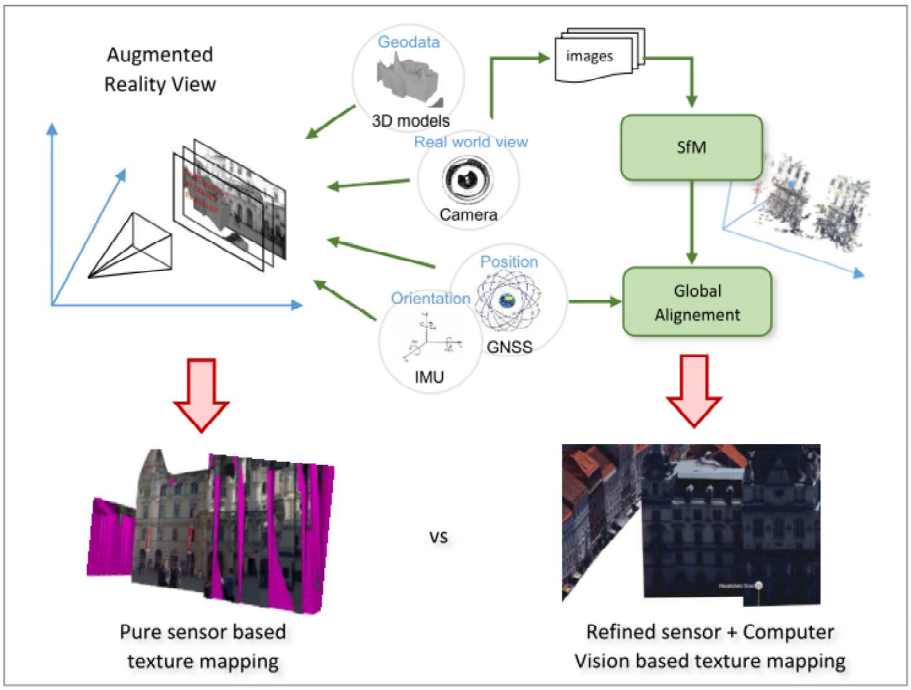


Figure 1.3: Conceptual overview as further discussed in Chapter 4

fields. Beginning with the hardware and sensor components, capturing all measurements, we continue with the Coordinate Systems which are involved in an AR system and proceed presenting the individual steps of a 3D reconstruction using a Structure-from-Motion (SfM) approach. In Chapter 4 we outline details about the implementations of a mobile AR application, the texturing of 3D models and an image based scene reconstruction. We present our experimental results in Chapter 5 as long with the used data sets for verifying our system. This regards the AR visualization results, as well as the 3D reconstruction and texture mapping results. The last Chapter 6 concludes with final remarks and an outlook for further improvements.

Chapter 2

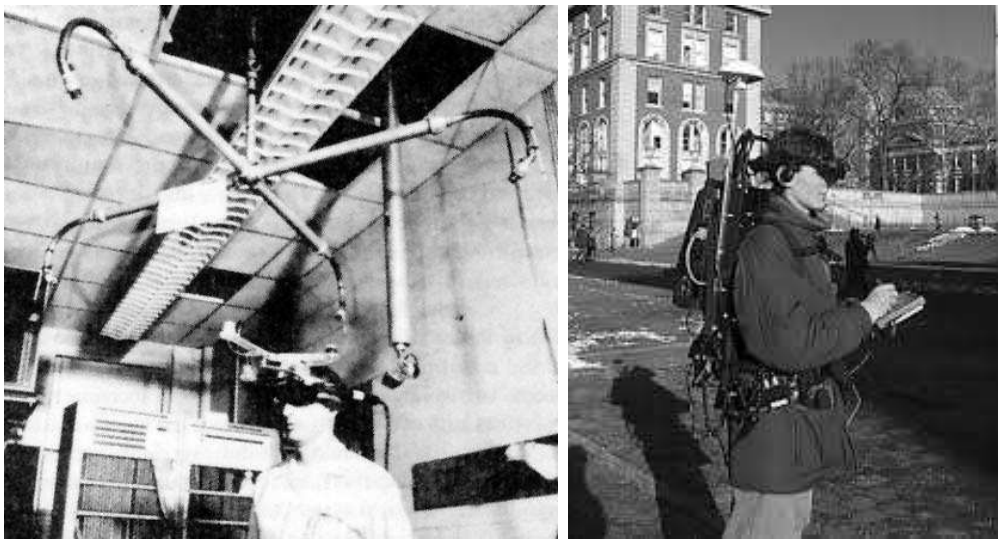
Related Work

This Chapter presents at first several examples of augmented reality (AR) research projects and applications from industries. It explains the concept of AR in practical use and show how modern mobile devices can enrich such applications. The next section introduces the basic concepts of geographical geoinformation systems (GIS) and tries to extend to the trend of mobile GIS. Mobile devices can locate themselves in outdoor environments using global navigation satellite systems (GNSS). However, GNSS positioning strongly depends among other factors on the environment and satellite visibility to the sky. Therefore new algorithms originated from the discipline of computer vision (CV) are currently the target of many investigations. Computer vision algorithms let consider a device's camera as an additional sensor which provides data for image based measurements. Theses image based measurements supplement traditional surveying techniques and bring the two disciplines of geodesy and computer vision closer together.

2.1 Augmented Reality Applications

The first augmented reality system was created by Ivan Sutherland in 1968, which was proposed as a new "Head-Mounted Three-Dimensional Display". It overlaid simple computer generated models on the real environment with a fixed apparatus, see Fig. 2.1(a) [55]. In 1996, Steven Feiner presented the Touring Machine as the first outdoor Mobile Augmented Reality System (MARS) [21]. The idea of this system (Fig. 2.1(b)) was to overlay additional information about the Columbia University Campus in an users view as the users walks around. It uses the following components:

- A see-through head-worn display with orientation tracker
- A backpack with a computer, differential GPS, and digital radio for web access
- A hand-held touchpad interface and stylus



(a) (b)

Figure 2.1: (a): "Head-Mounted Three-Dimensional Display" by Sutherland et al. [55], (b): Touring Machine, the first Mobile Augmented Reality System (MARS) by Feiner et al. [21]

Since the upcoming of modern smartphones, e.g. the iPhone in 2008, companies started launching AR software development toolkits (SDK), such as Wikitude or Metaio. These also started the distribution of AR applications on a wider user base. In 2008, Mobilizy launches The Wikitude World Browser (see Fig. 2.2(a)) as a location based AR app for Android devices. This application combines GPS and compass to overlay Wikipedia entries on the real-time camera view. A year later in 2009 SPRXmobile launches Layar (Fig. 2.2(b)), which extended the idea of Wikitude by also including additionial third party content via a free API as multiple layers. For a more comprehensive list of AR applications, the interested reader is referred to "The History of Mobile Augmented Reality" by Arth et al. [10].

One of the first connections between mobile AR and GIS took place in the project Vidente [52] and the follow-up project Smart Vidente [53] by Schall et al. The focus of this project was to access geospatial data directly in the field for civil engineering. In the Augmented Reality application of Smart Vidente a user can create, edit and update geospatial data representing real-world objects in the field of utilities. Some typical processes are on-site inspection and planning, data capture and as-built surveying of assets. The Smart Vidente prototype is a handheld device including a Windows Tablet PC, external GNSS receiver, an inertial measurement

(a) (b)

Figure 2.2: (a): Wikitude AR Browser [7], (b): Layar AR browser [5]

unit (IMU) and a camera. The user's pose is estimated using a sensor fusion approach based on GPS, IMU and the camera. Besides of accurate pose estimation, various visualization techniques were also part of investigation. Figure 2.3(a) shows an excerpt of methods on how to draw underground (and in reality hidden) assets, such as pipes and electrical cables, superimposed in the camera's image. The difficulty lies in the correct perception of depth and location of the underground pipes. Different methods, such as image based ghosting or geospatial cutaways by using virtual trenches, help for a better fusion of virtual and real image content. AR based on-site visualization techniques for georeferences environmental data in outdoor environments have been further investigated by Veas et al. [58] and Kruijff et al. [33]. The developed prototypes demonstrate AR systems specifically for environmental issues, e.g. analyzing and monitoring environmental processes in the field of hydrology.

The commercial AR application Augview continues the idea of Smart Vidente by making it possible to directly access the wide range of geospatial data of various geographical information systems. Augview therefore is the first commercial augmented reality mobile asset management application available for smartphones [1]. It functions as a full mobile GIS, on the one hand with a 2D map view and on the other hand by adding GIS object interaction methods in an additional AR view. Augview demonstrates the benefits of adding augmented reality visualization and interaction techniques to traditional mobile GIS applications and how industries can profit with these new technologies.

Arth et al. [11] incorporates georeferenced data from multiple crowd-sourced platforms, such as images from Flickr or Instagram and maps from OpenStreetMap, and

(a) (b)

Figure 2.3: (a): Different Visualization techniques for a better perception of depth of underground GIS assets in the project Smart Vidente [53], (b): The mobile AR GIS Augview on a tablet, visualizing underground pipes [1]

merges them in a mobile AR application. Another large-scale mobile augmented reality system in urban scenes is presented by Takacs et al. [56] where the problematic of accurate localization is addressed by geo-registered panoramas.

2.2 Surveying with Computer Vision

Surveying is traditionally associated with high capital and logistical costs, mostly due to expensive surveying equipment operated by trained professionals. This results in high costs of data collection for many applications in the fields of engineering, geodesy and earth sciences. Revolutionary improvements in computer science and the drastic price reductions and miniaturization of hardware emerges new possibilities. One method is termed Structure from Motion (SfM), which is a range imaging technique in the field of computer vision. A SfM algorithm recovers 3D structures from a sequence of 2D images, similar to stereoscopic photogrammetry systems. Reconstructed 3D maps can - next to surveying purposes - additionally be used for localization of the camera in 3D space. This is referred as Simultaneous Localization and Mapping (SLAM). A comprehensive SfM reconstruction is typically very time consuming and is therefore traditionally post processed and decoupled from the capturing process. SLAM is, in contrast, usually an online mapping technique, used e.g. in robotic applications for a real-time localization in unknown environ-

ments.

AR applications can benefit from these techniques in the the field of Computer Vision, as many researches have demonstrated. Oskiper et al. use a stereo visual odometry system to estimate relative camera motions and improve accuracies in pose estimation for augmented reality applications, by matching against a large-area feature database [41]. Their monocular camera approach [42] also integrates MEMS IMU and GPS sensor measurements for indoor and outdoor AR applications. Reitmayr and Drummond [48] used textured building models and presented a system for edge-based tracking. Ventura and Höllerer preparing wide-area environments for mobile visual tracking with handheld devices by generating point cloud models beforehand. Offline generation and live camera pose tracking is separated in a client/server system [60]. Arth et al. continiue this approach and describe how to efficiently create, handle and organize large-scale Structure-from-Motion reconstructions of urban environments. With a globally aligned reconstruction of the environment, live localization and tracking are used to visualize AR content [11].

2.2.1 Structure from Motion (SfM)

Traditional photogrammetric methods require the pose of the camera(s), or use control points with known 3D locations to reconstruct scenes from images. In contrast, the SfM method solves the camera pose and scene geometry simultaneously by matching features in multiple overlapping images as illustrated in Figure 2.6.

Different feature matching approaches are available. Lowe's SIFT (Scale Invariant Feature Transform) features [36] for instance extract local image feature descriptions based on local extrema of the image's Difference of Gaussians (DoG). SIFT features from different images than can be matched, whereas incorrect matches are filtered by an RANSAC (Random Sample Consensus) algorithm. If the camera poses are known, then it is possible to reconstruct 3D points of the observed object directly by triangulation. The triangulation is managed with intersecting rays from the camera's center through the feature points of one particular correspondence. Reconstructions then result in a Point Cloud which can further be used to generate solid 3D mesh models. Agarwal et al. [8] show this in an example of reconstructing Rome's Colosseum by using a collection of public photographs (see Figure 2.5). The used SLAM algorithm is based on a large number of public photographs from which matching feature points can be extracted. It also demonstrates the 3D reconstruction and modelling of buildings and cities with crowdsourced data and therefore as

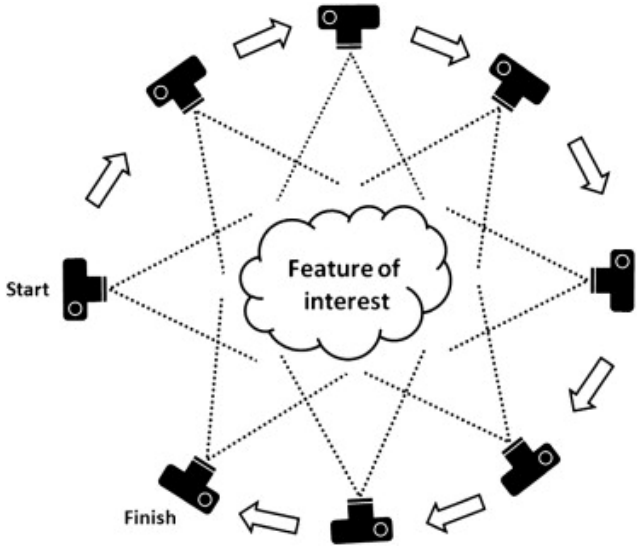


Figure 2.4: Structure-from-Motion (SfM): 3D reconstruction of an object of interest by using multiple overlapping images by Westoby et al.[62]

a cheap alternative to classic scanning techniques such as LIDAR (light detection and ranging) or Photogrammetry. Leberl et al. [35] demonstrated the reconstruction of point clouds at sub-pixel accuracy, based on SfM Photogrammetry and identified various advantages compared to LIDAR systems in this context. Irschara et al. [29] presented a fast location recognition technique based on SfM generated point clouds. Camera poses are estimated by searching for corresponding features within a tree-based index of feature points, extracted by SfM.

2.2.2 Simultanious Localization and Mapping (SLAM)

One logical continuation of the SfM approach is by not only creating a digital representation of their surroundings, but also use it as a map to localize yourself. This approach is called simultaneous localization and mapping (SLAM) and is a technique of both, constructing a map of an unknown environment while simultaneously tracking a user's location within it. Originated from the field of robotics, SLAM algorithms must often fulfill the requirement of working in real-time with limited available resources in regards to computational capacity. Therefore SLAM is often used to solve fast and sufficient, but approximate solutions prior to creating complete maps of an environment. Figure 2.6 illustrates the result of a SLAM application with a reconstructed 3D point cloud and the tracked trajectory of a moving camera in the scene.

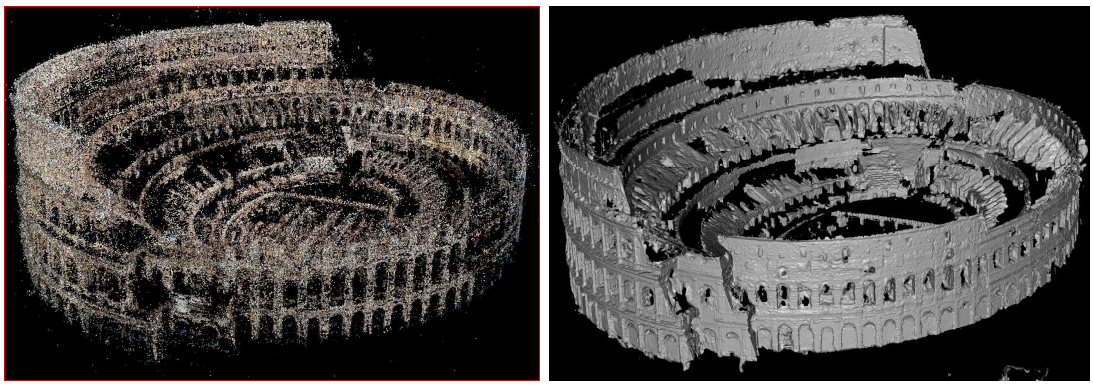


Figure 2.5: Reconstruction the Colosseum of Rome with SfM by Agarwal et al. [8] Left: Point Cloud of 819,242 points extracted from 2,097 images. Right: Interpolated mesh model

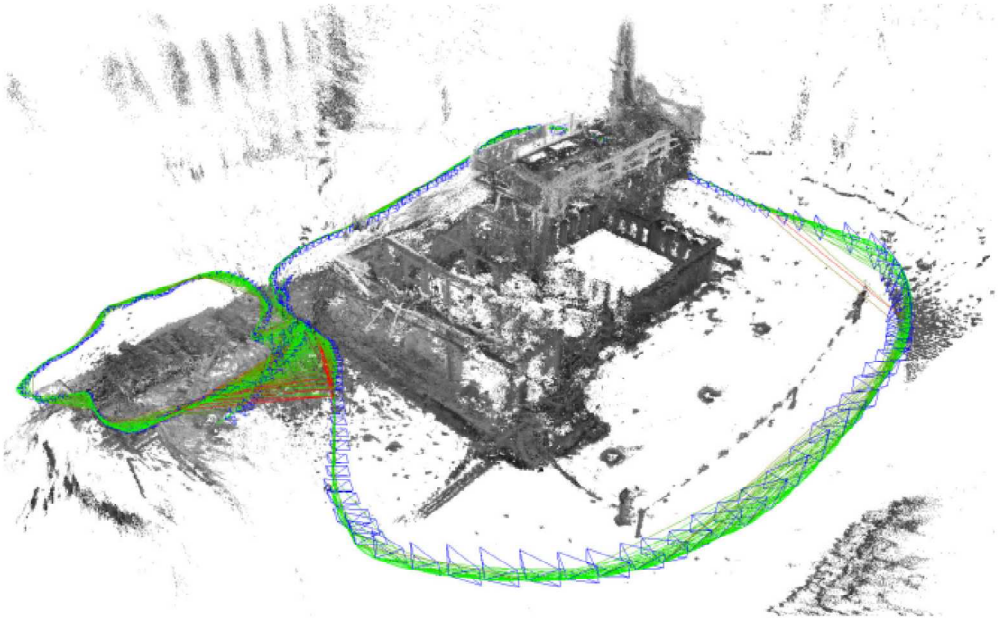


Figure 2.6: Examplemtary large-scale SLAM application: 3D reconstruction (mapping) of a point cloud and simultaneous localization of the camera's poses. Image by Engel et al.[20]

Two concurrent problems emerge with SLAM:

- A map is needed for localization, and
- an estimated pose is needed for creating a map.

Davison et al. [19] [18] proposed a SLAM system based on a single-camera and started with a natural feature detection and filtering approach. Klein and Murray demonstrated their SLAM system with a keyframe-based updating mechanism [31]. As Reitmayr et al. [49] in "Simultaneous Localization and Mapping for Augmented Reality" show, SLAM and sensor fusion in general can help to improve AR applications. An improved localization and tracking also improves an AR experience by stabilizing a camera's pose and stabilizing virtual content, such as annotations or geospatial 3D models, in the camera view.

Most of the initial SLAM algorithms are feature-based, which means 3D reconstruction takes place by using SIFT, SURF, or other feature descriptors. Feature descriptors strongly rely on the texture or structure of objects and their images. This directly influences the density or sparseness of reconstructed point clouds. Engel et al. [20] proposed with "LSD-SLAM: Large-Scale Direct Monocular SLAM" a direct and feature-less SLAM algorithm, which promises more dense reconstruction by using image intensities instead.

As recent research projects have shown, SLAM in the right combination with advanced algorithm have the potential for a precise and global positioning. As Ventura et al. [59] presented, a monocular SLAM system can provide a globally registered 6-DoF tracking and mapping in real-time. Whereas a SLAM algorithm on a mobile device helps the user for a fast initial tracking, a map on a server registers the users track in a global context.

2.3 Texturing 3D Models

Accurate digital 3D models of real objects and surfaces can nowadays easily be created by laser scanning, Photogrammetry or Computer Vision methods. Whereas the prior requirement is to describe and reconstruct the object's geometry, another demand goes towards photorealistic models. With the method of texture mapping, photos or image information can be mapped onto the digital surface of a 3D model. Texture mapping is targeted by many researchers, whereas in recent years the focus on automated approaches has been laid. Laycock et al. [34] proposed Automatic Techniques for Texture Mapping in Virtual Urban Environments, in which occluding

objects, such as a tree in front of a building's facade, can be identified and removed. This on the one hand can be achieved by using multiple images of one object of interest, on the other hand by detection and repetition of patterns in visible parts of one object.

Other research addresses the automatic mapping of uncalibrated pictures on dense 3D point clouds (Davelli et al. [17]) in situations where 3D scanners provide texture less point clouds. The images are often provided by multiple pictures which raises the difficulties of image mosaicing, contrast-, colour- and radiometric-adjustments and dynamic mapping, i.e. selecting the best texture from multiple textures (Rocchini et al. [50], Genc et al. [23], Previtali et al. [44]).

2.4 Geographical Information Systems

A Geographical Information Systems (GIS) manages geospatial data, visualizes them in form of maps and provides tools for geospatial manipulation and analyzation. GIS data is often managed in thematic layers and visualization in maps often comes with topographical or orthographic background maps. Geospatial information is kept by 2D, 2.5D or 3D geometries, dependent on the application and requirements.

The Open Geospatial Consortium (OGC) is the leading international non-profit organization for standardizations and interoperability in the context of GIS and geospatial data. One of the most influential OGC standards define the OpenGIS Web Services (OWS) such as Web Map and Web Feature Services (WMS and WFS). Amongst many more important definitions, the Augmented Reality Markup Language 2.0 (ARML 2.0) is one of the most recent contributions to AR and is currently in the OGC Candidate state. Implementations of ARML are amongst others supported by Wikitude, Junaio and Layar. The XML based language helps to describe AR scenes by defining Features as physical objects, VisualAssets as virtual objects and Anchors as the spatial relationship between them. The latter covers a wide range of tracker mechanisms, starting from simple geolocations to complex natural feature tracking techniques. A mobile AR GIS clearly benefits from an increased interoperability by supporting global standards.

Another Standard from the OGC is CityGML, a XML based information model and file format for representing whole city models in 3D. All topographic objects can be represented in different levels of detail. Amongst the geometrical description of objects, CityGML's focus is further more on defining semantic information

and describing the topography of objects. A similar approach is termed Building Information Modelling (BIM) and comprises of planning, creating and managing buildings with BIM enabled software. BIM starts it's applications from architecture, engineering and construction, over to maintenance and facility management. The concept includes plans, such as technical 2D CAD drawings, extends them with a geometrical third dimension, the time as a fourth dimension and various cost relevant information as another dimensions.

Mobile GIS

The demand of mobility brings geographical information systems to smaller and mobile devices such as smartphones, tablets, and handhelds. With the ability to access geospatial data and location-based services in the field, GIS improves its user value proposition for many industries. Workforce management can be optimized by using smart mobile application. However, the trend of working mobile also brings up some challenges. Cloud based services presupposes mobile internet connectivity, which is in practice often not or insufficient available. Other, more hardware specific difficulties come from limited power supply, poor visibility of smartphone displays in direct sunlight or sometimes limited functionality and simplified user interfaces due to touch based input methods. Augmented Reality could therefore revolutionize mobile field work by introducing a more intuitive user interface and geospatial data manipulation.

2.5 GNSS accuracy

One key component for a successful geospatial application is to achieve the positioning within the required accuracy, which differs for each application. In the context of mobile AR GIS applications, required positioning accuracies are typically around the metre range. However, GIS applications with surveying characteristics demand higher accuracies up to the centimetres level.

Low-cost consumer devices can't meet these requirements. Code-based one frequency GNSS measurements, as used by the built in GNSS receivers of nowadays smartphones in the consumer market, achieve a typical accuracy of < 13 metres (based on a 95 % confidential interval under normal conditions). Additional GNSS signal corrections may be applied to improve positioning results. Differential GNSS (also called DGPS from Differential Global Positioning System) achieves under good con-

ditions accuracies in the metre level and Real Time Kinematic (RTK) solutions up to 5 centimetre. [27, pp. 309–439].

RTK surveying grade receiver are typically available from 3.000 to 10.000 €and also need an open communication stream to a correction service. RTK correction services, based on a mobile internet connection are available as commercial services (e.g. in Austria APOS[1] or EPOSA[2]) and non-commercial but often regionally restricted services (e.g. the European EUREF-IP project, operated by the German Federal Agency for Cartography and Geodesy [2]). These services usually have a widespread network of GNSS reference stations and offer interpolated virtual references stations in the operating area. Alternatively other forms of RTK correction systems build up an own reference network with real reference stations. Such RTK corrections signals are often transmitted via radio signals, e.g. such as the Piksi RTK receivers from Swift Navigation [3].

Real GNSS measurements highly depend on the on-site conditions, especially regarding the number of "visible" satellites and signal quality due to disturbances. In practice the working conditions are manifold and signal shadowing and multipaths reduce the positioning accuracy drastically. This is a constant issue of various researches. One currently ongoing project, which addresses this issue is called PARADISE and is committed by the European Global Navigation Satellite Systems Agency (GSA). The name stands for Precise and Robust Navigation enabling Applications in Disturbed Signal Environments and their primary objective is to develop a solution that makes survey-grade GNSS-based positioning available in situations with bad signal conditions, such as indoors, urban canyons, forests etc. The project focuses on applications such as surveying in forests or construction works in urban canyons with augmented reality. PARADISE targets the development of a GNSS based navigation and attitude determination system with improved accuracy of positioning in a cost efficient way. A receiver will achieve this by improving GNSS signal processing with an inertial measurement unit (IMU) and combining it together with a precise-point-positioning (PPP) algorithm. [24]

[1] http://www.bev.gv.at/
[2] http://www.eposa.at/

Chapter 3

Hardware properties, Coordinate Systems and SfM

Mobile augmented reality applications are based on a variety of input parameters such as the devices position and orientation in the real world. Modern mobile devices, such as consumer grade smartphones and tablets, are already packed with a bunch of hardware sensors which enable the capability for AR applications. In order to improve the pose accuracy and quality we at first take a look in this Chapter at the available sensors, their measurements and limitations in accuracy. Bringing all the parameters together also means to take a closer look at the different coordinate systems which are involved in a mobile AR system and also answer the question of how to transform these input parameters between different coordinate systems. The following considerations regard the process of 3D modeling and texturing. Details about feature detection algorithms such as SIFT expose the capabilities and limitations of feature based structure from motion (SfM) and SLAM techniques.

3.1 Sensors and Measurements

In order to augment the reality, we have to capture some information from it beforehand. Most of the modern smartphones and tablets already capture them with different kind of physical sensors. Before you use there measurements as input parameters it's important to understand how they work, their capabilities and also their limitations in accuracy.

3.1.1 Mobile Devices and Hardware

A mobile augmented reality system is mostly based on the devices position and orientation, also called pose. As our intention is to work with geospatial AR applications, we are reliant on a absolute position, obtained by a global navigation satellite system (GNSS). The relative movement and orientation of the device can

be measured by combining a accelerometer and a gyroscope. Together with a third parameter, we can obtain an absolute orientation in the 3D space, here measured with a magnetic compass. The visual capturing of the reality takes place by a camera, which also builds the very basis of our augmented reality system and for further computer vision reconstruction methods.

3.1.1.1 Camera

The camera sensor of a smartphone captures the real world as single images or as video live stream. The video stream of the camera is used as a background of the AR application with virtual objects as the foreground. The camera sensor plays therefore a critical part of the AR concept as it represents the reality of Milgrams reality-virtuality continuum [38] (also see Chapter 1).

Camera calibration is a necessary step in 3D computer vision and photogrammetry in order to extract metric information from 2D images. Five intrinsic parameters are used to describe the geometric properties of a standard Pinhole Camera model, as illustrated in Figure 3.1. These parameters build the camera matrix $\mathbf{K}$ [25]:

$$\mathbf{K} = \begin{pmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.1}$$

- The focal length f as the distance between the pinhole and the image plane. The focal length is measured in pixels. In practice, the resulting image has non-square pixels and is described with the two focal length parameters f_x and f_y. The discrepancy between the ideal pinhole model with f and the practical description with f_x, f_y is due to a number of reasons, such as flaws in the digital camera sensor, unintentional camera lens distortions or errors in camera calibration.
- The axis skew s describes the skew coefficient between the x and the y axis. Their value is usually Zero or close to Zero.
- The principal point ($\mathbf{p}$) offset p_x, p_y describes the origin of the pixel coordinate system in the center of the image. In the pinhole model, the camera's principal axis is the line perpendicular to the image plane that passes through the pinhole.

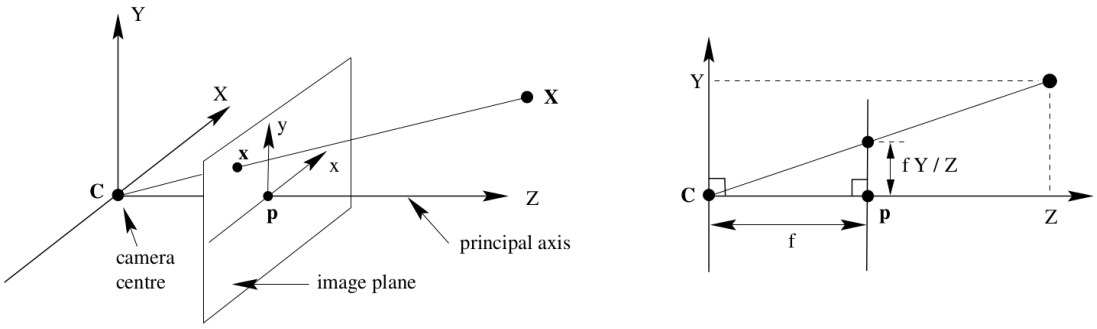


Figure 3.1: The Pinhole Camera model is used to mathematically describe the concept of the projective geometry of the digital camera. **c** is the camera centre (placed in a world coordinate system) and **p** the principal point. The image plane, here in front of the camera, placed in the distance *f*, the focal length, to the camera centre. Illustration by Hartley and Zisserman [25]

For a further projective geometry model, matrix **P**, called the "projection matrix", describes the projective camera model with an 3×4 matrix, where

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}] \tag{3.2}$$

K is the just described camera matrix, **R** a 3×3 rotation matrix and **t** a 3×1 translation vector which moves the camera centre **c** such that $\mathbf{t} = -\mathbf{Rc}$. This camera projection model is essential for both, visualization in augmented reality and reconstruction for SfM and SLAM. The projection matrix is defined up to a scale factor, which means linear mapping between 2D and 3D coordinates is ambiguous.

3.1.1.2 GNSS Positioning

Global navigation satellite system (GNSS) is a system of satellites that provide a global geospatial positioning for devices with a GNSS receiver. The two best known satellite systems are US NAVSTAR Global Positioning System (GPS) and the Russian equivalent GLONASS. The built in GNSS receivers in today's smartphones enable the user to determine his location on earth and in case of movement also the speed and heading.

To find a course geographic location, smartphones can also use the mobile network and the Wi-Fi antennas as 'triangulation sensors'. However, only a smart combi-

nation of all location sensors brings the most accurate and especially a quicker position fix. The usage of additional information, such as a course position, is also called Assisted GPS (A-GPS) and allows to shorten the time until a first GNSS position.

Accurate positioning is one of the most important issues for mobile AR applications. The GNSS position provides an estimate for the camera center $\mathbf{c}$, as described in the previous section 3.1.1.1. One of the drawbacks of GNSS signals is, that for a high positioning accuracy, they are only available in outdoor situations with a high open-sky visibility. Close buildings or vegetation can lead to shadowing or multipath effects and affect the calculated position.

3.1.1.3 Accelerometer and Gyroscope

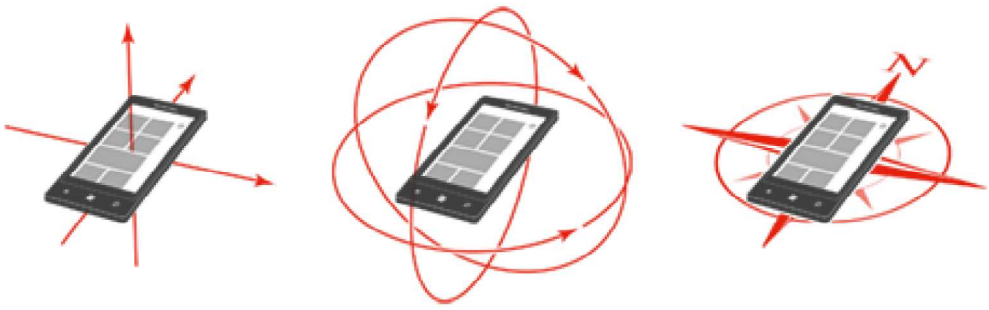
An accelerometer measures the forces or accelerations in one direction in the SI unit $\mathrm{m\,s^{-2}}$. With an typical arrangement of three accelerometers in different directions, one can sense accelerations in all three axes, as shown in Figure 3.2(a). This enables the device to find out two things: First the linear motions of the device and secondly the orientation relative to the earth. The reason therefore is, the gravitational force of the earth is omnipresent and with the vector direction it's possible to infer the orientation.

The gyroscope (or briefly gyro) measures the rotational velocity over the devices three axes with the SI unit $\mathrm{rad\,s^{-1}}$, Figure 3.2(b). The gyro therefore doesn't measure an absolute orientation, but the relative angular rotations with respect to inertial space. The gyro and also the accelerometer is usually realized as miniaturized MEMS-components (microelectromechanical systems).

Only after a combination of the accelerometer and gyroscope measurements it's possible to conclude to the full movement of a device, which is a composition of rotational and linear motions.

3.1.1.4 Magnetometer

A Magnetometer measures the magnetic field strength and direction. With the knowledge about the magnetic field in combination with an accelerometer - to get to an horizontal plane - the sensors function as a magnetic compass, Figure 3.2(c). Accelerometer and magnetometer give the device an estimate of the rotation $\mathbf{R}$ (see section 3.1.1.1).



(a) (b) (c)

Figure 3.2: (a): Linear 3-axes movements sensed by an accelerometer. (b): Relative angular rotations around 3-axes. (c): Magnetic compass by Daubmeier [16]

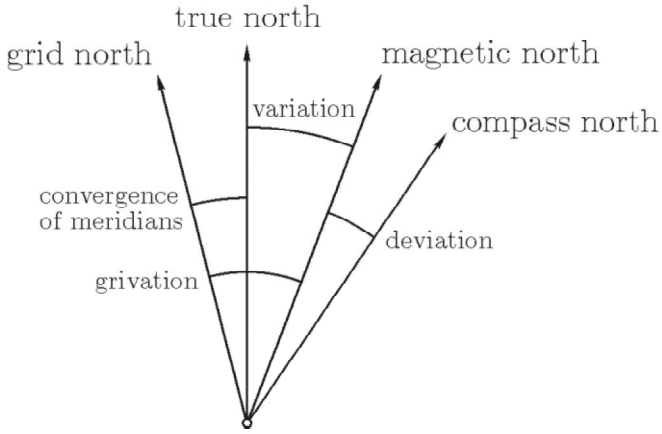
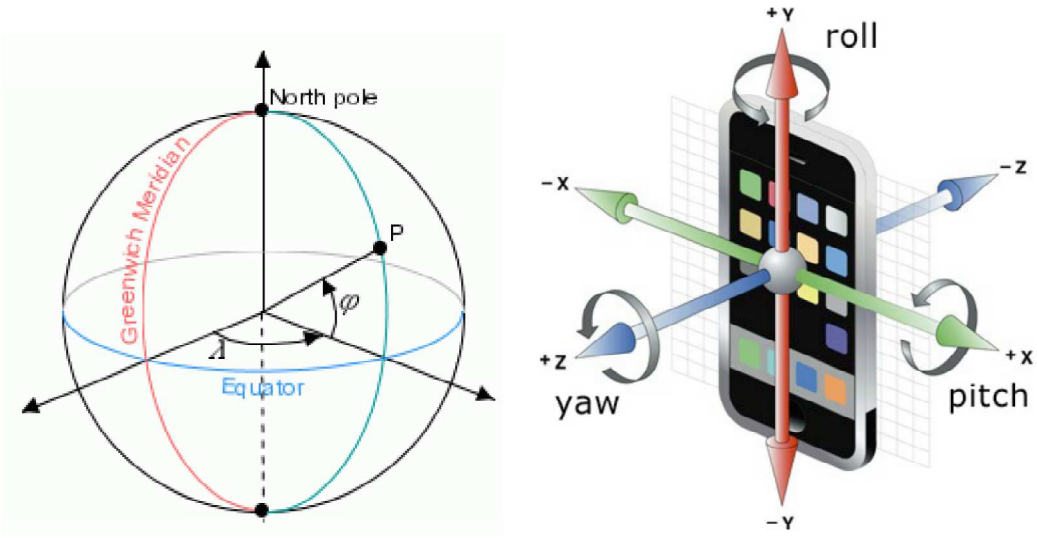


Figure 3.3: Transformation angles between the reference directions by Hofmann-Wellenhof et al. [26]

The measuring of the compass is usually disturbed by local magnetic fields, e.g. caused by close metallic objects or electric currents. The discrepancy between the compass north and the earth's magnetic north is called deviation, as depicted in Figure 3.3. The term true north is used for the direction to the earth's geographic North Pole, determined by the earth's rotation axis. The variation or magnetic declination is the difference between magnetic and true north. Declination varies over the location on the earth and changes over time. It describes the geomagnetic field of the earth's surface. Most software development kits already process the raw compass measurements and apply the variation angle according to a World Magnetic Model, which is updated regularly and valid for 5-year intervals. [6]



(a) (b)

Figure 3.4: (a): Geographic Coordinate System of a sphere by Peryt [43]. (b): Device orientation angles roll, pitch and yaw by Reitmayr [46]

3.1.1.5 Position and Orientation

For a full pose - position and orientation - the previously mentioned sensors have to be combined in terms of a sensor fusion approach. Nowadays mobile operating systems, namely Android, iOS and Windows, already process all these input information and already offer the software developer in a convenient way access to the output parameters position and orientation.

The location sensors usually deliver and regularly update the devices position as the geographic coordinates latitude and longitude (ϕ, λ) (see Figure 3.4(a)) according to the WGS84 coordinate reference system. A third parameter altitude defines the height above the WGS84 reference ellipsoid. Some API's also estimate a horizontal positioning accuracy of an 68% confidence interval [4].

The composite of the, in the previous section mentioned, orientation sensors (accelerometer, gyroscope, magnetometer) is used to determine the device's attitude in 3D space. Three rotation angles roll, pitch and yaw define a full Euler rotation in three-dimensional space. As in Figure 3.4(b) shown, the device axes x and y, with the rotation angles pitch and roll, lie in the device's display plane. Extra caution should be taken as some devices define the x-y plane in the user's portrait mode, whereas others in landscape mode. An additional ± 90 degree rotation around the z-axis may be applied. In practical use, the three parameterized rotation

angles are replaced by a 3-by-3 (respectively 4-by-4 in homogenous coordinates) rotation matrix (see Equation 3.3). In Computer Vision, and applied mathematical reference in general, this notation is preferred due to reasons of computational performance.

$$\mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{3.3}$$

In theory, with a known attitude and linear acceleration, it is possible to track the device's position in 3D space relative to some known starting position. This is called Inertial Navigation System (INS) and allows, for example, indoor navigation systems to continue tracking, where some starting position outside the building is known from a satellite navigation system. In practice, the sensors of current smartphones are not accurate enough and have a too strong drift after a short period of time to maintain an accurate INS based position. [16]

3.2 Coordinate Systems

In the context of outdoor AR, there are various components involved in generating or processing different forms of information. Each of them dealing with different coordinate systems. Starting with the user's pose, GNSS positioning takes place in a global reference system and a device reference system for inertial tracking sensors (Figure 3.5). Another component concerns the geospatial 3D models and data, often coming from GIS related coordinate systems, here referred as data reference frame. For the matter of computational processing for an AR application, it's crucial to work in one common reference frame and to transform different input streams from one into another reference system.

Global or World Coordinate System

The user's outdoor position is tracked by a GNSS sensor which operates within a global coordinate frame, such as WGS84 (World Geodetic System 1984). The geographic coordinates ϕ, λ and altitude describe a position on the system's reference ellipsoid, also referred as the datum. However, for geometrical computations a Cartesian coordinate system is preferred. Ellipsoidal coordinates such as WGS84 can also be expressed as three-dimensional Cartesian coordinates x, y and z. These are typ-

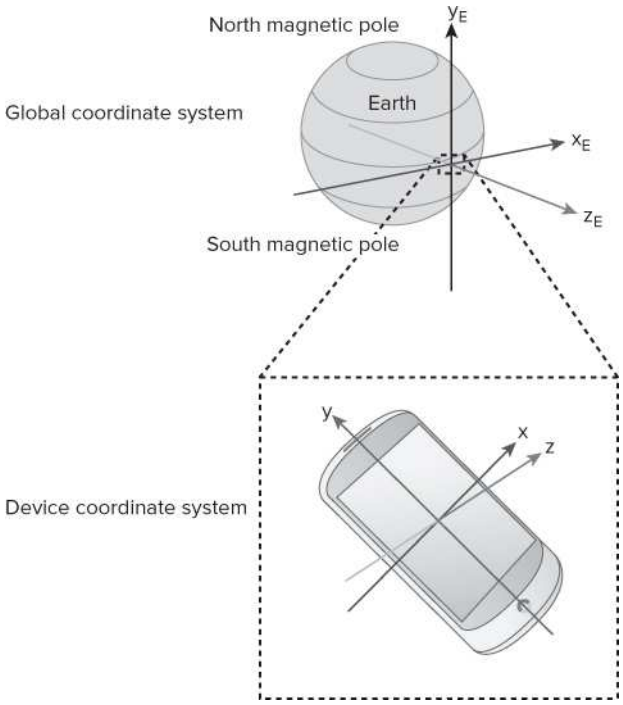


Figure 3.5: Coordinate reference systems of a mobile system by Milette et al. [37]

ically based on an earth-centered, earth-fixed (ECEF) system, which rotates with the Earth and has its origin at the Earth's centre. WGS84 coordinates are also often projected on a map onto two-dimensional coordinates. One well-known map projection, also used in this thesis, is the metric Universal Transverse Mercator (UTM) geographic coordinate system. Geographic information systems (GIS) usually manage data storage and coordinate transformations and projections between different reference systems.

Object or Data Coordinate System

Data models are typically represented in a local data reference frame. Due to computational reasons of computer graphics and 3D rendering, working with big numbers (of global coordinate systems) can lead to numerical problems in mathematical operations. One solution therefore is to transform geospatial data to a local horizontal (or tangential) coordinate system. Representing coordinates within a local East, North, and Up (ENU) axis system is for augmented reality applications more convenient. Figure 3.5 shows the ENU axis ($\mathbf{x_E}, \mathbf{y_E}, \mathbf{z_E}$), whereas the Up axis refers to the zenith of a map projection.

Camera or Device Coordinate System

The device's sensors work in their own coordinate system and must be combined into one common reference frame. Orientation (Figure 3.4(b)), position (Figure 3.4(a)) and the projection of the camera (Equation 3.2) must be fused for an augmented reality application.

Figure 3.6 illustrates the typical transformation pipeline of a 3D rendering engine. The computational process of coordinate transformation consists of multiplications of multiple matrices. Beginning with the Model Matrix, 3D vertices of an object are transformed from Object Space (the data's 3D model) into World Space. The View Matrix transforms the vertices from World Space to Camera Space. The scene is now built up as seen from the camera's or device's view. Another matrix multiplication by the Projection Matrix brings the once 3D scene onto the flat display in Screen Space. The following formula describes the relationship between the different transformations:

$$\mathbf{M_{MVP}} = \mathbf{M_{Projection}} \cdot \mathbf{M_{View}} \cdot \mathbf{M_{Model}} \tag{3.4}$$

The resulting **MVP** (Model-View-Projection) Matrix translates any 3D object $\mathbf{v_c}$ directly to its final 2D pixel location on the screen $\mathbf{v_s}$:

$$\mathbf{v_s} = \mathbf{M_{MVP}} \cdot \mathbf{v_c} \tag{3.5}$$

3.3 3D Reconstruction with SfM

Structure from Motion allows us to reconstruct scenes from image pairs. By comparing two or more images, it's possible to compute how much a pixel has moved between two images. In calibrated stereo systems this difference is called disparity. In a monocular camera system the depth information, or in other words, the distance from a camera can be reconstructed with SfM. The whole process consists of the following sequence of steps:

1. Feature Detection
2. Matching
3. (Iterative) Motion Estimation and Outlier Elimination
4. Reconstruction by Triangulation

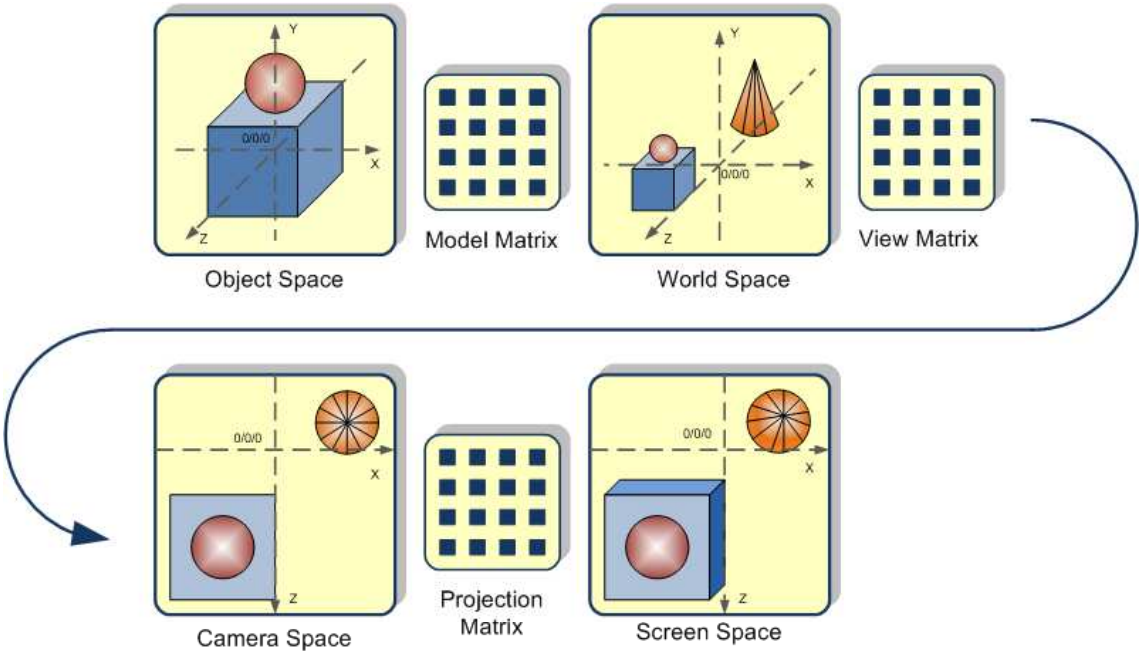


Figure 3.6: Graphical overview of the Transformation pipeline of a 3D rendering engine, such as OpenGL. By Rath [45]

5. Global Alignment of Reconstructed Poses

3.3.1 Feature Detection

As a first step it's necessary to identify the same objects or more precisely the corresponding pixels in both images. Many different algorithms are available to fulfill this task which basically can be divided in the two groups of *OpticalFlow* and *FeatureMatching* approaches. Optical flow algorithms, for instance the Least Squares Matching (LSM), are trying to track the pixels from one image to another and assume a certain range of movement, which is often realized with a moving window (cf. Kaufmann [30]). Hereby it's possible to use and compare every pixel, which allows a much denser reconstruction. This advantage also comes with the drawback of high computational costs.

Therefore most of the SfM methods used in real-time mobile computing, with limited processing resources, avail Feature Matching. In this thesis the SIFT algorithm is used and will be described in more detail, vicarious for its class. SIFT stands for Scale Invariant Feature Transform and uses local image features to determine correspondences. The extracted image features are - especially in comparison to other algorithms - invariant with respect to translation, rotation, scaling and changes in

lighting. This algorithm finds its application in image processing and computer vision, and was published in 1999 by David G. Lowe (University of British Columbia). The implementation of SIFT includes a smoothing of the image noise with a Gaussian filter and eventually a Difference-of-Gaussian (DoG). The next step is trying to find particularly prominent keypoints. The immediate neighboring pixels of a keypoint are considered pixel by pixel, to find local extremes - i.e. minima and maxima. Finally, the found keypoints are classified by a histogram of gradients of its local surrounding, which is called the keypoint descriptor. [36]

3.3.2 Matching

Finding matching keypoints in different images is a very time consuming process when using exact nearest neighbor matching algorithms. A drastic reduction in computation time can be achieved by a search for approximate results. The in this thesis used algorithm is FLANN, which stands for Fast Library for Approximate Nearest Neighbors. The FLANN library is optimized for fast approximate nearest neighbor search in large data sets with only a minor loss in accuracy. [39]

3.3.3 Motion Estimation

The relation between corresponding points of two (stereo) images from uncalibrated cameras is described by the fundamental matrix $\mathbf{F}$, such that

$$\mathbf{x}'^{T}\mathbf{F}\mathbf{x} = 0 \tag{3.6}$$

whereas $\mathbf{x}$ is the 3-dimensional image of a 3D point in homogenous coordinates and $\mathbf{x}'$ the according point in the other image. $\mathbf{F}$, a 3 x 3 matrix, thereby is the transformation matrix and also referred as the bifocal tensor. However, in a calibrated camera system with known intrinsic camera parameters $\mathbf{K}$, the geometrical relation is often described with the essential matrix $\mathbf{E}$, which is defined in the following equation:

$$\mathbf{E} = \mathbf{K}'^{T}\mathbf{F}\mathbf{K} \tag{3.7}$$

$\mathbf{K}$, the intrinsic parameters from the first camera and $\mathbf{K}$', the parameters from the second camera are equal ($\mathbf{K}' = \mathbf{K}$), when the same camera is used for both images. $\mathbf{E}$ can be used here to describe the camera movement from one image to another

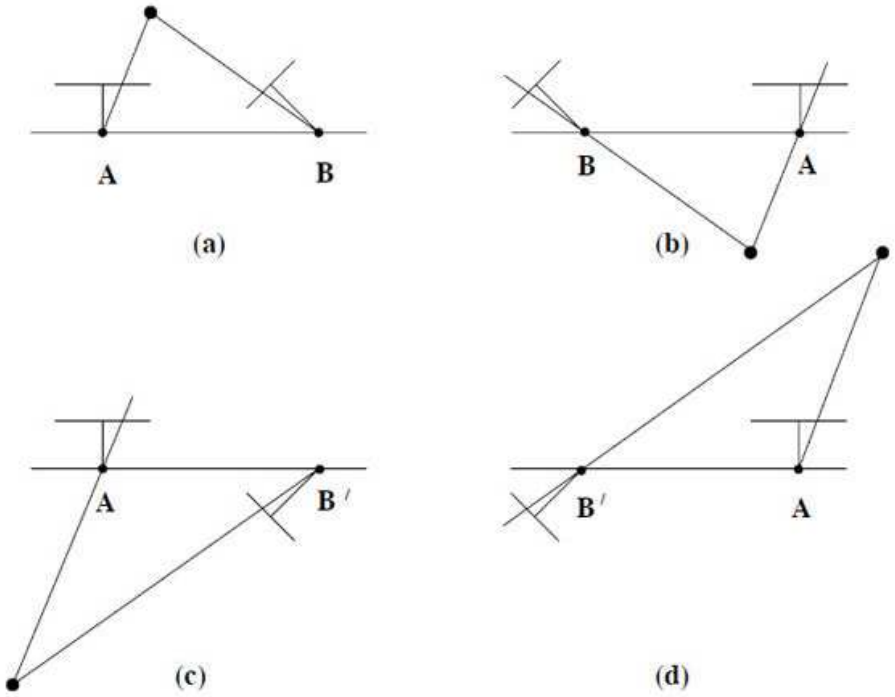


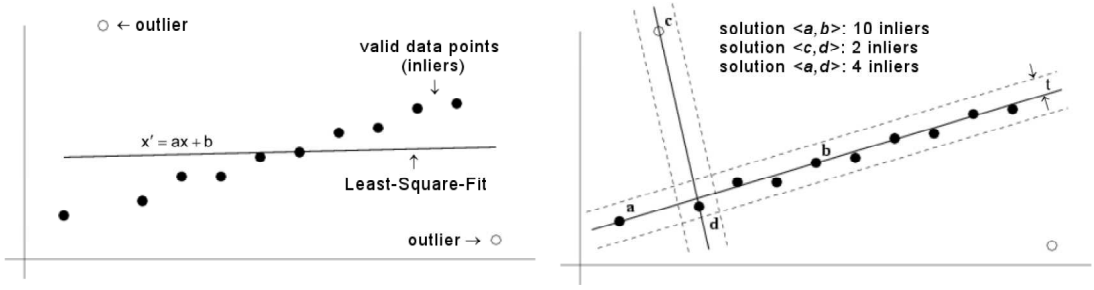
Figure 3.7: The four possible solutions for calibrated reconstruction from an essential matrix $\mathbf{E}$ by Hartley and Zisserman. [25]

and with the definition of $\mathbf{E} = [\mathbf{t}]_{\times}\mathbf{R}$ this movement consists of a rotation $\mathbf{R}$ and a translation $\mathbf{t}$. These components build a 5 degrees of freedom (DoF) system and can be used to provide a metric reconstruction up to a scaling ambiguity. To receive the components $\mathbf{R}$ and $\mathbf{t}$ from the essential matrix, $\mathbf{E}$ has to be factorized with a singular value decomposition (SVD). As Harley and Zisserman show in [25, pp. 258-260], the SVD leads to two possible choices for $\mathbf{R}$ and $\mathbf{t}$, giving combined the in Figure 3.7 illustrated four possible geometrical constellations. In practice the right solution can be determined by assuming that all points are placed in front of the two cameras (Fig. 3.7 (a)).

The decomposed parameters $\mathbf{R}$ and $\mathbf{t}$ define the relative motion, or more precisely the relative pose, from the first camera to the second. Any prior camera poses in a multiple camera system must considered and combined accordingly.

3.3.4 Outlier Elimination

In combination with RANSAC, it is possible to compare features of multiple images and find matches. RANSAC stands for Random Sample Consensus and is a model estimation method, which has an extremely high robustness against outliers. Espe-



(a) (b)

Figure 3.8: A robust line estimation. (a): Failing solution of a least square fit, due to the high influence of outliers. (b): Finding a good and quick estimate with RANSAC by comparing only a few random solutions. Thus is by counting the number of points within a threshold distance $\mathbf{t}$ (given by the dashed line). by Hartley et al. [25] and Kaufmann [30]

cially in contrast to the classical method of least squares RANSAC finds an estimate even with a very higher number of outliers in the data sets. As already included in the name, random samples are used in this algorithm to estimate a model. This is accomplished by not including all of the records for calculations, which gives RANSAC also a particular speed advantage over other algorithms.

In combination with SIFT, random feature points respectively their descriptors are compared and tested for compliance. While single outliers have a big impact on the solution in a least squares model calculation method (illustrated in Figure 3.8(a)), RANSAC eliminates those false solution candidates, containing high outliers (see Figure 3.8(b)). What remains is a solution of inliers. Figure 3.8 shows the schematic two-dimensional application of RANSAC at a point map. [25]

3.3.5 Reconstruction by Triangulation

Reconstructing 3D points of a scene takes place by intersecting the rays, coming from the two camera centres through the according points in the image plane (see Figure 3.9). The intersection point gives the location of the object (or pixel) in 3D space. This back projection is a triangulation in a geometrical sense.

In practice the two ray vectors hold inaccuracies and won't intersect in an exact point. To find a triangulated 3D position it's possible to set up and solve an overdetermined non-homogeneous linear equation system $\mathbf{Ax} = \mathbf{b}$, with $\mathbf{A}$ as the coefficients matrix, $\mathbf{b}$ the right hand sided coefficients vector and $\mathbf{x}$ the vector of the

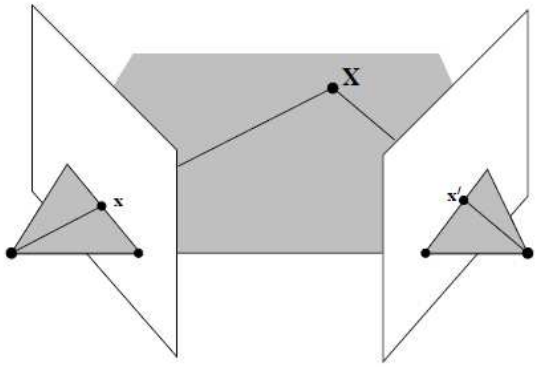


Figure 3.9: Triangulating a point $\mathbf{X}$ in 3d space by intersecting two projected rays from the two camera centres through the corresponding image points $\mathbf{x}$ and $\mathbf{x}'$. By Hartley et al. [25]

unknown variables. To avoid distortion and reduce reprojection errors of the reconstructed point, the linear triangulation can be performed iterative by weighting with the reprojection error. [14]

3.3.6 Global Alignment of Reconstructed Poses

A point cloud model, reconstructed from image based feature triangulation, always comes in its own relative coordinate system. Finding the relationship to another, global coordinate system is a classic task of photogrammetry and computer vision. By using pairs of measurements - here the reconstructed camera positions and their GNSS based position measurements - a transformation from one system to the other can be found and the relative point models therefore aligned to a global references system.

Some popular algorithms to find an absolute orientation are by Horn et al. [28], Arun et al. [12] and by Umeyama [57]. For a similarity transformation between two point clouds are at least three corresponding points in both coordinate systems necessary. More point correspondences can be solved in an overdetermined system with a least square estimation. With the similarity transformation the following three basic components are to be estimated: translation $\mathbf{t}$, rotation $\mathbf{R}$ and a scale s (similarity scale). The optima can be found by the minimum of the mean squared error $e^2(\mathbf{R}, \mathbf{t}, s)$:

$$e^2(\mathbf{R}, \mathbf{t}, s) = \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{y_i} - (s\mathbf{R}\mathbf{x_i} + \mathbf{t})||^2 \tag{3.8}$$

Chapter 4

Implementation

The practical part of this thesis includes multiple implementation steps targeting for the different computations, covered in the previous chapters. The individual subjects are structured for a better understanding in the following sections:

- **Mobile Augmented Reality Viewer**: Implementation of an Augmented Reality application with the underlying 3D rendering engine OpenSceneGraph. This AR viewer visualizes superimposed 3D models onto a camera view of a mobile device.
- **3D Scene Reconstruction and Refinement**: The reconstruction of point clouds and mesh models from a image sequence.
- **Texturing of 3D Models**: Algorithm for texturing blank 3D models with texture data coming from registrated images with known poses.

Figure 4.1 depicts the conceptual overview of these components. On the left side of Fig. 4.1 is shown the principle of a pure sensor base Augmented Reality view with all individual components of geodata, camera and GNSS/IMU sensors. Based on this alone, a texture mapping result will only be as accurate as the camera poses and is highly influenced by positioning or orientation inaccuracies. On the right side, the two sensors GNSS/IMU are also getting supported by a computer vision technique, considering the camera view as additional sensor. With a Structure-from-Motion approach, camera poses can be estimated by a visual, or image based, reconstruction. At first in a local reference system, but aligned with previous sensor measurements, brought to the global coordinate system. The texture mapping result benefits from this sensor fusion approach by a more accurate image-model registration.

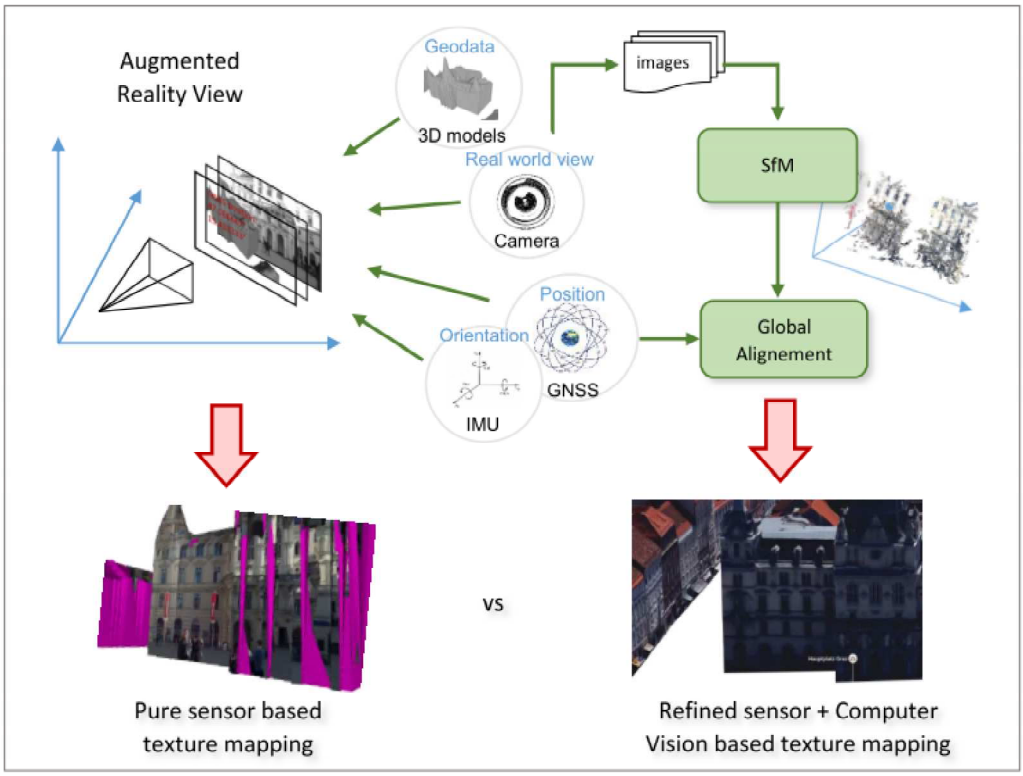


Figure 4.1: Conceptual overview

4.1 Mobile Augmented Reality Viewer

During this thesis a prototype of a mobile AR application has been created. This application has been used to discover the fundamental principles of Augmented Reality and to gain experience in the development of mobile applications. This section describes the architecture and basic concept of the implemented AR viewer application.

Listing 4.1 describes the used key components for the development and the environment under which the implementation took place.

4.1.1 OSG - OpenSceneGraph library

The mobile AR application is based on the rendering engine OpenSceneGraph (OSG). OSG is an open source rendering software development kit (SDK) for 3D graphics applications. Written in C++, it is a cross platform tool kit, which supports desktop applications as well as the application development for mobile platforms. OSG serves as a modular middleware framework, based on the rendering application interface (API) OpenGL.

Development Environment	
Development Platform:	*Windows 7 Professional N, 64bit*
IDE[3] & compiler:	*Microsoft Visual Studio 2013, 32bit*
Building tool	*CMake*
Mobile Testing Device	
Operating System:	*Windows 8.1, x86*
Device:	*Toshiba Encore (WT8-A-103)*
Key libraries	
3D rendering engine:	*OpenSceneGraph (OSG) v3.2.1 RC2*
Computer Vision processings:	*OpenCV v3.0.0 Beta*
Mathematical calculations:	*Eigen v3.2.1*
Coordinate transformation:	*GeographicLib v1.37*

Table 4.1: Overview about the mobile AR application development environment

As already included in the name, OSG implements the scene graph concept. A scene graph is a tree based collection of nodes, which in total describe the rendering of objects in a 3D scene. The concept of scene graphs is a common data structure concept and is applied in many modern 3D applications and tools, such as AutoCAD, Maya, VRML, X3D, etc.

As described by Wang et al. [61] and illustrated in Figure 4.2, the architecture of the tool kit is divided into the three main components of core libraries, "NodeKits" and plugins. The core libraries can be further divided into four main modules:

- osg: Summary of all the basic elements for building a scene graph
- OpenThreads: Provides the system wide used functionality for thread handling
- osgUtil: The rendering backend and scene graph traversal
- osgDB: I/O handling for a wide range of 3D data formats

The variety of available NodeKits already realize solutions for the most common problems in advanced 3D applications. As an open source library, OSG additionally allows the implementation and usage of diverse plugins. [40]

[3]IDE... Integrated Development Environment

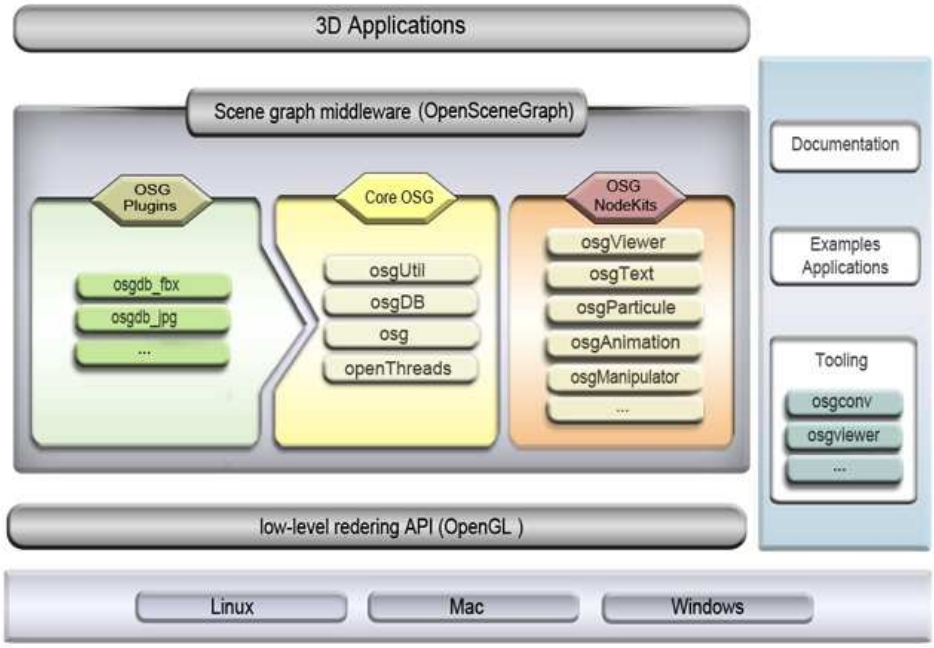


Figure 4.2: Overview of the OpenSceneGraph architecture from Wikipedia [9]

4.1.2 Application Architecture

This Section describes the architecture and basic concept of the implemented AR viewer application.

The Scene Graph

Since the main architecture and rendering related workload is implemented with the OpenSceneGraph library, the concept of the Augmented Reality application can be described in a scene graph related manner. The scene graph is built as a hierarchical and tree like relation of nodes. Figure 4.3 illustrates the implemented structure of the application. Any other AR application, based on OSG, could easily be inspired by this concept since it's universally applicable.

The rendering of an AR scene in this application is realized with an OSG viewer (NodeKit). The viewer takes care of the initialization of the whole rendering process and handles various GUI related processes. It is also holding the control of general user input and is responsible for the user related workflow. Any OSG application is holding a root node at its hierarchical top of the scene graph. The root node keeps three virtual cameras, which can be seen as the observing and recording elements of the 3D world. A camera's parameters dynamically define how and what is shown

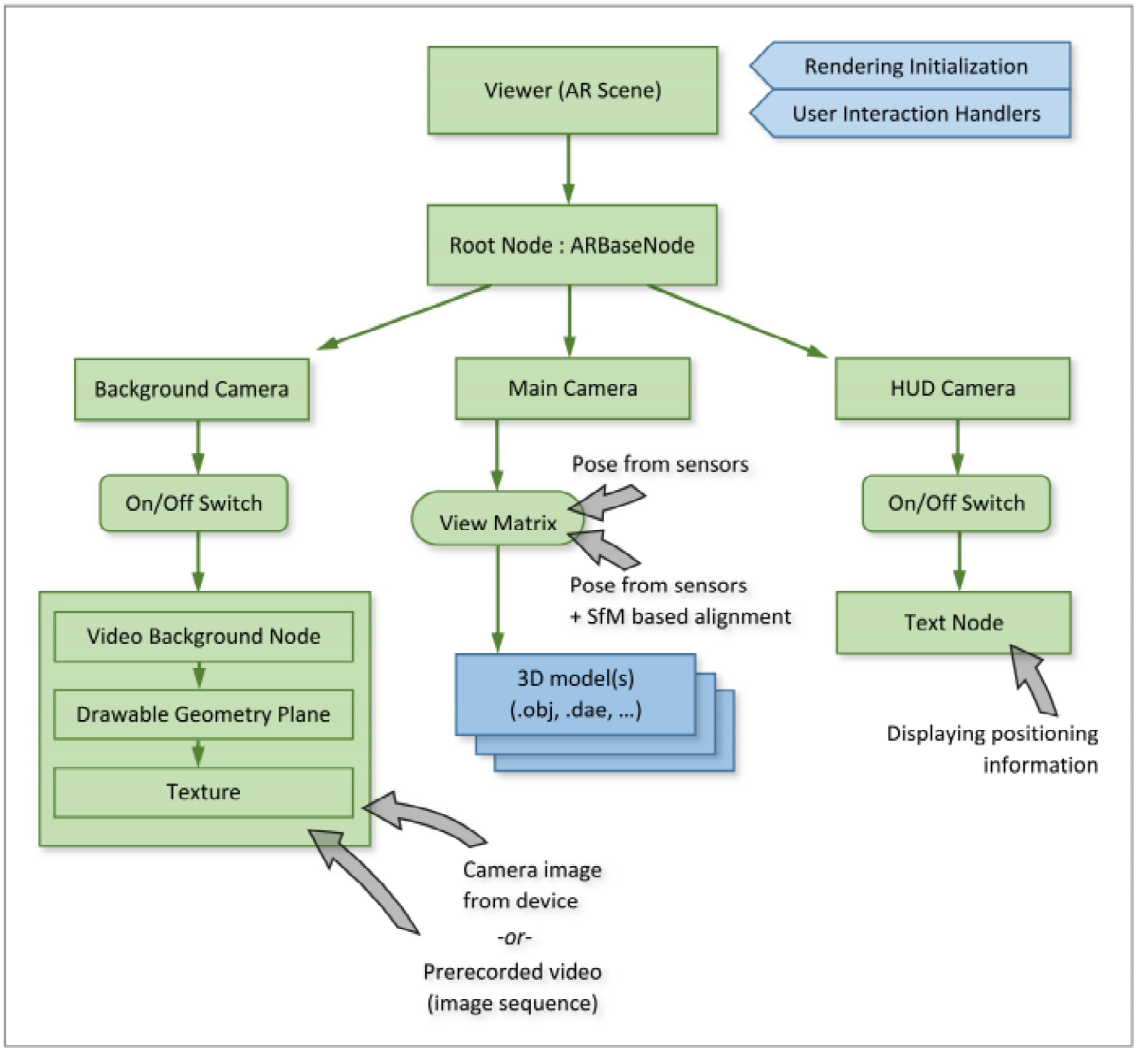


Figure 4.3: Illustrating the structure of the application's scene graph

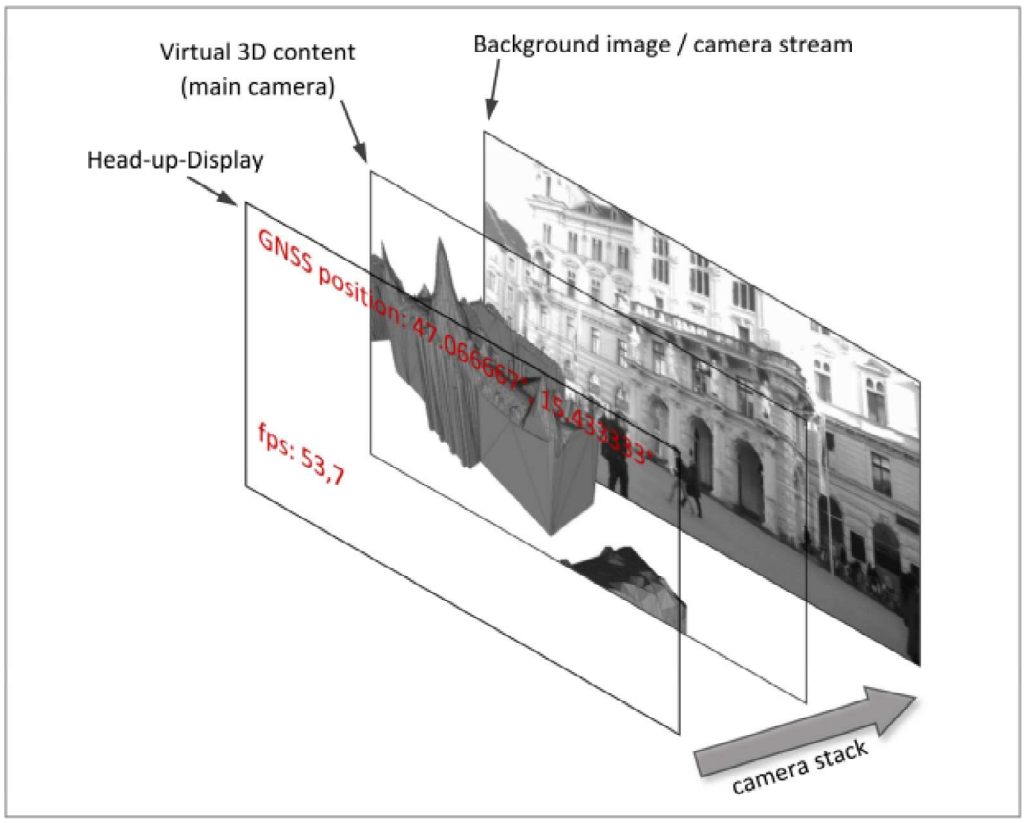


Figure 4.4: Illustrating the three cameras, stacked together to build a simple Augmented Reality scene: Background image stream, virtual 3D content and a Head-up-Display (HUD) overlay.

in the virtual scene and finally on the display. Most important parameters are the projection type and the movement of the camera, respectively the position and orientation. The three here used cameras can also be seen as three visible layers, stacked together to build an Augmented Reality scene. Figure 4.4 also shows this camera stack in another form with the three layers:

- **Head-up-Display (HUD) layer**: This is the overlaying HUD camera which is placed on top of all other rendering scenes. Its purpose is to display additional - in this application textual - information.
- **Virtual content**: The main camera shows the actual virtual 3D content. The camera view has to be updated frequently and in real time according to the pose of the device.
- **Background image stream**: This camera can be seen as a background layer or as a flat surface in the back, showing the camera's video stream on a canvas. All other content is superimposed onto this image.

OpenSceneGraph provides the class of Switch nodes, which are built in this application at two places. With this switches it's possible to change the visibility of the underlying child nodes to on and off. Therefore the user can show and hide the additional information from the HUD overlay. With the second switch it's possible to turn of the camera stream in the background. Doing so, the AR application would lose its visual context to the real world and change to a VR application.

The visualized 3D models are predefined through configuration parameters and for the sake of simplicity preloaded in the initialization step of the application. OSG supports many different file formats from default and even more through its provided plugin mechanism for reading and writing 2D and 3D files (osgDB library). In this demonstrative implementation mostly the file formats Wavefront OBJ and Collada DAE have been used.

The children nodes of the background camera in Figure 4.3 are grouped, since their main functionality is to create a background canvas for the camera feed. In more detail this can be divided into a general OSG node, a Drawable geometry which can be seen as plane in a fixed distance to the camera and their texture component. The texture of this surface can now be dynamically mapped to a sequence of images. The application is designed to use two different data sources as input stream. The first is the real camera of the mobile device and secondly a mockup image stream. With an prerecorded image sequence a fake camera view can be used. This is especially useful for testing and verification purposes of predefined, and in later chapters presented, data sets.

The Base Node

The camera's pose and the pose related functionality of the AR viewer application is extra highlighted in Figure 4.5. Analogous to the previously presented mock up functionality of the camera stream (Figure 4.3), the input information of the position and orientation can also be simulated. An abstracted Sensor component handles the two different input sources for position and orientation. This can either be the device's physical sensors or file based, prerecorded poses (the matching poses to the prerecorded images, as mentioned previously), coming from text files.

An absolute location in world coordinates, coming from GNSS sensors, is transformed to a relative position of the 3D scene. The known centre of the 3D models

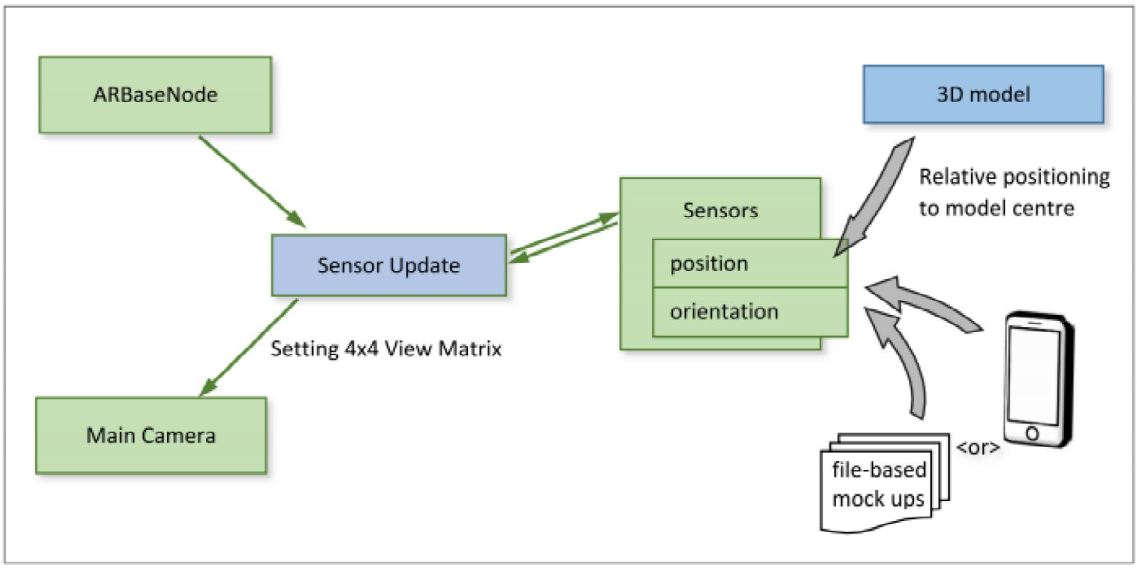


Figure 4.5: The functionality and interaction between the implemented classes "ARBaseNode" and "Sensor". The scope contains updating the camera pose respectively the user's position and orientation according to the chosen input stream (from device sensors or a prerecorded file base)

(in UTM zone 33N coordinates) must be subtracted from the incoming location information. The composed pose of the device is finally used to transform the 3D scene accordingly and to register the superimposed 3D models to the real environment.

4.1.3 Alternative and Related Software Libraries

The popularity of Augmented Reality led to a wide variety of available AR SDKs which provide to developers the tools and libraries for an easier development of AR applications. Reitmayr et al. published under [47] a comparison list of current AR related SDKs. Each seeking to provide functionalities and features for different uses and platforms. Some of the most popular SDKs are Qualcomm's Vuforia and Wikitude from Wikitude GmbH. This is due to the wide range of features and their support of mobile platforms.

Many AR applications also focus on the visualization of 3D content. That's why the underlying 3D rendering engine often plays an important role in the decision process of choosing an AR SDK. Unity is one of the most popular 3D gaming engine for mobile platforms. Its capabilities in rendering 3D scenes is well suited for VR and AR applications. Unity comes with its own development environment and supports

Figure 4.6: Changchang Wu [63] provides here a short instruction of the 3D reconstruction process with his GUI based software VisualSFM.

most of mobile and desktop platforms. The extension to AR is also supported by an AR plugin from Vuforia.

4.2 3D Scene Reconstruction and Refinement

Reconstruction of 3D scenes in form of 3D point clouds can be accomplished by already existing Structure-from-Motion (SFM) algorithms. Therefore multiple SfM software packages have been tested and evaluated in an experimental approach.

4.2.1 3D Reconstruction Tools

One promising and easy to use SfM software is VisualSFM: A Visual Structure from Motion System by Changchang Wu [63]. VisualSFM is a GUI application for a SfM based 3D reconstruction. Feature detection is based on SIFT and an additional bundle adjustment step is optimized for multi core CPU usage for a higher performance. The usually sparse 3D reconstruction can be densified with the external PMVS/CMVS (Patch-based-/Clustering Views for Multi-view Stereo) tool by Yasutaka Furukawa [22].

An alternative SfM application would be Bundler by Noah Snavely [54] or with its GUI based version of osm-bundler from Haiyang Xu [64]. Bundler is a SfM system for unordered image collections. Similar to VisualSFM, the main differences are from a user's point of view the different data formats and the command line based usability.

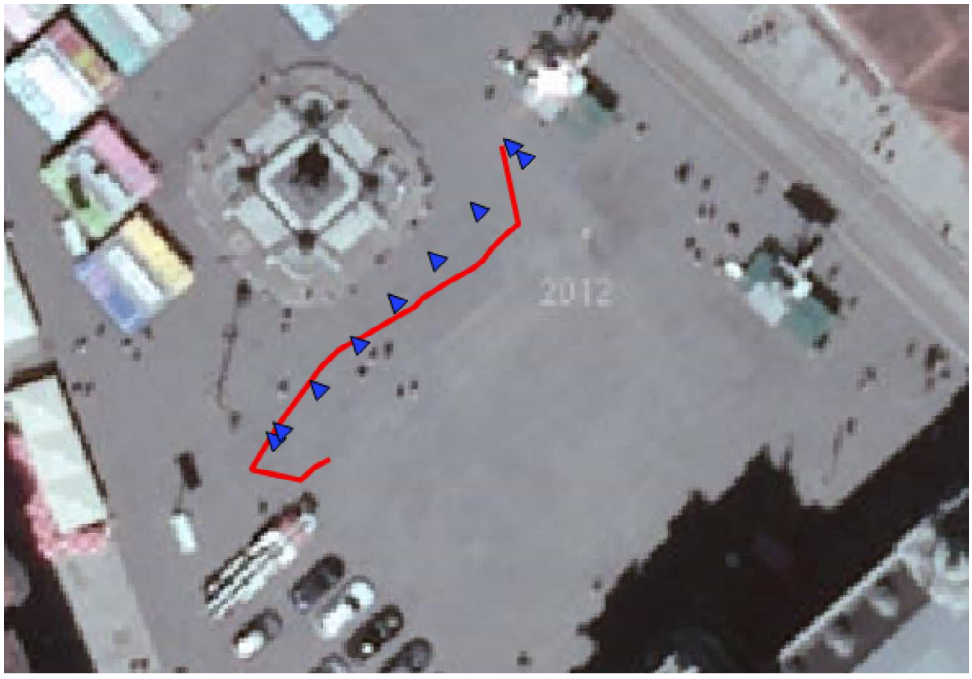


Figure 4.7: Camera poses (blue triangles) after an image based camera pose reconstruction and a global alignment step, based on the method of Horn [28]. The red line represents the sensor based track recording of a smartphones GNSS receiver. Background satellite image from www.basemap.at

4.2.2 Point Cloud Alignment

Mapping the reconstructed camera point coordinates to their GNSS based point coordinates can be done with the absolute orientation method of Horn [28]. Based on these corresponding local and global points the rotation, translation and scale are determined with an least squares estimation. These parameters of the similarity transform can now be applied to the point cloud, to transform the reconstruction to the global coordinate system.

Figure 4.7 depicts the adjusted camera poses (blue triangles) in comparison to a recorded GNSS positioning track (red line). The original track, recorded by the GNSS receiver of a smartphone, has a low positioning accuracy. The camera poses where at first reconstructed by a SfM algorithm and then globally aligned with the original GNSS positions, resulting in a much smoother track, which matches better to the real movement pattern of the recording user.

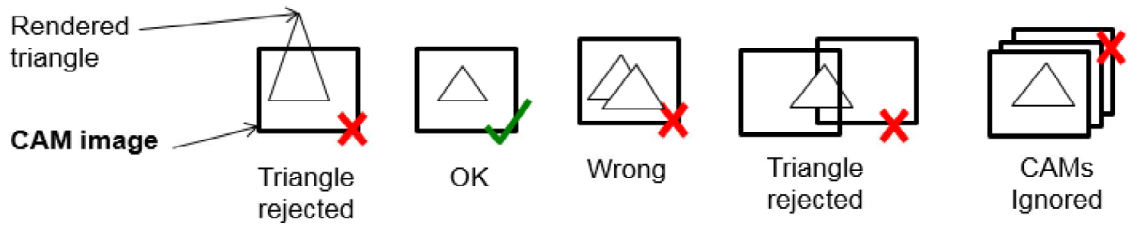


Figure 4.8: Restrictions in the mapping process of the used texture mapping algorithm by Klug [32]

4.3 Texturing 3D Models

The texture mapping of geospatial 3D models has been implemented by using and adapting some related work from the Institute of Computer Graphics and Vision (ICG). A reference implementation of a texture mapping pipeline has been modified to meet the requirements of this thesis. The input parameters for the texturing algorithms are basically the following:

- Georeferenced camera images (images + poses)
- Camera parameters
- 3D (urban) model

The texturing algorithm uses the Point Cloud Library (PCL) for mapping the texture from images on surfaces of the 3D model, visible from the camera's view. It's a basic implementation with the purpose of simply mapping the given images, whereas the camera selection of multiple cameras takes place by a first come, first serve principle. Overlapping images undergo no further correction. Only fully visible surface meshes are considered, hence partly out-of-view mesh triangles are being rejected. Figure 4.8 summarizes these processing limitations. [32]

4.3.1 Processing Time and Real-Time Considerations

Processing takes place offline and is currently not optimized for use in real-time. Calculation times rise linear with an increasing number of images and with the complexity of the 3D model, respectively the number of meshes.

The texturing algorithm has also been integrated in the mobile AR viewer. With that it's possible to take pictures on site and map them on the fly onto the 3D model's surfaces. Once again, processing time is increased on the mobile platform

due to its less powerful processing capabilities, compared to average desktop computers.

4.3.2 PCL - Point Cloud Library

The open source library for texture mapping used in this master thesis is Point Cloud Library (PCL). It is designed for processing and visualizing point cloud data, which plays an increasingly important role in the fields of computer vision, 3D GIS and autonomous robotics.

PCL is based on multiple other libraries, such as FLANN (Fast Library for Approximate Nearest Neighbors), Boost (for a fast data management), VTK (visualization library), just to mention some (cf. Rusu et al. [51]). As a C++ library, PCL is available for multiple platforms. As the experience within the scope of this thesis shows, raises the installation and development with PCL under MS Visual Studio for Windows many practical difficulties. The circumstance of its dependencies to various libraries increases the complexity tremendously under the stated development environment. This is amongst other things especially due to the fact of the very little existing and up to date documentation or online instructions.

Chapter 5

Evaluation and Experimental Results

This Chapter presents the results of development and experimental investigations. Beginning with exploring suitable data sets, we present the implemented AR viewer, evaluate texture mapping and discuss outcome and problems of all covered topics.

5.1 Testing Constellation and Data Sets

All used data sets were provided from the Institute for Computer Graphics and Vision, Technical University of Graz, and took their origin from previous research projects. There have been used the following two data sets, as well for offline testing and verification purposes as for the on site evaluation in context of Augmented Reality:

- Data set 1: Inffeld, TU Graz
- Data set 2: Main Square of the city of Graz

5.1.1 Data set 1: Inffeld, TU Graz

Data set 1 represents the buildings of the Technical University of Graz around the Inffeldgasse, see Figure 5.1.

The 3D models are non-solid models and represent for the sake of simplicity building facades only. The surfaces are simplified to as little as possible triangular planes. Bigger surfaces are teared down into smaller faces. This circumstance is due to some limitations in the chosen texturing algorithm and is explained in further detail in Section 5.4. The coordinative center of the 3D model is used as ground truth.

As reference data for further processing, captured camera shots of the buildings (Fig. 5.6) with pose information are available. Data has been recorded by walk-

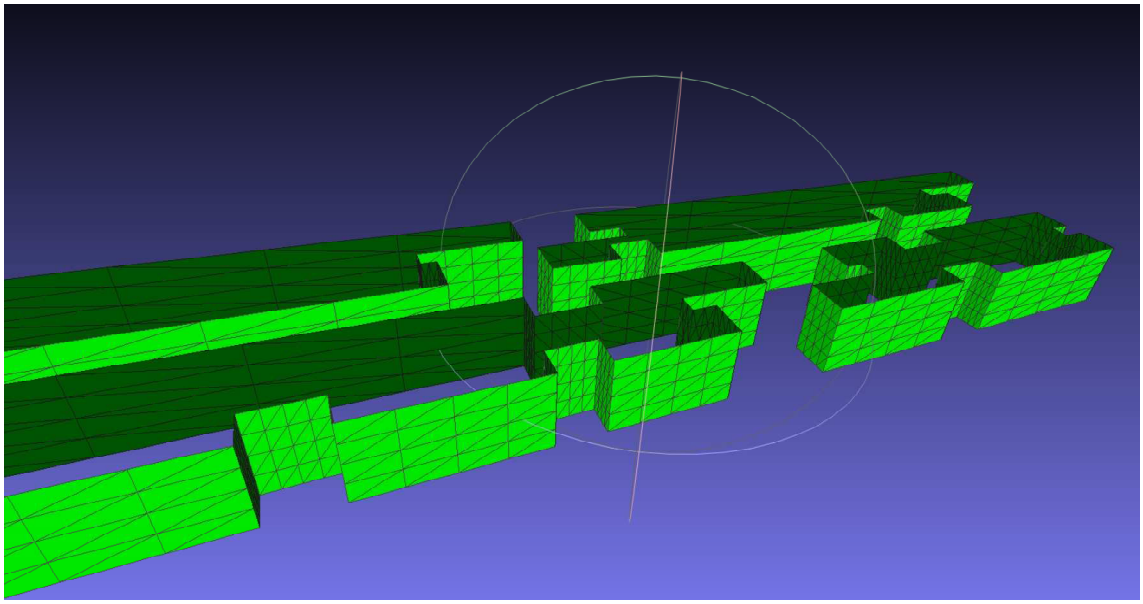


Figure 5.1: Data set 1: Simple 3D model of the ”Inffeld Campus” buildings of TU Graz

ing around the whole building complex. The camera pictures have been taken with five cameras on a backpack, evenly aligned each facing a different direction and taking pictures of a 360 degree surrounding. The sample rate is one second. The cameras are calibrated beforehand and all camera parameters, as long with camera poses are stored in a SQLite database. The corresponding pose information has already been post processed and improved in a global bundle adjustment step. For further calculations, these poses have been assumed as the ground truth information.

Figure 5.2: Some exemplary images from data set 1 (Inffeld, TU Graz)

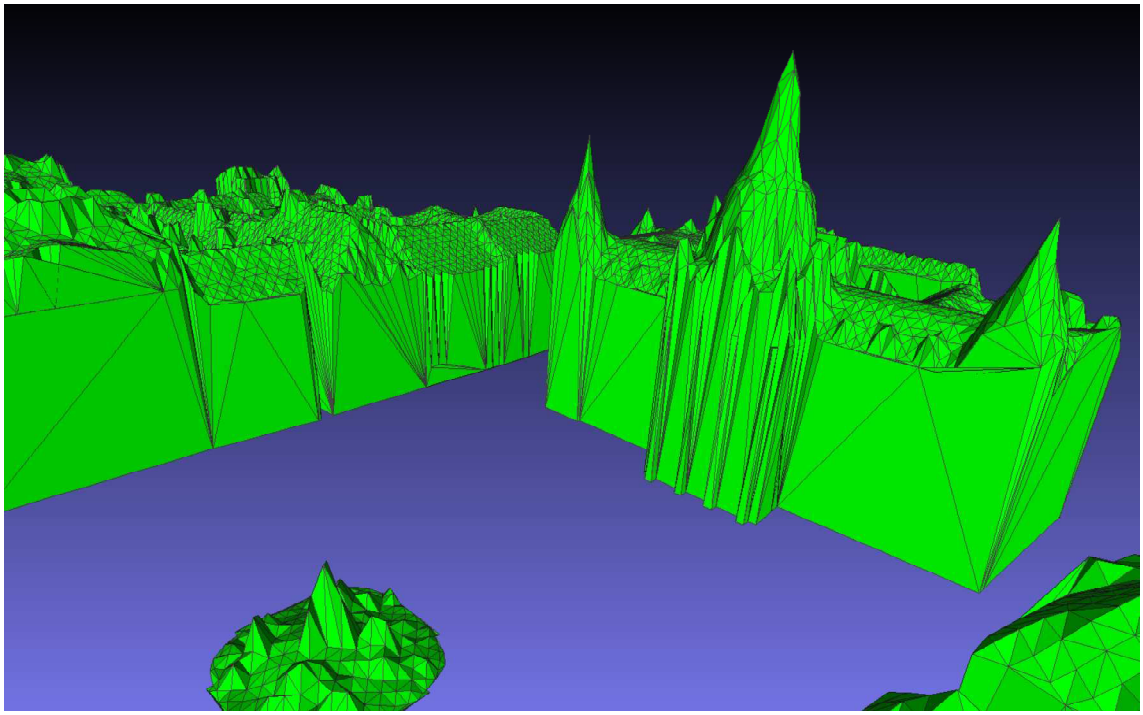


Figure 5.3: Data set 2: 3D building models for the main square in Graz, AT

5.1.2 Data set 2: Main Square of the city of Graz

Data set 2 is located at the main square of the city of Graz, Austria, including all surrounding buildings as well as the city hall. The 3D models have been created as a conglomerate of airborne laser scanning data and 2D cadastral information. The building's outline geometries have been extruded to match the laser scanning data in height. The point cloud from the laser scanning has been triangulated and builds the roof surfaces. The intersected roof surface with the extruded cadastre polygons builds up the 3D model as shown in the snapshot in Figure 5.3.

The sensor recordings of this data set constitutes a real case scenario of a mobile AR application. All recordings originate from a standard consumer grade smartphone and haven't been filtered or modified. Poses from the sensor readings are logged for every image and include full inaccuracies as from a real use case. For each timestamp of the pose sensor readings, a snapshot from the recorded footage has been extracted. Figure 5.4 gives an overview of the recorded track. The video has been taken while moving on the path (shown as a red line), filming the city hall south-east from the location, here in the lower-right corner of Figure 5.4.

The following listing is an excerpt of the recorded sensor information for the im-

Figure 5.4: Data set 2: Captured track of the sample data showed in a Google Maps areal map. The red line represents the track. The device's camera is pointing south-east to the town hall.

ages. It includes data from accelerometer (A), gyroscope (G), magnetometer (M), GNSS location (L) in UTM 34N coordinates, and the compound orientation as 9 components of a 3-by-3 rotation matrix (R).

```
1 A 0.008639 -0.001956 0.009080
2 G 0.088457 -0.021746 -0.026762
3 M -0.986071 -0.015622 0.165587
4 L 533277.384424 5213152.782051 353.823547
5 R 0.129105 -0.104857 -0.986071 -0.555765 -0.831193 0.015622 -0.821254
6   0.546007 -0.165587
```

5.2 Visualization Results - Mobile Augmented Reality Viewer

During this thesis a prototype of a mobile AR application has been created. This application has been used to discover the fundamental principles of Augmented Reality and to gain experience in the development of mobile applications.

Details about the development environment and the application's architecture have been discussed in the previous Chapter, in Section 4.1. This Section discusses the usability and experience with AR.

5.2.1 User Interface (UI)

The implemented interface for user interactions has been designed for an easy usage. Being aware of difficulties in user handling with mobile devices and touch screens, the user actions can be activated by a single touch (respectively click in the application). The AR viewer allows basically two main user functionalities:

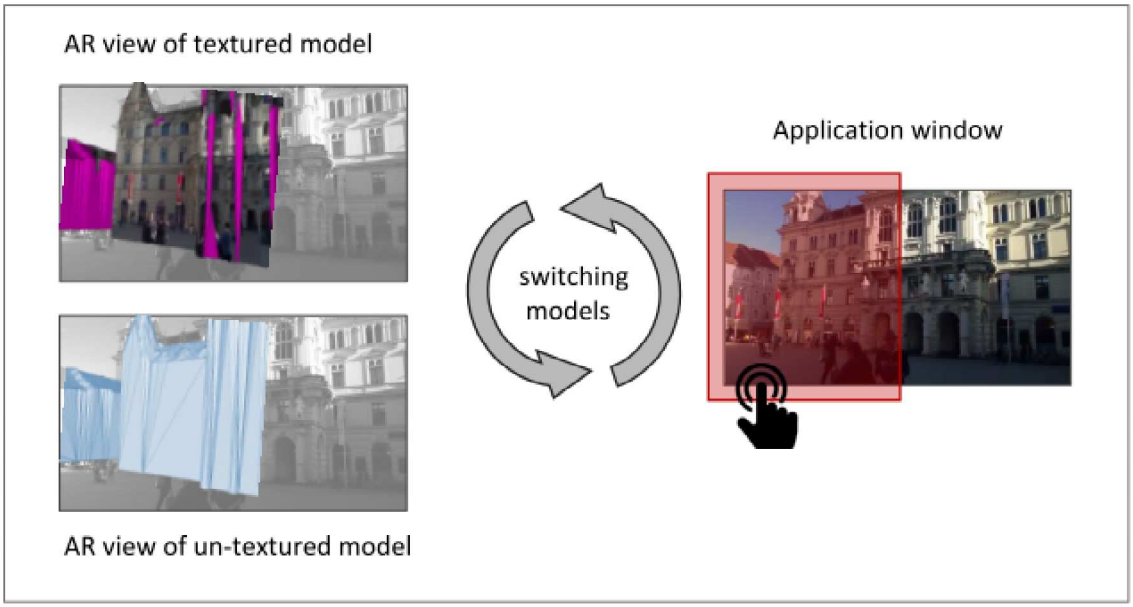
- Action (a): Switching between the visible and superimposed 3D models (between a textured and texture-less model). See figure 5.5(a)
- Action (b): Save and add the current camera view for a new texture mapping process (see figure 5.5(b)). The image file and the pose information (as a text file) is stored in a preconfigured folder in the file system. This action also initializes the texture mapping algorithm, which generates and saves a new 3D model (OBJ + MTL file), considering also all previously stored camera images/poses.

The two actions can be activated by a single click or touch on the display of the AR viewer window. The two actions are also illustrated and graphically explained in Figure 5.5. The intention of the UI is that a user of the mobile device is touching the screen with his left on the left side of the display to start action (a) and with the right hand on the right side of the screen to start action (b).

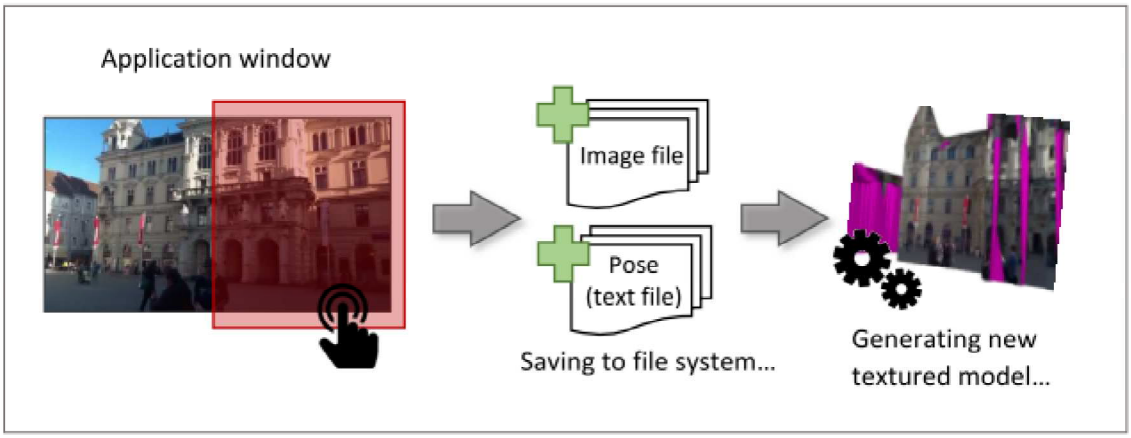
5.2.2 Further Remarks

The mobile AR viewer is designed to work with the device's positioning and orientation sensors. Therefore it's an outdoor application which relies on the GNSS position. The Figures 5.6 and 5.7 are showing screenshots of the AR application. Since the here used tablet delivers only poor GNSS positions (the error level is within several meters), the registration between virtual and real content is clearly missaligned. The offset also varies due to magnetic influences and drifts in the device's orientation sensors.

Experiences with the AR client brings us to the following conclusion: Basing a geospatial Augmented Reality application purely on the sensors of a consumer grade device can't guarantee a satisfying user experience. Influences on the GNSS position are especially in urban environments too high and so are the location displacements.



(a)



(b)

Figure 5.5: Illustrating the two actions of the user interface (UI). (a) Touch on left screen side: switching between two 3D models (textured and textureless model). (b) Touch on right screen side: Adding current image and pose to a newly started texture mapping process.

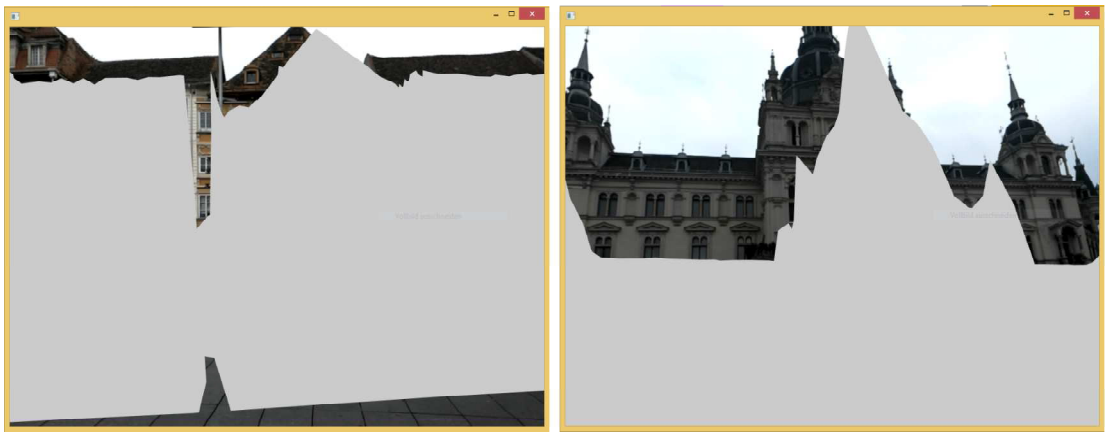


Figure 5.6: Screenshots from the AR viewer with a texture-less 3D model. Based on the device's GNSS position and orientation sensors, one can clearly see its inaccuracies in the wrong registration between the real camera view and the superimposed 3D model. Both screenshots were taken using the 3D model from data set 1 (See Section 5.1.1).

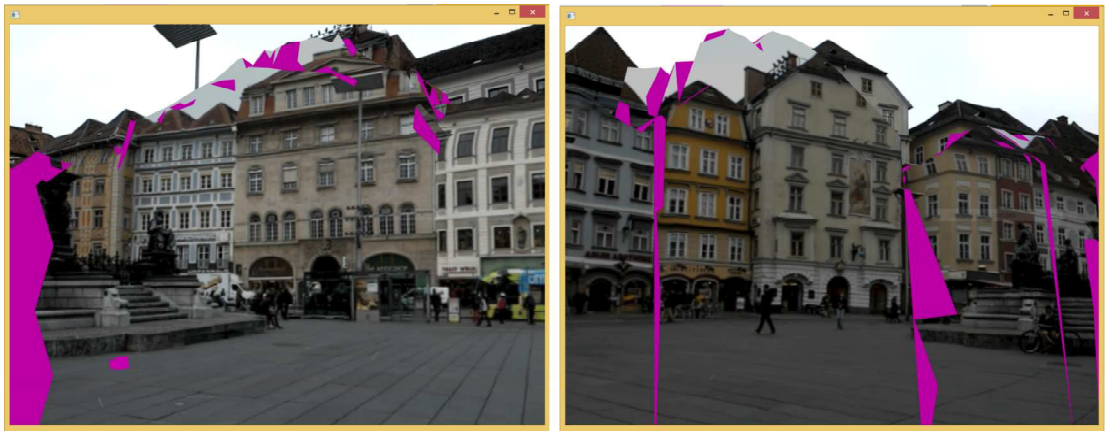


Figure 5.7: Screenshots from the AR viewer with a textured 3D model. The texture mapping where generated on site and includes the later discussed texturing inaccuracies. Individual texture-less mesh surfaces are colored in pink. Both screenshots were taken using the 3D model from data set 1 (See Section 5.1.1).

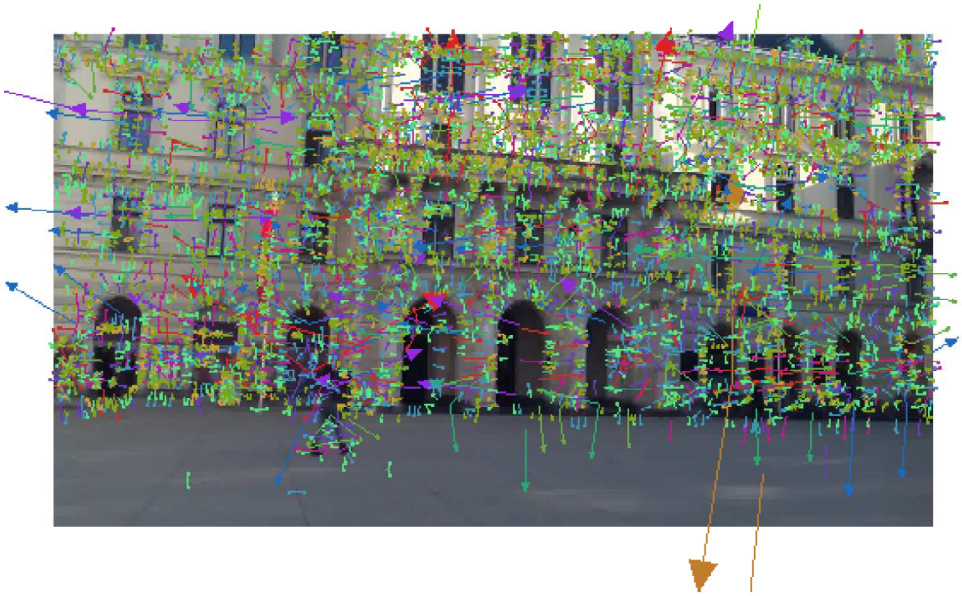


Figure 5.8: SIFT keypoints of one of the images in data set 2. Each arrow visualizes a SIFT keypoint with its gradient main orientation (direction of arrow) and its magnitude (length of arrow). Representation comes from VisualSFM.

5.3 3D Reconstruction Results

In Section 3.3 of Chapter 3 we already introduced the necessary steps in the context of a SfM based reconstruction of point clouds. Section 4.2 presented some SfM related tools and applications which has been used during this thesis. In this Section we describe some of the results of reconstructing 3D scenes form unordered image sequences.

At first it's necessary to identify unique feature points in all images (see Figure 5.8 for SIFT results in data set 2). As a next step the keypoints of objects, visible in multiple images, must be found and matched. The matching takes place pair-wise for all images. An outlier detection reduces false matches. Figure 5.9 shows some already filtered feature matches, shown as connected lines between corresponding feature points of two images.

Regarding the digital camera models, the reconstruction process follows two different approaches. Camera parameters can be predefined or be part of the estimation itself. In the latter approach, estimation for different individual camera's is supported (e.g. for processing images from various sources) as well as using one overall camera model.

The software now estimates camera poses and uses a triangulation algorithm for a

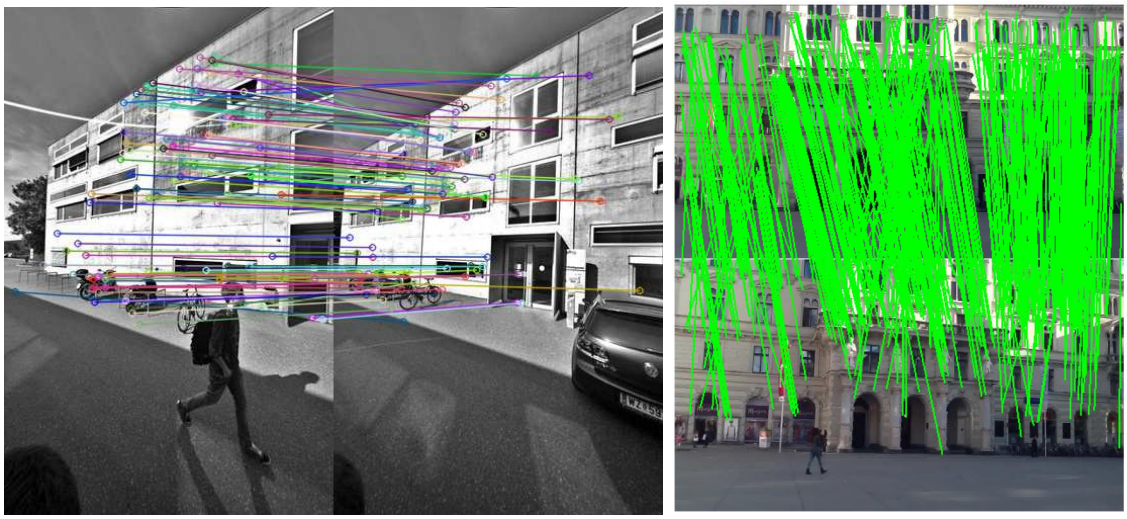


Figure 5.9: Matching SIFT feature points of two key-frames in each data set. On the left side the illustration of matches, calculated by using the OpenCV library. On the right the visualization form the VisualSFM software.

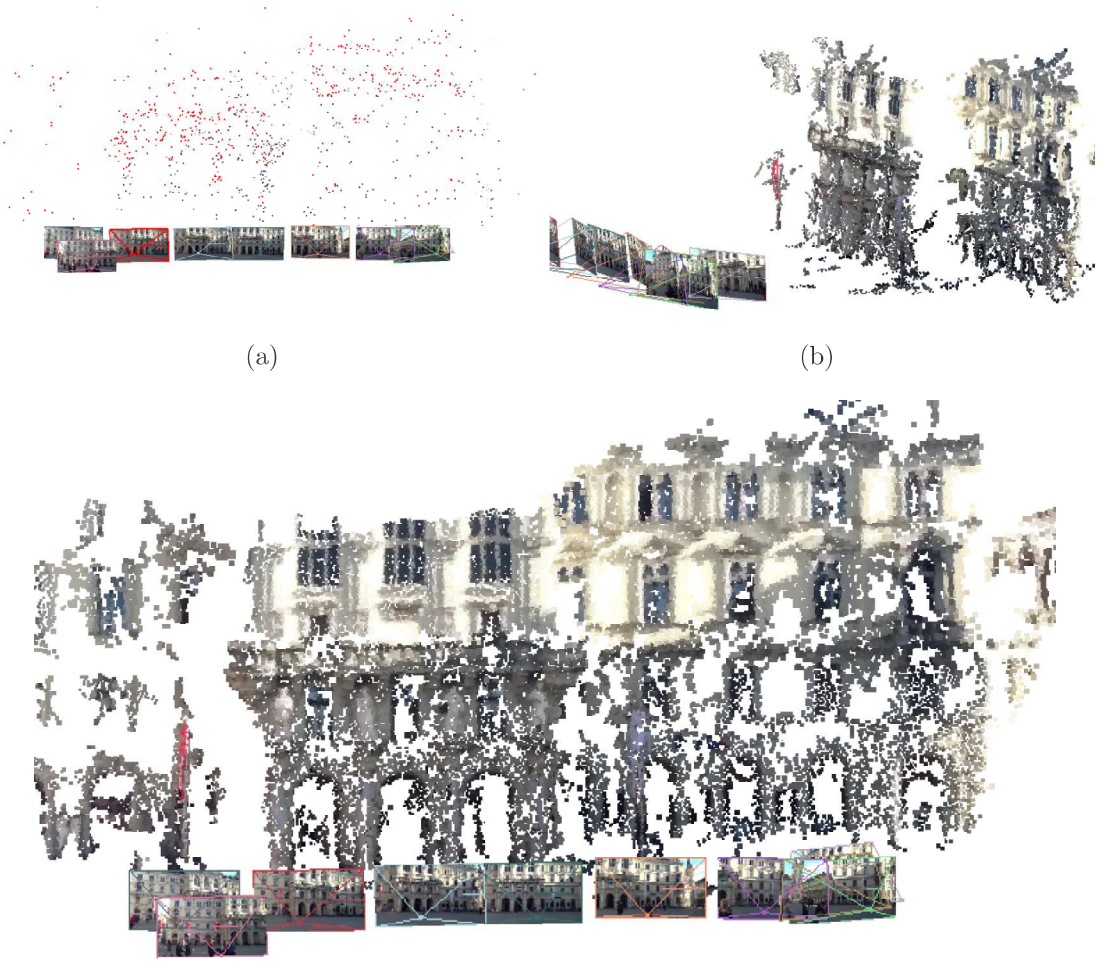
3D point reconstruction. With the limitation of having only a relativ small number of matching feature points, a sparse point cloud model - as in Figure 5.10(a) shown - is estimated. With a second reconstruction step (using PMVS/CMVS) a much denser 3D model has been calculated. Figure 5.10(c) shows the same scene as before, but with the much denser point cloud. The same point from data set 2, but from another perspective can be seen in Figure 5.10(b). Color information of the according pixels in the images are applied to build a textural information.

An optional Bundle Adjustment (BA) step refines the results. BA refers to an optimal adjustment of bundles of rays, reprojecting 3D feature points through the camera centres back on the 2D images. Optimization takes place over all cameras and 3D points to minimize the reprojection error. A bundle adjustment has been applied in this examples.

The reconstructed 3D scene is still in its own relative coordinate system. The initially recorded camera positions are now being used as so called ground-control-points (GCP). GCPs associate camera's with the recorded GNSS position, coming from the mobile recording device. Based on this information a global transformation matrix and scaling factor can be estimated.

The global transformation parameters for the data set 2 poses is listed as follows. Position translation ($\mathbf{t}$) refers to a UTM zone 33 coordinate system. $\mathbf{R}$ is the rotation matrix and s is the scaling factor.

```
1 s =     4.325002511953
```



(a) (b)

(c)

Figure 5.10: Reconstructed 3D point clouds of data set 2. (a) An initial and sparse point cloud. (b) Showing the same scene, a much denser reconstruction has been accomplished with an additional PMVS/CMVS step. (c) The same result of data set 2 from another perspective. Camera poses are shown as pyramids, holding the original image within.

```
R =     -0.58803966 -0.80357433 -0.09207418
        -0.60652389 0.51340189 -0.60708095
        0.53510572 -0.30114248 -0.78928770
t =     533259.631052926300
        5213133.957764443000
        375.193976735814
```

This transformation parameters must be applied to the extracted point cloud in order to change to world coordinates. The mathematical formula is listed in equation 5.1.

$$\mathbf{X}' = s \cdot \mathbf{R} \cdot \mathbf{X} + \mathbf{t} \tag{5.1}$$

5.4 Texture Mapping Results

5.4.1 Preliminary Steps

The limitations of the here used texture mapping algorithm have already been discussed in Section 4.3, Chapter 4. In order to achieve a higher degree of texture coverage, it was necessary to reduce the mesh sizes of the models. With smaller meshes it's more likely to have the whole mesh triangle visible in the field of view of a camera image. The open source software MeshLab[4] has been used to accomplish this tasks and to prepare the 3D models. Figure 5.11 shows the reduction of mesh sizes and the theoretical impact on surface coverage for a successful texture mapping.

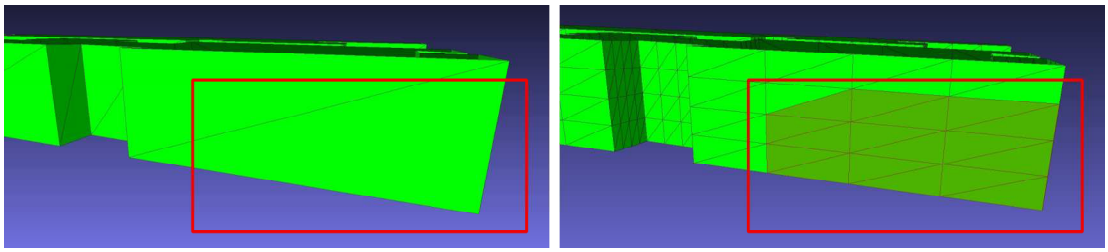
5.4.2 Texture mapping results

The Figures 5.12 and 5.13 show the texture mapping results of data set 1 and data set 2. The registration between image and 3D model in data set 1 is quite accurate. Whereas data set 2 shows a shifted texture due to inaccurate camera poses.

5.4.3 Texture mapping in "real-time"

Going a step further towards a real-time texture mapping application, the mobile AR viewer has been extended with the PCL based texturing algorithm. A user can col-

[4] http://meshlab.sourceforge.net/



(a) (b)

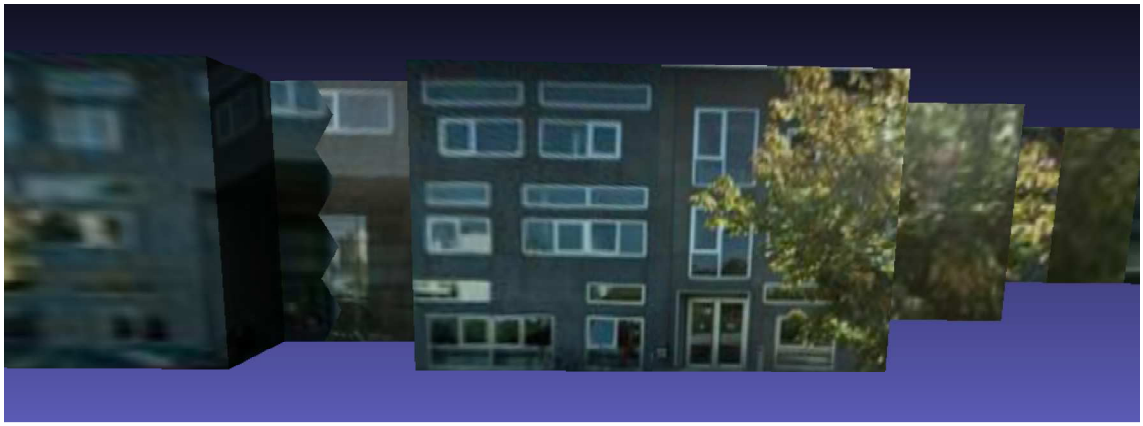
Figure 5.11: This figures show the result of subdividing the meshes of data set 1. (a) With the original mesh sizes and (b) with the reduced mesh size. Taking a theoretical photo of the scene, covering only a part shown as a red rectangle, would fully cover some of the smaller meshes (highlighted in (b)). Whereas the texture mapping process would fail completely in scenario (a), due to its limitations described in Chapter 4, Section 4.3.

lect snapshots from the camera's live stream by taping on the screen. The image and the accompanying pose is stored in separate files for an eventual later use. Simultaneously a new textured 3D model is being generated.

Figure 5.14 shows the texture mapping result from data captured and processed directly on site with the testing device. The pose information has been taken as is from the tablet's sensors. GNSS positions have inaccuracies of several meters which led to a rather poor mapping result.

5.4.4 Adjusted poses

Figure 5.15 depicts texture mapping results after a global adjustment step. As previously presented, the reconstructed poses from the SfM algorithm are situated in a local coordinate system. A purely image based reconstruction leads to relative good self-contained poses (as depicted in Fig. 5.10). After a similarity transform to multiple sensor based poses, the two systems are aligned. Multiple adjustment steps with the provided test data sets have shown that coarse pose inaccuracies in the global system, e.g. coming from a poor GNSS positioning quality, have a high influence on the previously good relative poses. Large outliers affect the global adjustment and therefore can reduce the quality in some details, whereas the overall quality has been optimized. Figure 5.15(b) shows some improvements, especially through adjusted heights, compared to a texture mapping result with raw sensor poses (Fig. 5.15(b)). This is due to the fact, that GNSS based height measurements are usually the



(a)

Figure 5.12: Left: Excerpt from the texturing result of dataset 1 (Inffeldgasse). Below the two corresponding photos. The photo on the right has been used first for mapping. That's why the tree is on the building's texture even though on the left photo the view would have been clear (first-come first-serve principle of the texturing process).

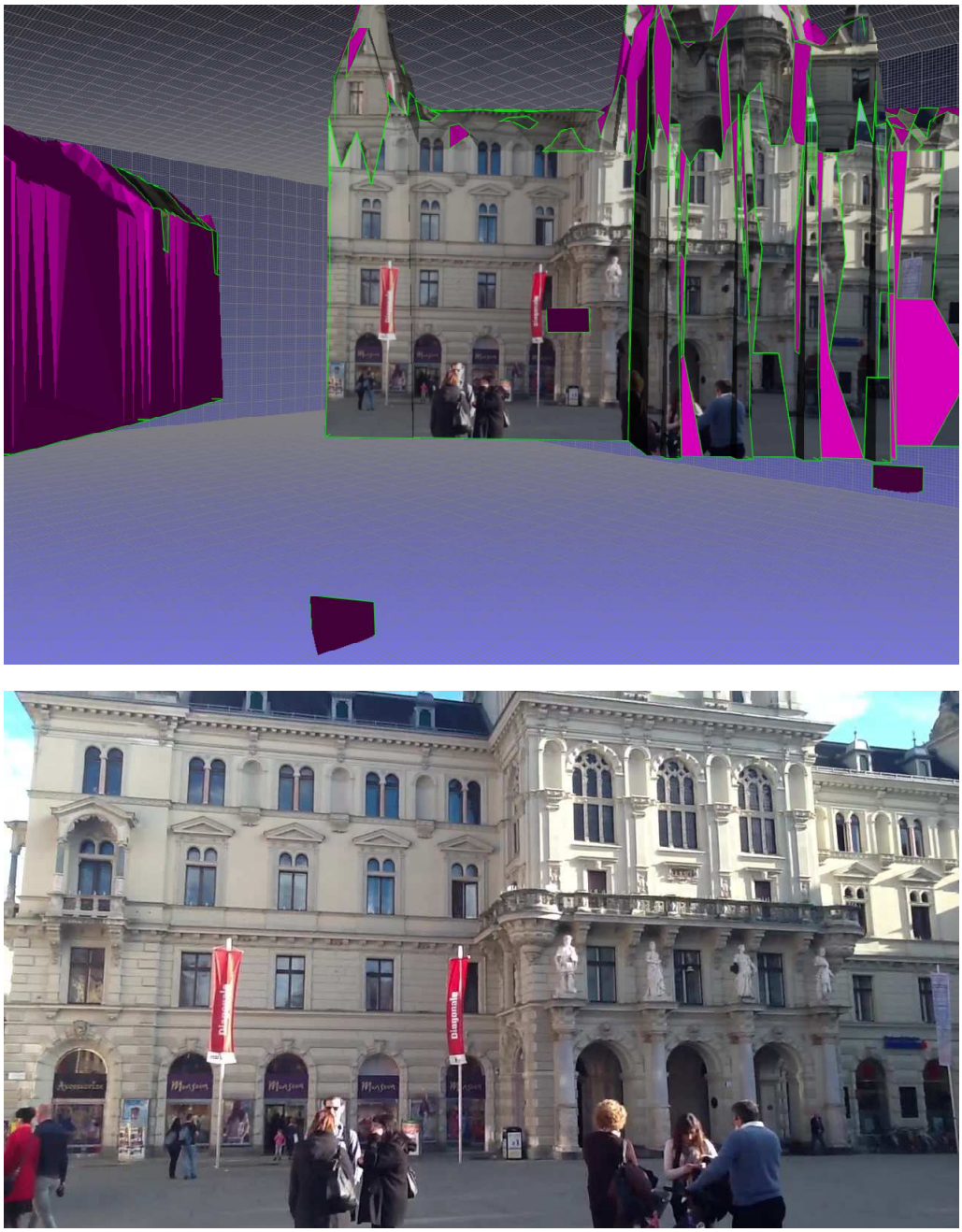


Figure 5.13: On Top: Excerpt from the texturing result of dataset 2 (city hall). Green lines are the texture seams. The biggest part of the facade comes from the one image shown below (the pose of the camera is also sketched as a pyramid in the above picture). The camera pose is shifted in position, that's also visible in the texturing result: Only four of the five windows are visible as the whole texture is shifted to the left and top.

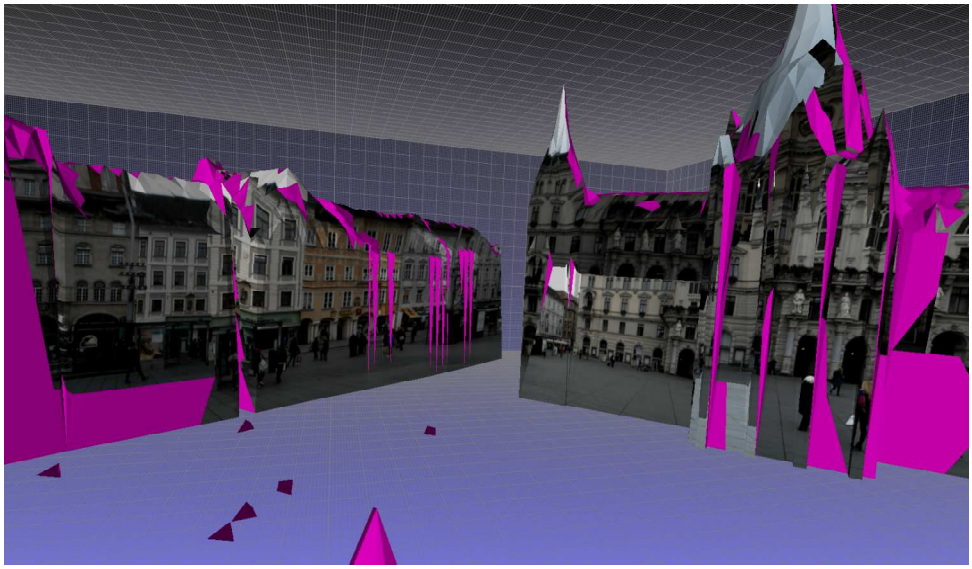


Figure 5.14: Model of the main square (Graz) after a texture mapping on site based on poses from the device's sensors (Toshiba Encore tablet)

most inaccurate, compared to a horizontal positioning. However, the overall texture mapping quality, achieved with this approach, is still insufficient and needs further improvements, which are addressed in Chapter 6.

5.4.5 Comparison to related models

Mapping textures in a geometrically accurate sense is just one of the challenges in texturing 3D models. Some other issues regard the mosaicing of multiple images, especially when taken under different lighting conditions or at different times of a day. Figure 5.16 shows fully textured 3D models of the two in our data sets used building models. These screenshots, coming from Google Earth[5], show rather good textured 3D models, but also identifiy the radiometric problems of shadowing and sunlight. The image material comes from aerial and/or satellite based photos. Advantages from close range cameras, such as with our AR prototype, would be an eventual higher resolution and better image quality.

[5] `https://earth.google.com/`

(a) (b)

Figure 5.15: Texture mapping results with (a) the raw sensor poses, (b) adjusted poses by location and height from a SfM reconstruction + global alignement. The biggest improvements are coming from adjusted camera heights, whereas the overall quality, achieved with this approach, is still insufficient.

(a)

(b)

Figure 5.16: Textured 3D models of the buildings used in data set 1 (a) and data set 2 (b) from Google Earth[5]. Whereas these examples show a good registration of the textures, (b) also has a dark north-facade, caused by daytime dependent recording moment of the aerial/satellite photo.

Chapter 6

Conclusion

6.1 Concluding Remarks

In scope of this thesis multiple subjects have been addressed. The mobile AR viewer application has been used to discover the fundamental principles of Augmented Reality. Especially during the developing phases for the different topics, the author gained much experience in the development of mobile applications, working with open source libraries and 3D graphics and computer vision mechanisms. Once more it's worth to mention that the right choice of software libraries has been demonstrated to be a very important step, since it can imply many difficulties in development details with no or a barely maintained and documented software. Likewise the compiling for the right developing architecture and the choices of platforms and compilers.

Texture Mapping has been integrated to the mobile AR viewer. This scenario demonstrates for example the use case of a mobile texture mapping application by simultaneously adding an AR user interface. The texturing results also show some of the challenges of texture mapping, such as radiometric image adjustments under different recording conditions with multiple images.

SfM and SLAM are potential algorithms for improving camera poses in a global bundle adjustment. These algorithm also demonstrate the power of smartphones in the field of on-site and real-time reconstructions and generation of 3D models. Making smartphones to a new group of surveying devices in the fields of geodesy.

6.2 Future Work

Further improvements in investigations for a future work haven been identified. The most import topics shall be discussed here in the following structured list

Towards a fully mobile Augmented Reality application

- From a technical point of view, the developed AR viewer can only be seen as experimental prototype. Going towards a real case mobile AR application, more intensive considerations for the right choice of AR libraries and development platforms must be undertaken. When developing for various mobile platforms looking on their market share is often a good criteria. Therefore the top mobile operating systems (OS) at the moment are Android and iOS. Distribution of mobile Windows operating systems and others, especially OS for AR head-mounted-displays, will show in near future their significance in the market.
- A mobile AR system has to be specialized for specific use cases and markets. In the scope of this thesis a mobile AR GIS has been discussed. A better connectivity to GIS and geospatial data is just one but crucial criteria for a mobile GIS. One step therefore could be the implementation of the earlier discussed OGC standards.

Improvements in texture mapping

- In the area of texture mapping further investigations are advised. The capabilities of state-of-the art, and their still to-be-mastered difficulties weren't scope of this thesis. The integration of real-time capable algorithms for mobile platforms still has to be proven.

Realising the potential of SfM and SLAM

- SfM, SLAM and other Computer Vision related topics are highly suitable to be integrated in traditional geodetic applications:
  - Reconstruction of a user's surrounding with vision based SfM algorithms starting to be real-time capable.
  - Reconstructed 3D point clouds can be further used to interpolate and generated meshed 3D models.
  - Using local and on-the-fly updated 3D maps for localizations, have the potential of competing with traditional positioning systems. Advantages herby lie in the cheap set up of the necessary infrastructure, especially

since mobile and powerful devices and sensors are more and more available and connected. Fusing more sensors makes it easier to improve poses and be more reliable in difficult environments. The trend of the ”internet of things” brings up many new and exciting developments in this direction.

Appendix A

Camera Calibration for Toshiba Encore

In order to work with camera projections, texture mapping and reconstruction, it's important to know the intrinsic parameters of the camera model. Chapter 3 already covered and presented the mathematical backround therefore. The calibration of the tablet's camera intrinsic parameters has been accomplished using the Bouguet Camera Calibration Toolbox for Matlab [15] and a sequence of 23 checkerboard images (see Figure A.2). The Bouguet Toolbox allows us to calibrate a camera model of the following form:

$$\mathbf{K} = \begin{pmatrix} f_x & \alpha_c \cdot f_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{A.1}$$

- The 2x1 Vector **f** with the components f_x and f_y describes the focal length in x and y direction expressed in units of horizontal and vertical pixels.
- Principal point as the 2x1 vector **c** with pixel coordinates of c_x and c_y.
- Aspect ratio f_y/f_x which is different from 1 if the pixels are not square.
- Skew s is expressed as $\alpha_c \cdot fc(1)$ and denotes the angle between the x and y sensor axes.

The camera matrix **K** holds the intrinsic calibration parameters.

In order to get a good calibration result the checkerboard should be moved around in the camera frame (or vice versa) such that ...

- ... the checkerboard is detected at the left and right edges of the field of view (X calibration).
- ... the checkerboard is detected at the top and bottom edges of the field of view (Y calibration).
- ... the checkerboard is detected at various angles to the camera (Skew).
- ... the checkerboard fills the entire field of view (Size calibration).

- ... checkerboard tilted to the left, right, top and bottom (X,Y, and Size calibration) .

Intrinsic parameters of the RGB camera

Resolution of RGB 8.0 MP camera: 3264 x 2448 pixel

$$\mathbf{K} = \begin{pmatrix} f_c & \alpha_c \cdot f_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2666.12 \pm 3.36 & 0.0 \pm 0.0 \cdot fc(1) & 1630.25 \pm 4.50 \\ 0 & 2664.45 \pm 3.28 & 1223.01 \pm 3.69 \\ 0 & 0 & 1 \end{pmatrix}$$

Distortion: $\mathbf{k}_c = [\ 0.065 \quad -0.117 \quad 0.002 \quad 0.001 \quad 0.0\ ] \pm [\ 0.0037 \quad 0.0067 \quad 0.0005 \quad 0.0006 \quad 0.0\ ]$

Pixel error: $err = [\ 1.017 \quad 0.892\ ]$

The distortion parameters are also modeled and visualized in Figure A.1. The distortion model can be splitted in a radial and a tangential component to describe the total camera distortion effects. In the further computations in this thesis, the influence of the distortion have been neglected. This is firstly due to the usage of the simplier 5 parameter camera model and secondly, the faced influences of pose inaccuracies outweigh camera distortions by far. Being aware that distortion effects are stronger in the edge regions of the camera's image, the impact in practice has been left as subject for future investigations.

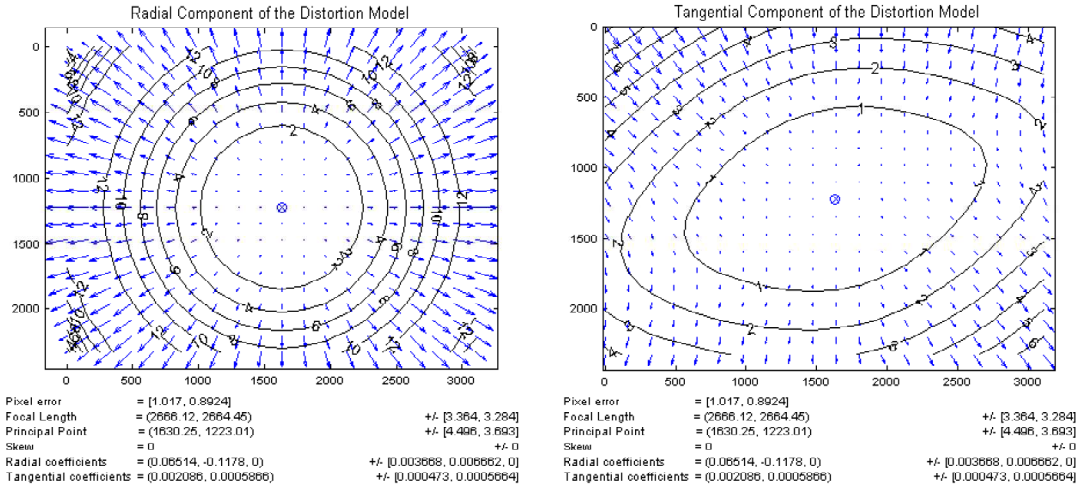


Figure A.1: Radial (left) and tangential component (right) of the distortion model

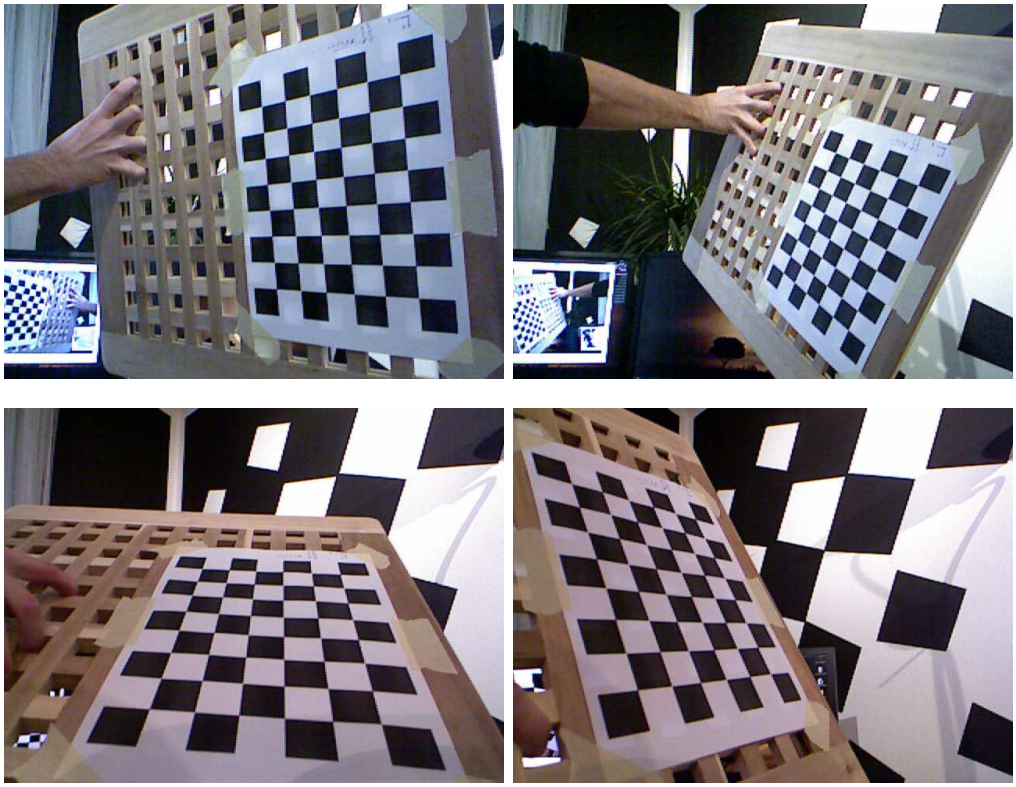


Figure A.2: Four images with a checkerboard, which has been used with 19 other images for the tablet's camera calibration. Photos have been taken from different angles and distances to the checkerboard.

List of Figures

List of Tables

Bibliography

[1] Augview. http://www.augview.net, 2013. last accessed: 2015-09-11.

[2] EUREF Permanent Network. http://www.epncb.oma.be/_organisation/projects/euref_IP/, March 2014. last accessed: 2015-09-11.

[3] Swift Navigation - Piksi. http://www.swiftnav.com/piksi.html, 2014. last accessed: 2015-09-11.

[4] Android Developers SDK. http://developer.android.com/reference/android/location/Location.html, 2015. last accessed: 2015-09-11.

[5] Layar. http://www.layar.com/, 2015. last accessed: 2015-09-11.

[6] United States National Geospatial-Intelligence Agency - The World Magnetic Model. http://www.ngdc.noaa.gov/geomag/WMM/DoDWMM.shtml, 2015. last accessed: 2015-09-11.

[7] Wikitude. http://www.wikitude.com/, 2015. last accessed: 2015-09-11.

[8] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. pages 72–79, 2009.

[9] Mohamed Alji. OpenSceneGraph. https://en.wikipedia.org/wiki/OpenSceneGraph, August 2011. last accessed: 2015-09-11.

[10] C. Arth, R. Grasset, L. Gruber, T. Langlotz, A. Mulloni, and D. Wagner. The History of Mobile Augmented Reality. 2015.

[11] C. Arth, J. Ventura, and D. Schmalstieg. Geospatial Management and Utilization of Large-Scale Urban Visual Reconstructions. pages 64–69. IEEE, 2013.

[12] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, 1987.

[13] R. T. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, August 1997.

[14] D. L. Baggio. *Mastering OpenCV with practical computer vision projects*. Packt Publ, Birmingham [u.a.], 1. publ. edition, 2012.

[15] J.-Y. Bouguet. Camera Calibration Toolbox for Matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/`, December 2013. last accessed: 2015-09-11.

[16] P. Daubmeier. Sensor Emitter - Learn about your phones sensors. `http://philip.daubmeier.de/sensoremitter/`, 2012. last accessed: 2015-09-11.

[17] D. Davelli and A. Signoroni. Automatic Mapping of Uncalibrated Pictures on Dense 3D Point Clouds. pages 576–581. University of Trieste and University of Zagreb, 2013.

[18] A. J. Davison, W. W. Mayol, and D. W. Murray. Real-time Localization and Mapping with Wearable Active Vision. pages 18–27. IEEE, 2003.

[19] A.J. Davison. Real-time Simultaneous Localisation and Mapping with a single Camera. pages 1403–1410 vol.2. IEEE, 2003.

[20] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. September 2014.

[21] S. Feiner, B. MacIntyre, T. Höllerer, and A. Webster. A touring machine: prototyping 3D mobile augmented reality systems for exploring the urban environment. pages 74–81. IEEE, 1997.

[22] Y. Furukawa. Clustering Views for Multi-view Stereo (CMVS). `http://www.di.ens.fr/cmvs/`, 2010. last accessed: 2015-09-11.

[23] S. Genc and V. Atalay. Texture Extraction from Photographs and Rendering with Dynamic Texture Mapping. pages 1055–1058, 1999.

[24] European Global Navigation Satellite Systems Agency (GSA). Precise and Robust Navigation enabling Applications in Disturbed Signal Environments. http://www.gsa.europa.eu/precise-and-robust-navigation-enabling-appli-cations-disturbed-signal-environments, 2015. last accessed: 2015-09-11.

[25] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge Univ. Press, Cambridge [u.a.], 2., 3. print.; 2; 2. edition, 2006; 2004; 2003.

[26] B. Hofmann-Wellenhof, K. Legat, and M. Wieser. *Navigation: Principles of Positioning and Guidance*. Springer, Wien [u.a.], 2003.

[27] B. Hofmann-Wellenhof, H. Lichtenegger, and E. Wasle. *GNSS - Global Navigation Satellite Systems: GPS, GLONASS, Galileo, and more*. Springer, Wien [u.a.], 2008.

[28] B. K. P. Horn. Closed-form Solution of Absolute Orientation using Unit Quaternions. *Journal of the Optical Society of America*, 4:629–642, 1987.

[29] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From Structure-from-Motion Point Clouds to fast Location Recognition. pages 2599–2606. IEEE, 2009.

[30] V. Kaufmann. Lecture notes 'VO Bildmessung', WS 2011/12.

[31] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. pages 225–234. IEEE, 2007.

[32] C. Klug. Presentation 'Texture mapping pipelines - Implemented by and available from KLUG' from 12/11/2014, 2014.

[33] E. Kruijff, E. Mendez, E. Veas, and T. Gruenewald. On-site Monitoring of Environmental Processes using Mobile Augmented Reality (HYDROSYS). *CEUR Workshop Proceedings*, 679, 2010.

[34] R. G. Laycock and A. M. Day. Automatic Techniques for Texture Mapping in Virtual Urban Environments. pages 586–589. IEEE, 2004.

[35] F. Leberl, A. Irschara, T. Pock, P. Meixner, M. Gruber, S. Scholz, and A. Wiechert. Point Clouds: Lidar versus 3D Vision. *Photogrammetric Engineering and Remote Sensing*, 76(10):1123–1134, 2010.

[36] D. G. Lowe. Object Recognition from local Scale-Invariant Features. volume 2, pages 1150–1157 vol.2, 1999.

[37] G. Milette, A. Stroud, and Inc Books24x7. *Professional Android Sensor Programming*. Wrox Press Ltd, US, 1 edition, 2012.

[38] P. Milgram and A. F. Kishino. Taxonomy of Mixed Reality Visual Displays. In *IEICE Transactions on Information and Systems.*, pages 1321–1329, 1994.

[39] M. Muja and D. G. Lowe. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09*, pages 331–340. INSTICC Press, 2009.

[40] OpenSceneGraph. OpenSceneGraph - Official Project Website. `http://www.openscenegraph.org/`, 2007. last accessed: 2015-09-11.

[41] T. Oskiper, H.-P. Chiu, Z. Zhu, S. Samarasekera, and R. Kumar. Stable Vision-Aided Navigation for Large-Area Augmented Reality, pages=63-70, isbn=1087-

8270, language=English. 2011.

[42] T. Oskiper, S. Samarasekera, and R. Kumar. Multi-Sensor Navigation Algorithm using Monocular Camera, IMU and GPS for large scale Augmented Reality. pages 71–80. IEEE, 2012.

[43] M. Peryt. Map Projections - Geographic Coordinate System. http://arch.web.cern.ch/arch/JViews55/doc/userman/mapsuser/html/projections8.html, 2002. last accessed: 2015-09-11.

[44] M. Previtali, L. Barazzetti, and M. Scaioni. An Automated and Accurate Procedure for Texture Mapping from Images. pages 591–594. IEEE, 2012.

[45] E. Rath. Coordinate Systems in General and in OpenGL especially. http://www.matrix44.net/cms/notes/opengl-3d-graphics/coordinate-systems-in-opengl, 2015. last accessed: 2015-09-11.

[46] G. Reitmayr. Mobile Visual Computing - Sensors. http://www.icg.tugraz.at/Members/G./mvc/MVC_04_Sensors.pdf/view/, 2011. last accessed: 2015-09-11.

[47] G. Reitmayr. Augmented Reality SDK Comparison. http://socialcompare.com/en/comparison/augmented-reality-sdks, 2015. last accessed: 2015-09-11.

[48] G. Reitmayr and T. W. Drummond. Going out: Robust model-based tracking for outdoor augmented reality. 2006.

[49] G. Reitmayr, T. Langlotz, D. Wagner, A. Mulloni, G. Schall, D. Schmalstieg, and Qi Pan. Simultaneous Localization and Mapping for Augmented Reality. pages 5–8. IEEE, 2010.

[50] C. Rocchini, P. Cignoni, C. Montani, and R. Scopigno. Multiple Textures Stitching and Blending on 3D Objects. In *In Eurographics Rendering Workshop*, pages 119–130. Springer-Verlag, 1999.

[51] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[52] G Schall, S Junghanns, and D Schmalstieg. VIDENTE - 3D Visualization of Underground Infrastructure using Handheld Augmented Reality. In *GeoHydroinformatics: integrating GIS and water engineering*, pages 207–219. CRC Press, 2010.

[53] G. Schall, S. Zollmann, and G. Reitmayr. Smart Vidente: Advances in Mobile

Augmented Reality for Interactive Visualization of Underground Infrastructure. *Personal and Ubiquitous Computing*, 17(7):1533–1549, 2013.

[54] N. Snavely. Bundler: Structure from Motion (SfM) for Unordered Image Collections. `http://www.cs.cornell.edu/~snavely/bundler/`, 2010. last accessed: 2015-09-11.

[55] I. E. Sutherland. A Head-Mounted Three Dimensional Display. In *Fall Joint Computer Conference. AFIPS Press, Montvale, N.J.*, pages 757–764, 1968.

[56] G. Takacs, M. El Choubassi, Yi Wu, and I. Kozintsev. 3D Mobile Augmented Reality in Urban Scenes. pages 1–4, 2011.

[57] S. Umeyama. Least-Squares Estimation of Transformation Parameters between two Point Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.

[58] E. Veas, R. Grasset, E. Kruijff, and D. Schmalstieg. Extended Overview Techniques for Outdoor Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):565–572, 2012.

[59] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg. Global Localization from Monocular SLAM on a Mobile Phone. *IEEE Transactions on Visualization and Computer Graphics*, 20(4):531–539, 2014.

[60] J. Ventura and T. Höllerer. Wide-area Scene Mapping for Mobile Visual Tracking. pages 3–12. IEEE, 2012.

[61] R. Wang and X. Qian. *OpenSceneGraph 3.0: Beginner's Guide*. Packt Publishing, GB, 2010.

[62] M. J. Westoby, J. Brasington, N. F. Glasser, M. J. Hambrey, and J. M. Reynolds. 'Structure-from-Motion' photogrammetry: A low-cost, effective tool for geoscience applications. *Geomorphology*, 179, 2012.

[63] C. Wu. VisualSFM: A Visual Structure from Motion System. `http://ccwu.me/vsfm/`. last accessed: 2015-09-11.

[64] Haiyang Xu. osm-bundler: A Python routine for running Structure From Motion pipeline with Bundler and dense reconstruction with PMVS(CMVS) to reconstruct 3D geometry from a set of photos. `http://haiyangxu.github.io/osm-bundler/`. last accessed: 2015-09-11.