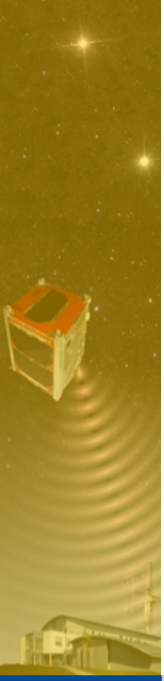
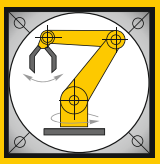


Institut für Informatik
Lehrstuhl für Robotik und Telematik
Prof. Dr. K. Schilling



Würzburger Forschungsberichte
in Robotik und Telematik

Uni Wuerzburg Research Notes
in Robotics and Telematics

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Band 14

Dorit Borrmann

Multi-modal 3D mapping

Combining 3D point clouds with thermal and color information

Die Schriftenreihe

wird vom Lehrstuhl für Informatik VII: Robotik und Telematik der Universität Würzburg herausgegeben und präsentiert innovative Forschung aus den Bereichen der Robotik und der Telematik.

Die Kombination fortgeschrittener Informationsverarbeitungsmethoden mit Verfahren der Regelungstechnik eröffnet hier interessante Forschungs- und Anwendungsperspektiven. Es werden dabei folgende interdisziplinäre Aufgabenschwerpunkte bearbeitet:

- Robotik und Mechatronik: Kombination von Informatik, Elektronik, Mechanik, Sensorik, Regelungs- und Steuerungstechnik, um Roboter adaptiv und flexibel ihrer Arbeitsumgebung anzupassen.

- Telematik: Integration von Telekommunikation, Informatik und Steuerungstechnik, um Dienstleistungen an entfernten Standorten zu erbringen.

Anwendungsschwerpunkte sind u.a. mobile Roboter, Tele-Robotik, Raumfahrtsysteme und Medizin-Robotik.

Lehrstuhl Informatik VII
Robotik und Telematik
Am Hubland
D-97074 Wuerzburg

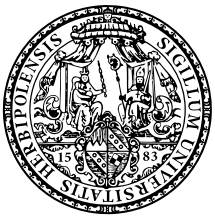
Tel.: +49 (0) 931 - 31 - 86678
Fax: +49 (0) 931 - 31 - 86679

schi@informatik.uni-wuerzburg.de
http://www7.informatik.uni-wuerzburg.de

ISSN: 1868-7474 (online)
ISSN: 1868-7466 (print)
ISBN: 978-3-945459-20-1 (online)

Zitation dieser Publikation

BORRMANN, D. (2018). Multi-modal 3D mapping - Combining 3D point clouds with thermal and color information. Schriftenreihe Würzburger Forschungsberichte in Robotik und Telematik, Band 14. Würzburg: Universität Würzburg.
URN: urn:nbn:de:bvb:20-opus-157085



Julius-Maximilians-Universität Würzburg
Chair of Computer Science VII
Robotics and Telematics

Dissertation

Multi-modal 3D mapping

Combining 3D point clouds with thermal and color information

submitted to the Faculty of
Mathematics/Computer Science
of the University of Würzburg
in fulfillment of the requirements for the degree of
Doctor Rerum Naturalium (Dr. rer. nat.)
by

Dorit Borrmann

September 2017

Supervisor and first reviewer:	Prof. Dr. Andreas Nüchter
Second reviewer:	Prof. Dr. Joachim Hertzberg
Third reviewer:	Prof. Dr. Claus Brenner

Danksagung

Diese Dissertation bedeutet den Abschluss eines wichtigen Kapitels meiner wissenschaftlichen Laufbahn und somit möchte ich die Gelegenheit nutzen, den Menschen zu danken, die mich auf meinem Weg begleitet haben und ohne deren andauernde Unterstützung dieses Projekt nicht möglich gewesen wäre.

Der erste Dank gebührt meinem Doktorvater Prof. Dr. Andreas Nüchter für seine unvergleichbare Betreuung, fortwährende Unterstützung und das mir entgegen gebrachte Vertrauen. In der langen Zeit der Zusammenarbeit hat er mir die Möglichkeit gegeben viele Erfahrungen zu sammeln und mich ständig weiter zu entwickeln und dabei auch stets meine Interessen im Auge behalten. Besonderer Dank gilt ebenfalls Prof. Dr. Joachim Hertzberg und Prof. Dr. Claus Brenner für die schnelle Begutachtung meiner Dissertation, sowie Prof. Dr. Frank Puppe und Dr. Christian Herrmann für ihr Mitwirken während der Disputation. Dem Inhaber des Lehrstuhls für Robotik und Telematik, Prof. Dr. Klaus Schilling, danke ich dafür, dass ich durch die fachliche Vielfalt am Lehrstuhl Einblick in spannende Forschungsgebiete erlangen konnte.

Diese Arbeit ist ein Ergebnis unzähliger Stunden Arbeit, die ich glücklicherweise nicht komplett alleine bewältigen musste. Im Laufe der letzten Jahre hatte ich die Gelegenheit mit vielen Kollegen zusammen zu arbeiten, die durch zahlreiche Ideen, fruchtbare Diskussionen, ihr Engagement und ihren Sachverstand einen wesentlichen Beitrag zu meiner Arbeit geleistet haben. An dieser Stelle möchte ich all meinen Ko-Autoren für die gute Zusammenarbeit herzlich danken. Ganz besonders erwähnenswert sind hier meine Projektpartner aus dem Projekt ThermalMapper (SEE-ERA.NET Projektnummer ERA 14/01), dessen Forschungsergebnisse wesentliche Bestandteile meiner Dissertation sind. Besonders hervorheben möchte ich Marija Seder und Ivan Maurović, mit denen die zahlreichen Stunden bei gemeinsamen Experimenten immer Spaß gemacht haben. Die Förderung des SEE-ERA.NET Projektes ThermalMapper und die Unterstützung des DAAD mit den PPP Projekten "Thermal Perception, Mapping and Exploration of Environments for Energy Efficiency" sowie ThermoBIM haben diese Zusammenarbeit erst möglich gemacht.

Ein besonderer Reiz in meiner Arbeit bestand in der Verwendung von Datensätzen aus unterschiedlichen Anwendungsgebieten. Dafür, dass sie dies auf unterschiedliche Weise ermöglicht haben, durch ihr Interesse, ihren Einsatz und ihre tatkräftige Unterstützung möchte ich hier stellvertretend Danke sagen an die Soprintendenza Speciale per i Beni Archeologici di Roma - Sede di Ostia (A. Pellegrino) und die Universitià Sapienza di Roma (S. Falzone) sowie Norbert Zimmermann und Irmengard Mayer für die Aufnahmen in Ostia Antica, der “Bayerische Verwaltung der staatlichen Schlö sser, Gärten und Seen für den Zugang zur Würzburger Residenz, der denkmal3D GmbH und der Landesarchäologie Bremen für den Bremer Bräutigam, dem AWI Helgoland sowie Vikram Unnithan und Angelo Rossi für die Aufnahmen der Langen Anna auf Helgoland und dem Universum Bremen für den Zugang zur Ausstellung mit unserem Roboter. Nicht vergessen möchte ich an dieser Stelle auch Irma3D für die zuverlässige Arbeit und die bereitwillige Befolgung unserer Befehle.

Bedanken möchte ich mich auch bei meinen aktuellen und ehemaligen Kollegen in Osnabrück, Bremen und Würzburg. Es war mir immer eine große Freude mit euch zusammen zu arbeiten und nicht nur die fachlichen Gespräche sondern auch die nicht so fachlichen Gespräche und Aktivitäten haben immer zu einer guten Arbeitsatmosphäre beigetragen und damit meine Ar-

beit bereichert. Besonders erwähnen möchte ich Kai Lingemann, der mich während meiner Bachelor- und Masterarbeit betreut hat und auch hinterher immer für Fragen und Diskussionen bereit stand. Repräsentativ für die gute Zeit in Würzburg stehen Michael Bleier, Daniel Eck, Christian Herrmann, Robin Heß, Florian Kempf, Florian Leutert, Tobias Lindeholz und Lakshminarasimhan Srinivasan. Dieter Ziegler danke ich für die technische Unterstützung bei der Roboterpflege. Meinen langjährigen Kollegen Hamidreza Houshiar und Jan Elseberg gebührt besondere Anerkennung, für die vielen Stunden, die wir bei gemeinsamen Projekten verbracht haben, und den Spaß der trotz der intensiven Arbeit nie zu kurz gekommen ist. Speziell bei meinem ehemaligen Bürokollegen Jan Elseberg, der mich bereits seit Beginn meines Studiums begleitet hat, bedanke ich mich für die ausgezeichnete Zusammenarbeit und den regen Meinungs- und Gedankenaustausch. Die vielen fachlichen Gespräche und Diskussionen haben meine Arbeit stets bereichert und wertvolle Anregungen geliefert.

Des Weiteren durfte ich in den letzten Jahren mit vielen Studenten zusammenarbeiten, von denen einige inzwischen Kollegen geworden sind. Stellvertretend möchte ich hier Sven Jörissen, David Redondo, Helge Lauterbach und Hassan Afzal für ihre Beiträge in Forschung und Lehre danken.

Der größte Dank geht an meine Eltern, die mich schon mein ganzes Leben lang unterstützen und mich in meiner Arbeit immer wieder bestärkt haben. Auch meinen Brüdern und Freunden danke ich für ihren Rückhalt, ihre Geduld sowie die gelegentliche Ablenkung.

Zuletzt möchte ich noch herzlich Christian Herrmann, Robin Heß, Hamidreza Houshiar, Florian Kempf, Ina Schulze-Rajabali, Lakshminarasimhan Srinivasan und ganz besonders Florian Leutert danken, die durch ihre fleißige Fehlersuche und ihre konstruktive Kritik beim Korrekturlesen einen großen Beitrag dazu geleistet haben, dass diese Arbeit nun in dieser Form vorliegt.

Abstract

Imagine a technology that automatically creates a full 3D thermal model of an environment and detects temperature peaks in it. For better orientation in the model it is enhanced with color information. The current state of the art for analyzing temperature related issues is thermal imaging. It is relevant for energy efficiency but also for securing important infrastructure such as power supplies and temperature regulation systems. Monitoring and analysis of the data for a large building is tedious as stable conditions need to be guaranteed for several hours and detailed notes about the pose and the environment conditions for each image must be taken. For some applications repeated measurements are necessary to monitor changes over time. The analysis of the scene is only possible through expertise and experience.

This thesis proposes a robotic system that creates a full 3D model of the environment with color and thermal information by combining thermal imaging with the technology of terrestrial laser scanning. The addition of a color camera facilitates the interpretation of the data and allows for other application areas. The data from all sensors collected at different positions is joined in one common reference frame using calibration and scan matching. The first part of the thesis deals with 3D point cloud processing with the emphasis on accessing point cloud data efficiently, detecting planar structures in the data and registering multiple point clouds into one common coordinate system. The second part covers the autonomous exploration and data acquisition with a mobile robot with the objective to minimize the unseen area in 3D space. Furthermore, the combination of different modalities, color images, thermal images and point cloud data through calibration is elaborated. The last part presents applications for the the collected data. Among these are methods to detect the structure of building interiors for reconstruction purposes and subsequent detection and classification of windows. A system to project the gathered thermal information back into the scene is presented as well as methods to improve the color information and to join separately acquired point clouds and photo series.

A full multi-modal 3D model contains all the relevant geometric information about the recorded scene and enables an expert to fully analyze it off-site. The technology clears the path for automatically detecting points of interest thereby helping the expert to analyze the heat flow as well as localize and identify heat leaks. The concept is modular and neither limited to achieving energy efficiency nor restricted to the use in combination with a mobile platform. It also finds its application in fields such as archaeology and geology and can be extended by further sensors.

说文章的第一部分解决了三个问题

Contents

Chapter 1

Introduction

Sensor technology has undergone a rapid development in recent years. Reliable GNSS (Global Navigation Satellite System), inclination and acceleration sensors are available in every smartphone nowadays, but even more evident is the entrance of optical sensors into our everyday life. Hardly anything happens anymore without it being captured on camera. Photos and videos have one major shortcoming however, they reduce the 3D scene into 2D representations by perspective projection. Most of the geometric information is lost in this step. Attempts have been made to overcome this issue. Virtual walks through panorama images, most prominently known from Google Street View but also used to promote hotels or tourist attractions on their respective websites, feign three-dimensionality without an actual enhancement. The logical consequence was a 3D view option, that is now also available for many areas on Google Maps.

Likewise, the entertainment industry has discovered the value of three-dimensionality. 3D movies can be watched not only in movie theaters but also on one's own television. The Microsoft Kinect, a 3D camera, was developed for the Xbox video game console to track the player's movement and has raised the attention of 3D measurement technologies in the consumer market. The Occulus Rift was the pioneer of virtual reality glasses. Projecting different images for the left and right eye onto the integrated screens of these glasses invokes the perception that the user is moving through a 3D scene.

To fulfill the desire for 3D data some smartphones are now equipped with 3D cameras or applications that allow the user to recover the 3D geometry from videos. While most of this development is entertainment-driven, there are also some practical applications. Apart from improving maps for navigation, planning of furniture arrangements is a common topic in this field. While 3D perception allows to view the scene from different perspectives, the accuracy of the 3D model is of minor importance in these applications.

In other areas, such as the building industry, in archaeology or geology, however, precise measurements are necessary. This is where 3D laser scanning technology has become the method of choice. Current laser scanners measure long distances at impressive rates and thus capture the environment precisely using light in the infrared range. But optical devices are not restricted to measuring distance and visible light. Thermal cameras relate radiation in the thermal infrared range to the surface temperature of the object emitting the radiation. Within the past few years non-contact temperature measurement devices have developed from single-point measurement

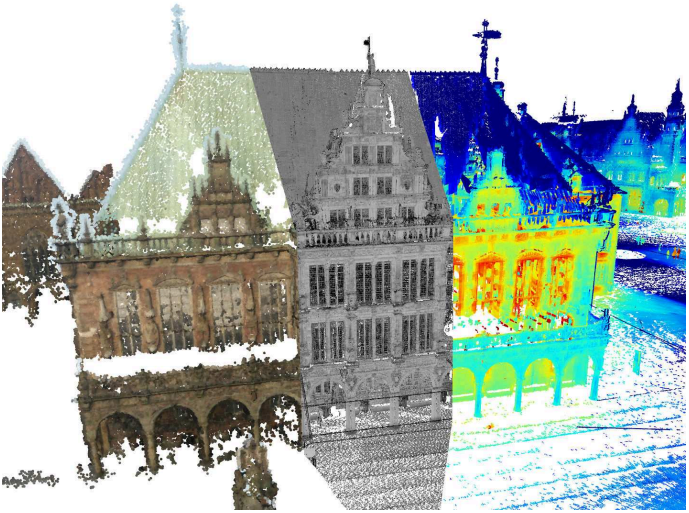


Fig. 1.1: The city hall of Bremen. Sketch of a multi-modal model, a point cloud enhanced with temperature and color information.

devices to sensor arrays with increasing resolution while decreasing in size and weight.

It is hard to imagine living in a building without electricity and a heating or cooling system these days. Heat and air conditioning losses in buildings lead to a large waste of the limited energy resources and pollute the environment unnecessarily. Factories and data centers are equally dependent on a continuous functioning of these systems. As beneficial as this development is for our daily life, the consequences of a failure are critical. Malfunctioning power supplies or temperature regulation systems can cause the close-down of an entire factory or data center. To detect these flaws as quickly as possible and to prevent the negative consequences, constant monitoring of power lines and heat sources is necessary. This is where thermal cameras come into play. But the analysis of thermographic data is tedious and requires expert knowledge. This thesis aims at changing the way thermography is performed by exploiting 3D sensing technology. It examines methods to combine the information from different optical sensors automatically, i.e., 3D geometry, color and temperature information as exemplified in Fig. 1.1.

A system that automatically creates a full 3D thermal model of an environment would be a big step towards the monitoring and inspection of existing buildings and technical assets as well as towards achieving energy efficiency in building construction. For example, data centers and factories rely on correct functioning of their infrastructure. Damaged pipes and cables or other parts endanger their functionality, cause pauses in the work flow and may lead to

harmful fires. In many cases leaks and overheating could be detected in their early stages by use of thermography thus preventing further damage. Building construction has undergone major changes in recent years. The importance of energy efficiency has attracted notice. To meet the Passivhaus, the Zero-energy building, or even the Energy-plus building standard modern building design makes use of all available heat sources including electrical equipment or even the body heat from people and animals inside the building. While this leads to changes in the way buildings are designed it also poses the question how existing buildings can be modified to meet these standards and to eliminate heat and air conditioning losses. According to the Action Plan for Energy Efficiency [48] of the European Commission the largest and cost-effective energy savings potential lies in residential ($\approx 27\%$) and commercial ($\approx 30\%$) buildings. The system proposed in this thesis is meant to aid in reaching these savings.

1.1 Best practices in thermography

The current state of the art for analyzing temperature related issues is thermal imaging. Fouad and Richter present a guideline for thermography in the building industry in Germany [77]. To detect thermal bridges of exterior building walls outdoor thermography is commonly used. Thermal bridges lead to a loss of energy and can cause humidity and mold growth. Only a few images are necessary to capture the entire building but their resolution is limited. To detect flaws in the construction a difference of 15 Kelvin is necessary between indoor and outdoor temperature to come to significant conclusions. It is desired that the weather conditions remain stable over a longer period of time, making the morning hours in the winter months ideal.

Keeping stable conditions is easier to achieve for indoor thermography. The analysis of back-ventilated walls and roofs is only possible from indoors. Thermal bridges at exterior walls and interior walls connecting heated and unheated rooms, pillars that interrupt the thermal insulation of a building, air leaks at windows and doors and the moisture penetration at basement walls are common applications for indoor thermography that focus on energy efficiency in existing buildings. Other applications aim at documenting and examining the run of heating pipes, detecting blocked pipes and construction units in a building to eliminate flaws, to make room for improvements and, in some cases, to ensure safety. For new buildings or the energetic restoration of existing buildings it has also become common to perform thermography before, during and after the construction phase for quality management.

Monitoring and analysis of the data for a large building is tedious. For indoor thermography the room should be prepared at least six hours before the inspection to achieve best results. A constant temperature is desired for this period. Furniture has to be moved away from the walls to allow their inspection. The difference between indoor and outdoor temperature has to be at least 10 °C. During the inspection each room is examined with the thermal camera. For each picture the inspector has to note the exact position and orientation from where the picture was taken [75]. After the inspection the images have to be analyzed taking into account the room temperature, the humidity, the material of the wall and the angle from which objects are seen. For some applications, e.g., inspection during construction or renovation, it is also necessary that the changes are documented over time, thus asking for comparability between independently acquired data sets [77].

1.2 Approaches towards 3D thermal modeling

Thermal images document the precise temperatures without any spatial dimensions. To identify a heat source and measure its extent it is necessary that the expert analyzes the scene on-site which is time-consuming and in some cases even dangerous. For applications that require repeated thermography over time, comparison of the 2D data is only possible to some extent. Successfully modifying a building with respect to thermal issues involves extensive planning. This planning would greatly be improved by the existence of a geometrically correct 3D thermal model. The full 3D model makes it easy to identify the location of each picture as it is shown within its surroundings and its position is known. This is especially important since indoor photos capture only a small part of the scene. Automatic registration, as known from robotics, enables the merging of two models acquired at different times. According to Fouad and Richter [77] thermography can only be an auxiliary device. The analysis of the scene is only possible through expertise and experience. By automatically pointing out regions of interest in the data our system helps to find damages quickly during the analysis.

The use of such data is manifold and finds application in many areas. The main idea is to co-calibrate different sensor modalities and to employ a mobile robot to explore the area autonomously. For a robot to move around in its environment and to gather data it needs to localize itself. Without the availability of a map the robot can only rely on its own information and create its own local map. This problem is known as Simultaneous Localization and Mapping (SLAM) and constitutes the basic problem for each mobile robot. In this work the SLAM problem is two-fold. The robot needs to find its way for localization, but more importantly the gathered sensor information needs to be joined with high accuracy. For the latter a GraphSLAM methods was developed and is evaluated in this thesis. A sophisticated exploration strategy is applied. Instead of exploring solely based on the floor plan of the building, the system tries to find positions from where the entire 3D space can be seen, reducing the amount of occlusions in the captured data. To automate the calibration a special calibration pattern is developed that is reliably detected in the point cloud and the camera images. Last, several applications are presented that build upon the acquired and registered data, or are designed to improve it.

1.3 Outline

This thesis is structured as follows. Chapter 2 introduces the hardware used for data acquisition. The automation process in this work includes the use of mobile robots. A short paragraph is dedicated to describing the setup of the different mobile robots. The sensor technology that acquires the data that is the foundation of this work is based on optical measurements. Therefore an introduction to optics is presented that provides the basis for the different sensors. The technologies for optical imaging, thermal imaging and 3D range sensing are described. Chapter 3 gives an overview over the data sets used throughout this thesis. It includes a description of the hardware setup, the location and the acquisition procedure.

Basic point cloud processing methods are introduced in Chapter 4. Apart from data structures to store and access the data efficiently, namely octrees, kD-trees and panorama images, feature detection methods are presented. Panorama images allow the use of image features

known from computer vision. The most prominent 3D features in man-made environments are planar structures. Accordingly, they are the most elementary features to be extracted from 3D point cloud data. Three classes of plane detection methods are explained and compared. When acquiring a 3D point cloud only the surface of the environment visible from that viewpoint is captured. For a 3D model of the entire scene several scanning positions are necessary that have to be brought into one common coordinate system. This process called registration is explained in Chapter 5. The well-known iterative closest point algorithm (ICP) is used in this work in combination with a global optimization. A comparison of different methods for generating the pose estimates required by ICP is performed.

For automating the data acquisition using a mobile robot it is essential to implement an exploration strategy. Chapter 6 presents and demonstrates a next best view strategy that takes into account the 3D geometry of the scene aiming at achieving a complete 3D model. The key for combining data from different sensors is the calibration, i.e., the determination of the transformation between the individual sensors. For this thesis a calibration procedure was developed that combines the intrinsic calibration for thermal and color cameras with an automatic detection of the calibration pattern in the 3D point cloud resulting in the extrinsic calibration between scanner and cameras. After calibration the image data is mapped onto the point cloud under consideration of occlusions. This procedure is explained and evaluated in Chapter 7.

Chapter 8 presents applications and extensions to the aforementioned methods. A Virtual Reality system is introduced that projects the temperature information into the scene for inspection purposes. The plane detection is used to determine the structural elements of building interiors. This is further elaborated by detecting and classifying windows in a 3D thermal model. Furthermore, methods for radiometric alignment of 3D point clouds and for automatic coloring of laser scanner data based on models generated from photos are presented. Finally, Chapter 9 concludes the thesis by summarizing and discussing the main results and presenting an outlook on open research topics following this work.

1.4 Scientific contribution

With the recent developments in 3D sensing technology and the accompanying cost reduction 3D point cloud processing has moved into the focus of many application fields including archaeology and cultural heritage, geology, and the building and entertainment industries. With the availability of faster sensors comes the need to process the data efficiently. This thesis is the result of research in the field of robotics with a focus on laser scanning and 3D point cloud processing and presents the effort to combine 3D point cloud data with other modalities such as images acquired by photo cameras and temperature information acquired by thermography and preparing the data to be used in various application areas. Many of the intermediate results adding to the entity of this thesis have previously been published and demonstrate the scientific contribution, which can be summarized in the following categories:

3D data structures. The efficient storage of 3D point cloud data is the key to 3D point cloud processing. The ability to handle large amounts of data transforms 3D point clouds from mere point data into meaningful representations of the environment. The efficient implementations of the data structures, namely kD-trees, octrees and panorama images

(cf. Chapter 4.1), are used in this work for various tasks and were previously published in [63, 67, 69, 106, 107].

Plane detection. Planes are the most common structures in man-made environments. To discover them in noisy 3D data is challenging (cf. Chapter 4.3). The developed data structure from [28] allows to employ the Hough Transform (HT) for plane detection. In the evaluation in [32] it was shown to outperform alternative methods for detecting the major structures in buildings, laying the foundation for the reconstruction of building interiors in [59] (cf. Chapter 8.1). For the detection of smaller planar structures such as calibration patterns (cf. Chapter 7) a region growing approach (RG) is adapted.

Registration. In [30,152] a GraphSLAM method to calculate six degree of freedom (DoF) poses (position and orientation) to solve the SLAM problem (cf. Chapter 5) was proposed. It is still one of very few methods that attempt to achieve globally consistent maps of 3D point clouds on a large scale. Evaluations of the approach can be found in [27,29,33,86,125]. The method was later extended to also handle data from mobile and personal laser scanning platforms [65, 68, 123].

3D exploration. The mobile robot Irma3D [150] was developed in 2009 to combine the research from laser scanning and robotics (cf. Section2.1). The intention of this development was to enable a robot to perform surveying tasks taking the effort of repositioning the heavy equipment and waiting for the data collection and the risk when operating in hazardous areas off the operator. For complete autonomy the robot needs to calculate the next scanning positions by itself (cf. Chapter 6). A 2D approach is presented in [38] that is extended to a novel next best view algorithm in [39], taking into account the full 3D environment and the sensor constraints.

Thermal modeling. With the increasing awareness of energy efficiency thermography has become an important tool in as-built documentation. As thermal images are hard to interpret there have been several attempts to create 3D thermal models. A unique approach to solve the problem with a high degree of autonomy by using laser scanners was presented in [24, 31, 38] (cf. Chapter 7).

Applications. The fundamental research in 3D point cloud processing was shown to be used for various applications in different fields [34–37, 53, 54, 59, 124].

Chapter 2

Sensor hardware technology

This thesis deals with the acquisition of data from different optical sensors, the combination into one common model and applications thereof. Although the main focus lies on the development of a technology that acquires a 3D thermal model autonomously, the research in this thesis is extended to the acquisition of color data as well. Not only thermal models benefit from the additional modality but this allows to extend the application fields.

Two major tasks arise: the autonomous data acquisition and the combination of the data into one common model. This chapter introduces the hardware used to accomplish the first task. To achieve autonomy a mobile robot is employed. The next section introduces briefly the robots used for the experiments. Three types of data are desired as sketched in Fig. 2.1, thermal information, a color photo and a 3D point cloud. As the hardware used to capture this kind of data relies on optical sensors, an introduction to the concepts behind the technology is given in Section 2.2, followed by the explanation of thermography. The chapter concludes by outlining the technology for acquiring 3D point cloud data, namely laser scanning and triangulation.

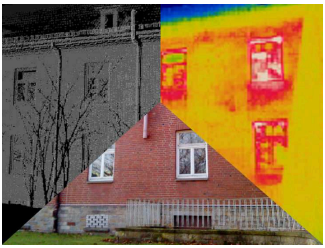


Fig. 2.1: A photo, a thermal image and a point cloud.

2.1 Robot

The Oxford Advanced Learner's Dictionary defines a robot as "a machine that can do some tasks that a human can do and that works automatically or is controlled by a computer" [102]. Following this definition the beginnings of robotics can be traced as far back as several centuries B.C. Back then mechanical automats were invented with the purpose of impressing humans. Thereafter automats were created for various purposes, many purely for entertainment but also various automats designed to take over tasks for humans.

According to the more explicit definition in Collins Cobuild Dictionary, "a robot is a machine which is programmed to move and perform certain tasks automatically" [99]. The burst of what

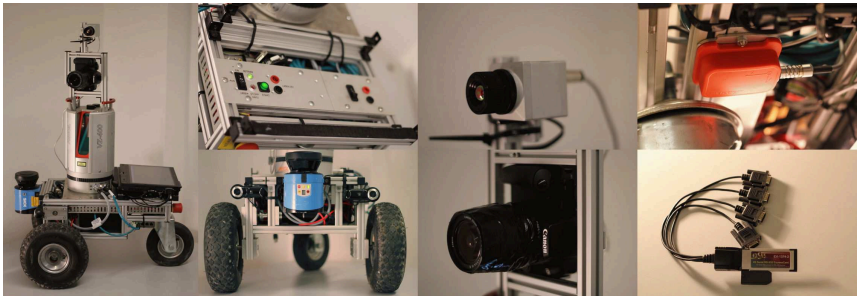


Fig. 2.2: The mobile robot Irma3D. From left to right: side view of the robot with all its sensors; Above: The control panel with switches for the two electric circuits and sockets for charging the batteries. The laptop mount fixes a laptop above the control panel. Below: Front view showing the 2D laser scanner and two small digital cameras that are attached to the robot for easier navigation; The thermal camera (above) and a DSLR camera (below) that are mounted on top of the laser scanner to sense thermal properties and color information of the environment to enhance the point clouds; Above: An IMU is mounted underneath the chassis next to the rear wheel. The position provides maximum shielding from the magnetic fields generated by the motors and the 3D laser scanner. Below: An ExpressCard to RS-232 Adapter is used to connect modern laptops without serial ports to the Volksbot motor controller and the IMU. (Source [150])

is considered a robot by this definition is marked by the introduction of robots into industry. Developed in the late 1950s after a patent in 1954 by George Devol and Josef Engelberger the robot Unimate was installed by General Motors [7] and from there on robots continuously took over many tasks not only in the automobile industry but also in other branches, often used for assembly, welding and other tasks that are performed repeatedly. Common tasks taken over by robots are those that require strength, high precision or put a worker in danger. Over the last decades the focus of robotic research has moved towards mobile robots. Compared to the common industrial robots mobile robots have one main difference, namely their environment. While in industrial settings the environment is mostly stable, mobile robots have to deal with dynamic changes in the environment and, more importantly, often have to face unknown environments. The purpose of mobile robots, however, remains the same, to take over tedious or potentially dangerous jobs. Mobile robots are typically understood as machines that move on land. They can be categorized by their means of locomotion, e.g., wheeled, tracked, or legged mobile robots.

In this thesis the use of a wheeled mobile robot for mapping applications using various sensors is evaluated. Laser scanning is applied where precise 3D measurements of the environment are required. This includes the modeling of building interiors during the construction or for reconstruction, of historic sites for documention of the current state, of entire cities for planning purposes, underground mines for economic and safety reasons [86], or of geologic areas for surveying tasks. Robots can help by releasing the worker of the need to carry the heavy equipment or even replace rescue workers in desaster areas to gather information about the situation without putting the rescue worker at risk. For this purpose, the robot Irma3D (*Intelligent robot*

for mapping applications in 3D) was developed [150]. Irma3D is a small battery powered three wheeled robot (cf. Fig. 2.2). The base is a Volksbot-RT3 chassis with a length of 58 cm, a width of 52 cm, a height of 31.5 cm and a weight of approximately 25 kg. The large weight comes mainly from the four 12 V 9.0 Ah lead-acid batteries that weigh about 2.5 kg each and provide the power for the robot and its sensor components. The two pneumatic front wheels with a diameter of 26 cm are each actuated by a 150 W DC Maxon motor. The back wheel is a swivel-mounted pneumatic castor wheel with a diameter of 20 cm, that follows the direction of the front wheels. The Maxon motors allow the robot to move at a maximum speed of 2.2 m/s. They are equipped with rotary encoders to measure the wheel rotations. Thus the movement of the robot can be tracked using odometry. The resulting pose estimates are enhanced by measurements from the Xsens MTi IMU device that is also attached to the robotic platform. The robot is controlled via an interchangeable Laptop that is mounted on the back. Currently it operates on a Samsung Q45 laptop with an Intel Core 2 Duo T7100 processor and 4 GB of RAM. The laptop runs ROS (the robot operating system [190]) to connect all components. Currently ROS Indigo is used. ROS is a middleware that provides the hardware abstraction, many of the device drivers and, most importantly, the message passing between the various components. To control the robot ROS communicates with the embedded Volksbot Motor Controller (VMC) that consists of a C167 micro controller to act as a PID controller for the two wheels and to process odometry data.

In addition to this minimum setup for the robot Irma3D several sensor components can be added when needed. For remote controlling the robot two Logitech QuickCam Pro 9000 webcams are attached to the front of the chassis. For obstacle detection when driving autonomously a SICK LMS100 laser scanner is mounted at the front of the chassis. This forward facing 2D laser scanner operates at a rate of 50 Hz and measures the distance to obstacles within its 270° field of view at a resolution of 0.5°. It can be used to create a floor plan of the environment and to improve navigation. The main sensor of Irma3D is a Riegl VZ-400 laser scanner. This sensor is mainly used for data collection and mapping. It creates a 3D point cloud representing the environment. This point cloud can be enhanced with data from cameras that are mounted on top of the scanner. For this purpose, different setups have been used consisting of different combinations of a Logitech QuickCam Pro 9000 webcam, a Canon EOS D1000 DSLR camera and an Optris PI160/400 thermal camera. These sensors will be explained in more detail in the following section. The robot can be operated in autonomous mode or be remote controlled via a Logitech Wireless Gamepad F710 or using a graphical user interface on a remote computer. The graphical user interface supports the visualization of the two Logitech webcams, the 2D laser scanner and the 3D laser scanner as well as navigation and the start of a 3D laser scan.

A low-cost alternative to a terrestrial laser scanner for 3D environment mapping often used in robotics uses actuated 2D laser scanners. The mobile robot Kurt3D Outdoor (Fig. 2.3(a)) from the University of Osnabrück has a size of 45 cm × 33 cm × 29 cm ($l \times w \times h$) and a weight of 22.6 kg. Two 90 W motors are used to power the six skid-steered wheels, where the front and rear wheels have no tread pattern to enhance rotating. The core of the robot is a Pentium-Centrino-1400 with 768 MB RAM [30]. For 3D mapping it is equipped with a SICK LMS200 2D laser scanner that has an opening angle of 180° at a resolution between 1° and 0.25°. For collecting 3D laser scans the laser scanner is fixed on a mount that is rotated around the horizontal axis with a servo motor to achieve a vertical opening angle close to 90°. An alternate setup is employed for the mobile robot Erika developed at the Lebniz University Hannover where two SICK LMS2XX

(a) Kurt3D (b) Erika (Source: [208])

Fig. 2.3: Mobile robots with rotating 2D laser scanners.

laser scanners are rotated around the vertical axis (Fig. 2.3(b)). This setup allows 360° scanning. The mobile robot Erika with a size of 95 cm × 60 cm × 120 cm is built of the Modular Robotics Toolkit (MoRob-Kit). The internal battery is dimensioned to supply the differential drive motors and a 700 MHz embedded personal computer for a minimum of two hours or a distance of 5 km. Localization is achieved via odometry, a three-axis gyroscope, and a low-cost SiRF III GPS receiver [208].

2.1.1 Odometry

On mobile robots pose estimates are usually attained from odometry. Odometry for wheeled robots such as Irma3D is based on calculating the distance traveled by the robot based on wheel rotations. For this purpose the the count c of the wheel encoders and the wheel rotations are related to each other using a factor f. Knowing the diameter d of the tires the distance traveled by one wheel is calculated as $\Delta s = \pi \cdot d \cdot f \cdot c$. Considering the distance B between the two wheels and the distances traveled by each wheel $\Delta s_l, \Delta s_r$ the pose (x, y, θ) of the robot at time step n is calculated as:

$$\begin{aligned} \theta_n &= \theta_{n-1} + (\Delta s_r - \Delta s_l)/B \\ x_n &= x_{n-1} + 0.5 \cdot (\Delta s_r + \Delta s_l) \cdot \cos(-\theta_n) \\ y_n &= y_{n-1} - 0.5 \cdot (\Delta s_r + \Delta s_l) \cdot \sin(-\theta_n) \end{aligned}$$

The quality of these measurements depends highly on the behavior of the robot on the ground. If the floor is slippery and the wheels spin uncontrolled the estimates lack precision. To increase the precision of the position estimates the Xsens IMU is attached to the robot and the thus measured accelerations are integrated into the position estimates. This is especially helpful for rotations. Odometry works sufficiently when short distances are covered and the robot follows smooth trajectories. With increasing path lengths, many rotations and frequent stops along the trajectory errors sum up.

2.2 Optics and cameras

Optics is the science about light and vision, but it includes the studies about radiation that is not visible to the human eye, such as radiant heat, radio waves and X-ray [78]. All measurement devices used in this thesis work by use of radiant energy. To understand their concept it is important to get a general idea about how light travels through space. The explanations in this section can be found in most optics books. Here, the books by Freeman and Hull [78], Hecht [98] and Predotti et al. [163] were used for reference. Sections 2.2.1 to 2.2.6 follow [78] while Section 2.2.7 is based on [98].

2.2.1 Propagation of light

Most luminous sources emit light due to their high temperature. The heat corresponds to the atoms being in a state of agitation. The effects of these disturbances are transmitted into all directions as vibrations or waves that radiate outwards at high speed. At low temperatures these disturbances can be felt by the sense of touch in the nearer surroundings. Increasing temperatures lead to faster waves that are visible to the human eye. As an example, a cold dark metal emits no light. When heated up, first the temperature is detectable before it glows red, yellow and later white. The exact nature of the wave motions is not known but as a model commonly compared to the ripples occurring when dropping a stone into water. The characteristics of these waves are the velocity v at which they travel outwards, the distance between the wave crests, known as wavelength λ and the frequency f, i.e., the rate of rise and fall. These are related by

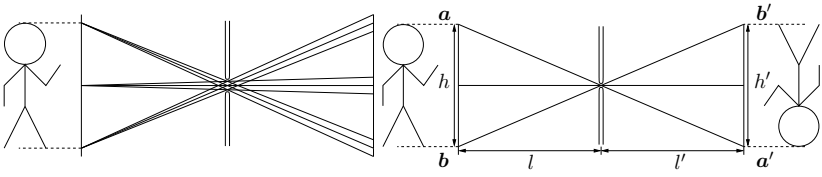
$$v = \lambda f \tag{2.1}$$

All optical waves in a vacuum share the common velocity $c = 299.792458 \cdot 10^6\,\mathrm{m/s}$, the speed of light, and vary only in frequency and wavelength. The example with the metal leads to the assumption that heat has a lower frequency than red light which has again a lower frequency than yellow and white light. From equation (2.1) follows that the wavelength is inversely proportional to frequency. Based on these characteristics the waves are classified into categories. The exact subdivision is not consistent in literature. Table 2.1 shows the approximated ranges and the relation between frequency and wavelength following the definition from [178]. The visible light covers only a very small part of the spectrum. It should be noted that what is perceived as white light is actually constituted of all colors equally. This was discovered by Newton. When focusing a narrow beam on a glass prism the white light is dispersed into its constituent colors, all the colors of the rainbow.

The wave motion is described by Huygen's principle. In a homogeneous isotropic optical media light spreads uniformly at the same speed in all direction. Assuming a point light source the light spreads in the form of a sphere at all times, the so called wave front. Due to the uniform spherical extension of the wave fronts each point of the wave travels in a straight line, the ray. Considering each point on the surface of the wave front as a new light source the light spreads again spherically in so called wavelets until the next wave front. The concept of wavelets explains the change of direction occurring when light enters a different medium or is reflected at a surface. Neither rays nor wavelets actually exist but they suffice as model to explain the behavior of light.

Table 2.1: Characteristics of radiation (based on [78, 178])

Radiation	Velocity v (m/s)	Frequency f (Hz)	Wavelength λ (μm)
X-rays	$300 \cdot 10^6$	$3 \cdot 10^{20} - 3 \cdot 10^{17}$	$1 \cdot 10^{-6} - 1 \cdot 10^{-3}$
Ultraviolet	$300 \cdot 10^6$	$3 \cdot 10^{17} - 789 \cdot 10^{12}$	$1 \cdot 10^{-3} - 0.38$
Visible Light	$300 \cdot 10^6$	$789 \cdot 10^{12} - 384 \cdot 10^{12}$	$0.38 - 0.78$
Blue	$300 \cdot 10^6$	$789 \cdot 10^{12}$	0.38
Green	$300 \cdot 10^6$	$545 \cdot 10^{12}$	0.55
Red	$300 \cdot 10^6$	$384 \cdot 10^{12}$	0.78
IR	$300 \cdot 10^6$	$384 \cdot 10^{12} - 0.3 \cdot 10^{12}$	$0.78 - 1 \cdot 10^3$
Radio	$300 \cdot 10^6$	$0.3 \cdot 10^8 - 0.3 \cdot 10^5$	$1 \cdot 10^3 - 1 \cdot 10^6$



(a) Extended hole (b) Pinhole

Fig. 2.4: Pinhole camera model. Reproduced based on [78].

When light from a point source passes through a small opening, in optics, especially photography, commonly called aperture, the small bundle of rays that pass through are diverging. There are no sharp edges, instead the waves bend slightly around the edges into the space behind the obstacle causing so called diffractions. If light from a larger light source passes through the small opening, each point from the light source causes a small bundle of rays which form together a beam.

2.2.2 The pinhole camera model

Cameras are based on the idea to capture the light that passes through a small opening, from now on called aperture. A pinhole camera supports the aforementioned model where light travels in straight lines. Given a small opening, as depicted in Fig. 2.4(a), each point of an illuminated object produces a bundle of rays that passes through the aperture and leaves a small patch of light on the image plane, a plane parallel to the plane of the opening. Each patch has the shape of the aperture resulting in a blurry image of the object that is inverted because of the straight lines. With a very small aperture (Fig. 2.4(b)), a pinhole, the overlap between the patches becomes so small that a fairly sharp image appears. The remaining blur is caused mainly by diffraction.

The mathematics of similar triangles in Fig. 2.4(b) show that the size of the projection

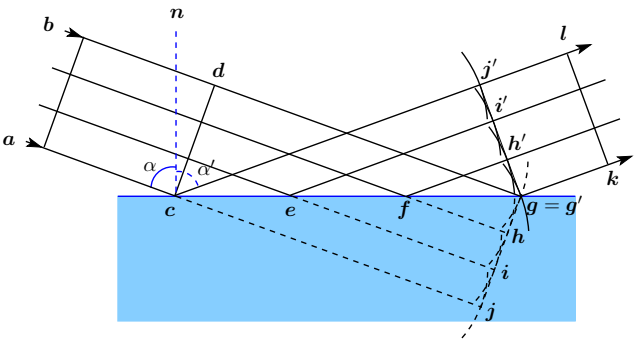


Fig. 2.5: Reflection of parallel rays on a plane surface. Reproduced based on [78].

depends on the distances of both the object l and the image plane l' to the center of projection:

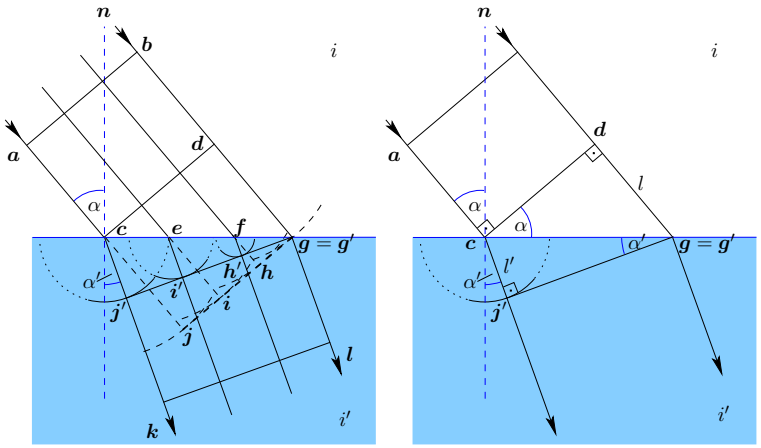
$$\frac{h'}{h} = \frac{l'}{l}.$$

The pinhole camera has no distortions and a long depth of focus, i.e., at greatly varying distances the sharpness remains the same. However, the small hole causes very low illumination as opposed to the larger aperture of lenses that allow more light to pass through.

2.2.3 Reflection and refraction

When light reaches a new medium, reflection, absorption and transmission come into effect. On most surfaces a combination of these effects happens. Some light is sent back, or reflected. Some light enters the new medium and is absorbed or transformed into another kind of energy while some part will continue its path through the new medium, i.e., is transmitted. A polished surface reflects almost all incoming light. It acts like an aperture except that it redirects the reflected beam into a certain direction instead of restricting it. This regular or specular reflection is demonstrated in Fig. 2.5 on parallel rays incident on the blue surface. The wave front $\overline{\boldsymbol{ab}}$ is propagated up to the new position $\overline{\boldsymbol{cd}}$. Here the first ray touches the polished surface with the incidence angle α between the straight line and the normal vector $\boldsymbol{n}$ of the surface. Instead of continuing its way towards $\boldsymbol{j}$ it is reflected at angle α' to propagate into the direction of $\boldsymbol{j}'$. Likewise the other rays each form a new wavelet with the center at $\boldsymbol{e}$, $\boldsymbol{f}$ and $\boldsymbol{g}$, and reflect the light towards the new wave front $\overline{\boldsymbol{g}'\boldsymbol{j}'}$. Given that the propagation speed remains the same and the rays are parallel, the new wave fronts form an angle of $90° - \alpha'$ with the surface as the light continues to travel in the direction of $\overline{\boldsymbol{kl}}$.

A non-polished surface causes irregular reflections where diffuse light spreads in all directions, meaning that the surface is visible everywhere. The amount of reflected light varies. On white paper about 80% of the incident light is reflected whereas black paper only reflects about 8%. If the irregularities of the surface are smaller than the wavelength of the incident light, the



(a) Refraction (b) Deriving α'

Fig. 2.6: Refraction on a plane surface. (a) Reproduced based on [78].

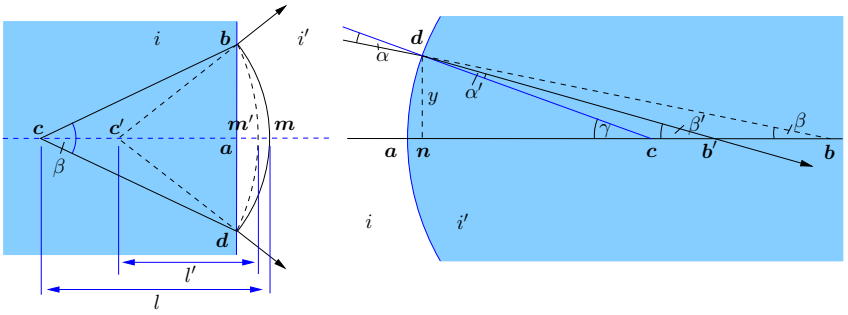
specularity grows. Consequently quite rough surfaces already reflect most of heat radiation and radio waves. The reflection of some materials is selective resulting in colored light being reflected by these materials. While some of the colors of white light are reflected others are absorbed by the material. When illuminated with green light, red objects appear black, as they only reflect the red light and absorb the other components of white light. Similarly some materials absorb light of a certain wavelength while they are transmissive for other wavelengths, e.g., protective glasses for ultraviolet or infrared light, that are however transparent to visible light.

Light that enters a new medium usually changes its direction. This process is called refraction. The refractive index that describes the amount of change is based on the change of speed at which the light travels through the old and the new medium. A vacuum is the medium with the highest velocity of light. The refractive index i of any medium is therefore calculated in relation to the vacuum

$$i = \frac{\text{velocity of light in vacuum}}{\text{velocity of light in medium}}.$$

Let v be the velocity of light in and i the refractive index of the first medium, α the incidence angle, i.e., the angle between the ray and the normal vector of a straight surface between the two media and v', i' the corresponding characteristics of the second medium then the angle of refraction stands in relation to these quantities by Snell's law as:

$$\frac{v}{v'} = \frac{i'}{i} = \frac{\sin\alpha}{\sin\alpha'}\,. \tag{2.2}$$



(a) Change in curvature. (b) Change in ray path.

Fig. 2.7: Effects of refraction reproduced based on [78]. (a) Change in curvature at a plane surface. (b) Change in ray path at a spherical surface.

The concept of refraction is sketched on parallel rays in Fig. 2.6. The wave front $\overline{\boldsymbol{ab}}$ is propagated through a medium with refractive index i up to the new position $\overline{\boldsymbol{cd}}$ where the first ray touches the surface to the medium with refractive index i'. The new medium has a higher density, i.e., i' is higher than i. The new wavelet with its center at $\boldsymbol{c}$ spreads at lower speed v' than the wavelet at $\boldsymbol{d}$. Instead of reaching $\boldsymbol{j}$ by covering the same distance as $\overline{\boldsymbol{dg}}$ the wavelet extends only to a radius $|\overline{\boldsymbol{cj'}}|$. Similar, the wavelets with the center at $\boldsymbol{e}$ and $\boldsymbol{f}$ when the corresponding rays enter the new medium extend to circles with smaller radii. By the time the light touches the new medium at $\boldsymbol{g}$ the new wave front $\overline{\boldsymbol{g'j'}}$ develops as the straight line from $\boldsymbol{g'}$ that is tangent to the wavelets with center at $\boldsymbol{c}$, $\boldsymbol{e}$ and $\boldsymbol{f}$ and continues in the direction of $\overline{\boldsymbol{kl}}$. This behavior is reflected in Fermat's principle of least time that states that when going from one point to another the actual path of light is the one that requires the least time under given conditions. It can be shown that the resulting refraction angle α' between the normal vector $\boldsymbol{n}$ and the parallel rays, e.g., $\overline{\boldsymbol{j'k}}$, fulfills equation (2.2) as illustrated in Fig. 2.6(b). According to the refractive index the distance l' traveled in the new medium relates to the distance l traveled in the first medium at the same time by $l' = l \cdot \frac{i}{i'}$. The sine of the incidence angle α and the refraction angle α' are therefore calculated and combined by substituting $\overline{\boldsymbol{cg}}$:

$$\begin{aligned}
\sin\alpha &= \frac{l}{\overline{\boldsymbol{cg}}} && \Leftrightarrow \overline{\boldsymbol{cg}} = \frac{l}{\sin\alpha} \\
\sin\alpha' &= \frac{l'}{\overline{\boldsymbol{cg}}} = \frac{li}{i' \cdot \overline{\boldsymbol{cg}}} && \Leftrightarrow \overline{\boldsymbol{cg}} = \frac{li}{i'\sin\alpha'} \\
\frac{l}{\sin\alpha} &= \frac{li}{i'\sin\alpha'} && \Leftrightarrow \frac{i}{i'} = \frac{\sin\alpha'}{\sin\alpha}.
\end{aligned}$$

The curvature R of a circle in reciprocal meters is calculated from the radius r by $R = \frac{1}{r}$. Refraction changes not only the direction of rays but also the curvature of wave fronts as illustrated in Fig. 2.7. Light is emitted from a point source $\boldsymbol{c}$ in an optical medium with refractive

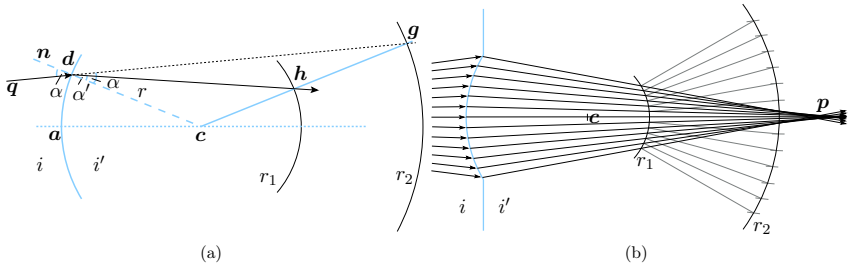


Fig. 2.8: Graphical construction of refraction on spherical surfaces reproduced based on [78]. (a) Construction of a single refracted ray. (b) Refraction of multiple rays coming from a distant point source.

index i. At the surface it enters an optical medium with lower refractive index i'. By the time the light reaches the points $\boldsymbol{b}$ and $\boldsymbol{d}$ in the first medium, it would have reached $\boldsymbol{m}$ if it had continued to travel at the same speed. However, by entering the medium with lower refractive index, it covers only the distance $\overline{\boldsymbol{am'}}$ at the same time it would have covered $\overline{\boldsymbol{am}}$. The new curvature converges to the point $\boldsymbol{c}'$, thus $R = \frac{1}{l}$ and $R' = \frac{1}{l'}$ are the curvatures of incident and refracted light. The angles and curvatures in the image are exaggerated for clarity. Assuming that β is small and thus the distances $\overline{\boldsymbol{am}}$ and $\overline{\boldsymbol{am'}}$ are small $L = \frac{i}{l}$ and $L' = \frac{i'}{l'}$ are the approximated vergences of incidence and refracted light and are measured in diopters. l is called the object distance while l' is the image distance. For plane surfaces $L = L'$ holds true, i.e., plane surfaces cause no change in vergence.

At a curved surface reflection and refraction take place in a similar manner. The surface normal of each individual point is the normal of the tangent plane at that point. Let r be the radius of the spherical surface $\overline{\boldsymbol{ad}}$ with its center of curvature at $\boldsymbol{c}$. The surface separates two optical media of refractive index i and refractive index i', where $i' > i$. Fig. 2.8 shows the graphical construction of the refracted ray. The ray starts from $\boldsymbol{q}$ and intersects the surface at $\boldsymbol{d}$ at an angle of incidence of α to the normal $\boldsymbol{n}$ that is the line connecting $\boldsymbol{c}$ and $\boldsymbol{d}$. Consider two circles with their center at $\boldsymbol{c}$ and the radii calculated based on the refractive indices as $r_1 = \frac{i'}{i}r$ and $r_2 = \frac{i}{i'}r$. $\boldsymbol{g}$ is the intersection of the extension of the ray $\overline{\boldsymbol{qd}}$ and the circle with radius r_2. The ray at $\boldsymbol{d}$ is then redirected in the new medium towards the point $\boldsymbol{h}$ that is the intersection of the line between the center of curvature $\boldsymbol{c}$ and the construction point $\boldsymbol{g}$ with the circle of radius r_1. Repeating this construction for several rays coming from a distant point source gives the result shown in Fig. 2.8(b). All rays are refracted towards the point $\boldsymbol{p}$, which lies on one line with the point source and the center of curvature $\boldsymbol{c}$, the principal ray. Speaking in terms of waves this means that a spherical surface with refractive index higher than the surroundings tends to make an incident wave front more convergent. However, one notices that rays that are distant from the principal ray are not deviated to the same point. This effect is called spherical abberation. It is overcome by limiting the aperture to the paraxial region, the region near the axis. This will be assumed in the following.

Based on Fig. 2.7(b) the change in vergence by refraction on a spherical surface is derived.

In the paraxial region α and α' are assumed to be small. Thus, according to the paraxial approximation the equation $i \sin\alpha = i' \sin\alpha'$ is approximated by $i\alpha = i'\alpha'$. Similarly, in the diagram $\tan\beta = \frac{y}{\overline{\boldsymbol{nb}}}$ is approximated by $\beta = \frac{y}{\overline{\boldsymbol{ab}}}$. The angles relate by $\alpha = (90° - \beta) - (90° - \gamma) = \gamma - \beta$. In the same manner $\beta' = \frac{y}{\overline{\boldsymbol{ab'}}}$ and $\gamma = \frac{y}{\overline{\boldsymbol{ac}}}$ are approximated and the refraction angle is described by $\alpha' = \gamma - \beta'$. Inserting these into the approximate law of refraction and reformulating gives

$$\begin{aligned} \frac{i'}{\overline{\boldsymbol{ab'}}} &= \frac{i}{\overline{\boldsymbol{ab}}} + \frac{i' - i}{\overline{\boldsymbol{ac}}} \\ \Leftrightarrow \frac{i'}{l'} &= \frac{i}{l} + \frac{i' - i}{r}, \end{aligned} \tag{2.3}$$

where r is the radius of the spherical surface, l is the object distance and l' is the image distance, similar to the example from Fig. 2.7(a). Substituting the definition of the vergences defined for plane surfaces and the curvature of the refracting surface yields

$$L' = L + \frac{i' - i}{r} = L + (i' - i)R = L + F. \tag{2.4}$$

This shows that a spherical refractive surface has the ability to change the incident vergence by the surface power or focal power

$$F = (i' - i)R.$$

resulting in the fundamental paraxial equation

$$L' = L + F.$$

From equation (2.4) follows that each object point $\boldsymbol{b}$ has one corresponding image point $\boldsymbol{b}'$. The distance of the image point can be calculated given the refractive index, the curvature and the object distance. As the path of light is reversible, the positions of object and image are interchangeable. A pair of corresponding image and object points is called conjugate points or conjugate foci. There exist two special points, the first and second principal foci. Given an object at infinite distance the rays are parallel and the incident wave front is planar, i.e., $L = 0$. Therefore, the refraction for positive refracting surfaces is given by $L' = \frac{i'}{f'} = F$ and the point $\boldsymbol{f}'$ on the optical axis at distance f' is the second principal focus. Assuming the image at an infinite distance, the refracting wave front is plane with $L' = 0$. Accordingly, the location where an object must be placed is calculated from $-L = \frac{-i}{f} = F$. The first principal focus $\boldsymbol{f}$ is located on the optical axis at a distance of the second focal length f to the surface. The planes passing through the principal foci perpendicular to the optical axis are known as focal planes. From the equations it is obvious that for a negative focal power the principal foci change sides.

2.2.4 Thin lenses

Using the concepts explained in the previous sections, the principle of imaging used in cameras is derived. Common cameras use spherical lenses for imaging. Each side of the lens has a curvature

which may be zero. The two radii may vary and be on the same or on opposite sides. The optical axis is the line connecting the two centers of curvature. The optical axis is normal to planar lens surfaces. An optical lens is centered if the the optical axis goes through the center of a circular lens. A lens with a refractive index higher than the surroundings makes any incident wave front more convergent. These are called positive lenses. All examples shown in the following use positive lenses. The equations however, hold also true for negative lenses. Lenses vary in shape and focal power. The basic equations are explained using the thin lens model, even though thin lenses are rarely used in practice. Lenses form an image in a different place by changing the vergence of light like two spherical refracting surfaces. For each surface the change in vergence is given through equation (2.3):

$$\frac{i'_1}{f'_1} = \frac{i_1}{f_1} + \frac{i'_1 - i_1}{r_1} \quad \text{and} \quad \frac{i'_2}{f'_2} = \frac{i_2}{f_2} + \frac{i'_2 - i_2}{r_2}. \tag{2.5}$$

For a thin lens the thickness of the lens is assumed to be zero. Therefore the intermediate image created at the first surface is the object for the second surface, i.e., the intermediate image distance l'_1 equals the object distance l_2 and both i_2 and i'_1 stand for the refractive index of the lens and are therefore identical. This allows to combine equation 2.5 to

$$\underbrace{\frac{i'_2}{f'_2}}_{L'_2} = \underbrace{\frac{i_1}{f_1}}_{L_1} + \underbrace{\frac{i'_1 - i_1}{r_1}}_{F_1} + \underbrace{\frac{i'_2 - i_2}{r_2}}_{F_2}.$$

The final refracting vergence L' of the lens equals the refracting vergence of the second surface and is the sum of the incident vergence L of the lens, i.e., the incident vergence of the first surface, and the total focal power F:

$$L' = L + F, \quad \text{with} \quad F = F_1 + F_2.$$

It is obvious that the same focal power can be achieved using different lens shapes resulting in changes in quality. For a thin lens in air, the refractive index becomes one. Thus, the focal power is expressed using the refractive index of the lens and the curvatures or the radii of the surfaces as

$$F = (n-1)\left(\frac{1}{r_1} - \frac{1}{r_2}\right) = (n-1)(R_1 - R_2)$$

and thus the equations for the principal foci become

$$L' = \frac{1}{f'} = F \quad \text{and} \quad L = \frac{1}{f} = -F.$$

The focal lengths f and f' have equal distance but opposite signs. For this case the relation between image and object distances becomes

$$\frac{1}{l'} = \frac{1}{l} + \frac{1}{f'}.$$

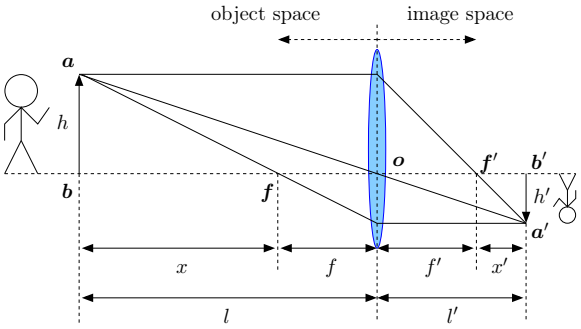


Fig. 2.9: Thin lens model. Reproduced based on [78].

Fig. 2.9 explains the geometry of image formation using a positive thin lens. The aperture is assumed to be limited to the paraxial region and the angles and heights are exaggerated to point out the effects. Given an object located at a distance of l from the lens, three points, the optical center $\boldsymbol{o}$ of the lens and the two principal foci $\boldsymbol{f}$ and $\boldsymbol{f'}$ and three rays emerging from point $\boldsymbol{a}$ and passing through these points are used to calculate the image distance l'. A ray parallel to the optical axis is refracted to cross the optical axis at the second principal focus. A ray passing through the first principal focus is refracted parallel to the optical axis. At the optical center the two surfaces of the lens are parallel and perpendicular to the optical axis. For a ray crossing the lens at this point, the lens appears as two parallel planes, thus due to the thinness assumption the ray passes without deviation. The intersection of these three rays gives the distance of the image plane l'. Two rays are sufficient to determine the distance, the third ray is only used for verification. With an ideal lens, all rays coming from point $\boldsymbol{a}$ intersect at point $\boldsymbol{a'}$ and form a sharp image and all points from point $\boldsymbol{b}$ intersect at point $\boldsymbol{b'}$ and form a sharp image. As for the pinhole camera model, the image is again mirrored. Using similar triangles the relations

$$m = \frac{h'}{h} = \frac{l'}{l} = \frac{x'}{f'} = \frac{f}{x}$$

are derived, where m is the magnification. For a negative lens divergence is added to the light so that a virtual image of the object is formed. The two principal foci switch sides, i.e., $\boldsymbol{f'}$ is closer to the object than $\boldsymbol{f}$. A virtual image appears to be in the object space. However, the image space is still to the right of the lens and the rays only exist there.

2.2.5 Thick lenses

The thin lens equations are approximations. Actual camera systems use lenses where the thickness d cannot be ignored. However, with a few considerations the equations developed for the thin lens assumption can still be used for thick lenses or even systems of lenses. Fig. 2.10 illustrates the idea of two virtual principal planes for determining the principal foci. Parallel rays coming from a distant object are refracted on the first and on the second surface before

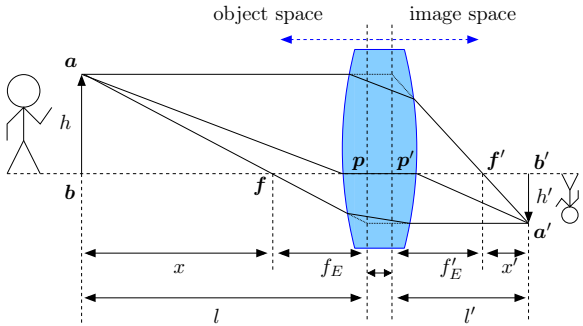


Fig. 2.10: Thick lens model. Based on [78].

being redirected to the second principal focus $\boldsymbol{f'}$. These two refractions can be simplified by one refraction, sketched as a dotted line in Fig. 2.10 on an imagined virtual plane, the second principal plane. The focal length f'_E of this virtual equivalent lens with thickness zero relates to the equivalent power F_E by $F_E = \frac{1}{f'_E}$. F_E is fixed by curvature, thickness and material of the lens, but the position may vary. The intersection of the second principal plane and the optical axis is called second principal point.

In the same manner the first principal plane is the virtual plane that represents the refraction taking place when an object is positioned at the first principal focus $\boldsymbol{f}$, resulting in a distant image. The first principal point is on the optical axis at a distance of the equivalent focal length f_E from the first principal focus at the intersection with the first principal plane. The actual distance between the surfaces and the principal foci are the second vertex focal length f'_v and the first vertex focal length f_v with their focal power $F'_v = \frac{1}{f'_v}$ and $F_v = \frac{-1}{f_v}$.

The focal powers and focal lengths are calculated from the refractive index i of the lens, the second focal length f'_1 and the focal power F_1 of the first surface, f'_2 and F_2 from the second surface and the distance d between the two surfaces:

$$F_E = F_1 + F_2 - \frac{d}{i}F_1F_2 \qquad f'_E = \frac{f'_1 f'_2}{f'_1 + f'_2 - (d/i)}$$

$$F_v = \frac{F_E}{1-(d/i)F_2} \qquad f_v = -f'_E\left(1 - \frac{d}{if'_2}\right)$$

$$F'_v = \frac{F_E}{1-(d/i)F_1} \qquad f'_v = f'_E\left(1 - \frac{d}{if'_1}\right).$$

Again the equations hold only true for the paraxial region. For the derivation reference is made to [78]. When using a thin lens, d is small, so that $-\frac{d}{i}F_1F_2$ can be ignored simplifying the equation for the total focal power to $F = F_1 + F_2$, as derived in the previous section.

Using the concept of similar triangles the fundamental paraxial equation

$$\frac{1}{l'} = \frac{1}{l} + \frac{1}{f_E} \tag{2.6}$$

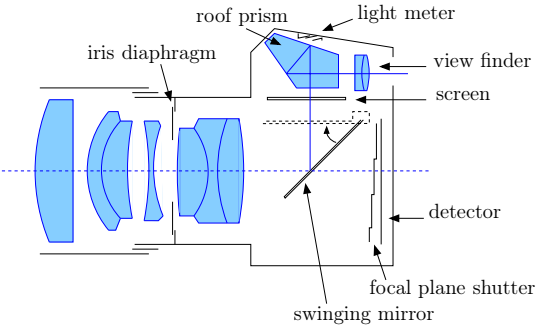


Fig. 2.11: Sketch of a lens system in a DSLR camera. Reproduced based on [78].

is shown to hold true for the equivalent focal length f_E and the magnification m has the same relations as for the thin lens model

$$m = \frac{h'}{h} = \frac{-f_E}{x} = \frac{-x'}{f'_E} = \frac{l'}{l}. \tag{2.7}$$

In a similar manner the equations for systems of multiple lenses can be traced back to the thin lens equation.

2.2.6 Principle of photo cameras

A digital camera uses a positive lens system to create an image on a flat plane containing an electronic detector. The setup of a digital single lens reflex camera (DSLR) is sketched in Fig. 2.11. To capture an image the focal plane shutter placed in front of the detector is opened for a short period of time to allow light from the lens to reach the detector. The mirror redirects the light through the roof prism to the view finder, where the user sees the image. Simultaneously, the light meter measures the amount of light. For the image capturing the mirror swings upwards letting the light pass to the detector instead. The iris diaphragm is an adjustable aperture stop that controls the amount of light and the depth of field. Generally, the aperture stop is either a lens or a metal plate with an aperture or in modern cameras an adjustable system consisting of a ring of blades that imposes the most limitations to the amount of light within the lens system. The purpose of an aperture stop is to let the most amount of light through without sacrificing image quality. The aperture stop is characterized by the entrance and the exit pupil. The entrance pupil E is the image of the aperture stop as seen by the object taking into account the lenses in front of the aperture stop and limits the number of rays that reach the image. With the same aperture stop the entrance pupil does not significantly change with varying object distances. The exit pupil E' is figuratively speaking the same, except for the image rays. Thus the lens images the entrance pupil onto the exit pupil and vice versa. Two rays are important for any object. First, the marginal ray is the outermost ray from the axis point that passes through the aperture stop and thus marks the size of the bundle that passes through the lens from an

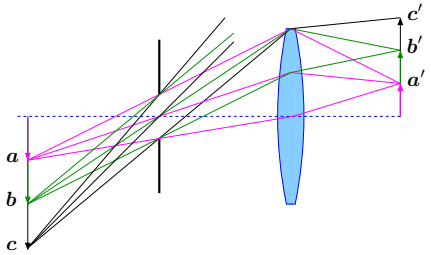


Fig. 2.12: Vignetting caused by the lens as effective field stop. Reproduced based on [78].

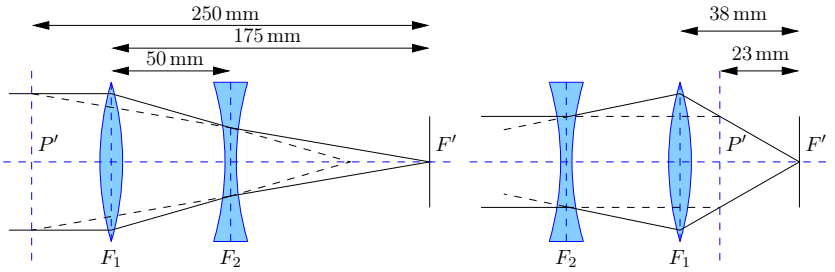
axial point. Second, the principal ray is the ray from an off-axis point that is directed towards the center of E and is refracted to pass through the center of the aperture stop. Together with the aperture stop the field stop limits the extent of the object the lens actually images, the so-called field of view (FOV). For off-axis points this means that, even though rays pass the aperture stop, the field stop might prevent them from reaching the image. This is depicted in Fig. 2.12 where the lens works as field stop. Here also the effect of optical vignetting is demonstrated. If the field stop is further away from the detector, the image brightness fades gradually. All rays from point $\boldsymbol{a}$ that pass through the aperture stop reach the image. The principal ray from point $\boldsymbol{b}$ passes through the top of the lens limiting the cone of light that reaches the image. For point $\boldsymbol{c}$ only the ray that touches the lower edge of the aperture reaches the image. Thus point a is the limiting view of full illumination while $\boldsymbol{c}$ is the outermost point that is still imaged and defines the total FOV.

In most cameras the aperture stop as well as the lenses are circular which means that the FOV must be guaranteed to cover the corners of the rectangular detector. Consequently the FOV depends on the size of the detector, the format, and the equivalent focal length f_E:

$$\mathrm{FOV} = 2\tan^{-1}\left(\frac{D/2}{f_E}\right),$$

where D is the length of the format diagonal. To allow comparison of devices with different formats instead of giving the actual focal length usually the 35 mm equivalent is given, referring to 35 mm photographic film. For a photo of a distant object a long focal length is required. With a focal length of 250 mm a person at 20 m fills the entire shorter length of a 35 mm × 24 mm film. To keep the size of the lens small, despite the required focal length of 250 mm, a telephoto lens combines a positive lens with a negative lens. The negative lens with its negative power reduces the back vertex focal length f'_v so that the lens only projects a distance significantly smaller than the effective focal length $f_E = 250\,\mathrm{mm}$ as depicted in Fig. 2.13(a) for two lenses with $F_1 = 10\,\mathrm{D}$ and $F_2 = -12\,\mathrm{D}$. Fig. 2.13(b) shows an inverted telephoto lens that is used to achieve a small focal length in cases where the f_E is smaller than the distance for the mirror needed between back vertex and the mirror.

In a zoom lens the distance between the positive and negative lenses is variable enabling a varying focal length. The challenging part is to control the aberrations in a way that they yield sufficient image quality over the entire range of the focal lengths.



(a) Telephoto lens (b) Inverted telephoto lens

Fig. 2.13: Telephoto lenses. Reproduced based on [78].

Table 2.2: Standardized series of f-numbers. Source [78].

f-number	$f/1.4$	$f/2$	$f/2.8$	$f/4$	$f/5.6$	$f/8$	$f/11$	$f/16$
Exposure time	1	2	4	8	16	32	64	128

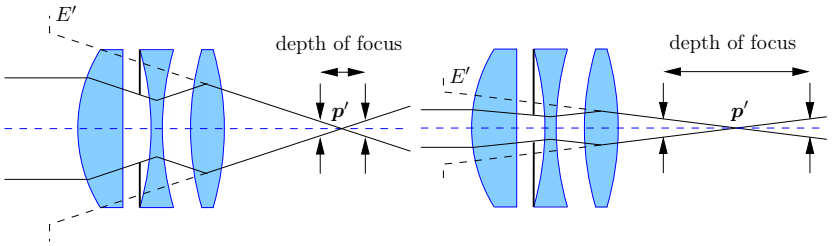
For the actual imaging process some further considerations are needed. For this the effect of the aperture is of importance. The aperture limits the amount of light that reaches the image. As rays do not exist, the amount of light is rather the size of the cone from a point source to the entrance pupil. For distant objects the brightness per unit depends also on the equivalent focal length. With an identical entrance pupil and a larger focal length a larger image receives the same amount of light, thus the brightness decreases. The image brightness is proportional to the square of the diameter of the entrance pupil and inversely proportional to the square of the equivalent focal length.

The aperture is described by the f-number, the inverse of the ratio of the equivalent focal length to the diameter of the entrance pupil D_{EP}:

$$D_{\mathrm{EP}} = \frac{f_{\mathrm{EQ}}}{k}. \tag{2.8}$$

A value of $f/8$ means that the equivalent focal length is $k = 8$ times the entrance pupil diameter. For variable aperture stops a series of f-numbers is implemented as shown in Table 2.2. The image brightness is given by the square of the f-number. Therefore each setting reduces the previous value by $\sqrt{2}$, thus giving half the brightness. The exposure time needs to be doubled to keep the brightness. For objects that are not at infinity adjustments depending on the focal length are necessary. With increasing image distance the aperture needs to be increased to maintain the effective f-number. Cameras with an integrated light meter compensate this so-called reciprocity failure.

A large aperture means shorter exposure times, but not all parts of the scene may be in focus due to different distances to the lens. The term depth of focus describes the area that is in focus in the image space while depth of field refers to the same problem in object space. Considering



(a) Wide aperture lens (b) Narrow aperture lens

Fig. 2.14: Depth of focus with different apertures. Reproduced based on [78].

an object at infinity, Fig. 2.14 shows the outermost parallel rays that pass through the aperture stop for two different aperture sizes. It is shown that the depth of focus, i.e., the range that falls below the maximum allowed blur is much wider for the narrow aperture, i.e., the possible range for the image positions is much larger. Seen from the object space, Fig. 2.15 illustrates the geometric derivation of the limits of the depth of field according to [163]. Here not the screen positions but the object positions are of interest. With the focal length f' the point at a distance of f_0 is projected sharply onto the image plane at a distance of f_0'. If the maximum allowed blur on the image plane has a diameter of p' there exists an object at the distance f_0 with a diameter p. According to the magnification equation (2.7)

$$\frac{p'}{p} = \frac{l_0'}{l_0}$$

holds true. From the fundamental paraxial equation (2.6) follows

$$\frac{l_0}{l_0'} = \frac{l_0}{l_0} + \frac{l_0}{f'} \Rightarrow \frac{p}{p'} = \frac{l_0 + f'}{f'} \Leftrightarrow p = p' \left(\frac{l_0 + f'}{f'} \right). \tag{2.9}$$

From similar triangles and substituting (2.9) and (2.8) follows:

$$\begin{aligned}
&\frac{l_f - l_0}{p} = \frac{l_f}{D_{\mathrm{EP}}} && \frac{l_0 - l_n}{p} = \frac{l_n}{D_{\mathrm{EP}}} \\
\Leftrightarrow\, &l_f = \frac{l_0 D_{\mathrm{EP}}}{D_{\mathrm{EP}} - p} && \Leftrightarrow l_n = \frac{l_0 D_{\mathrm{EP}}}{D_{\mathrm{EP}} + p} \\
\Leftrightarrow\, &l_f = \frac{l_0 \frac{f'}{k}}{\frac{f'}{k} - p' \left(\frac{l_0 + f'}{f'} \right)} && \Leftrightarrow l_n = \frac{l_0 \frac{f'}{k}}{\frac{f'}{k} + p' \left(\frac{l_0 + f'}{f'} \right)} \\
\Leftrightarrow\, &l_f = \frac{l_0 f'^2}{f'^2 - p' k (l_0 + f')} && \Leftrightarrow l_n = \frac{l_0 f'^2}{f'^2 + p' k (l_0 + f')} \\
\Leftrightarrow\, &\frac{1}{l_f} = \frac{1}{l_0} - \frac{p' k}{f'^2} - \frac{p' k}{l_0 f'} && \Leftrightarrow \frac{1}{l_n} = \frac{1}{l_0} + \frac{p' k}{f'^2} + \frac{p' k}{l_0 f'}.
\end{aligned}$$

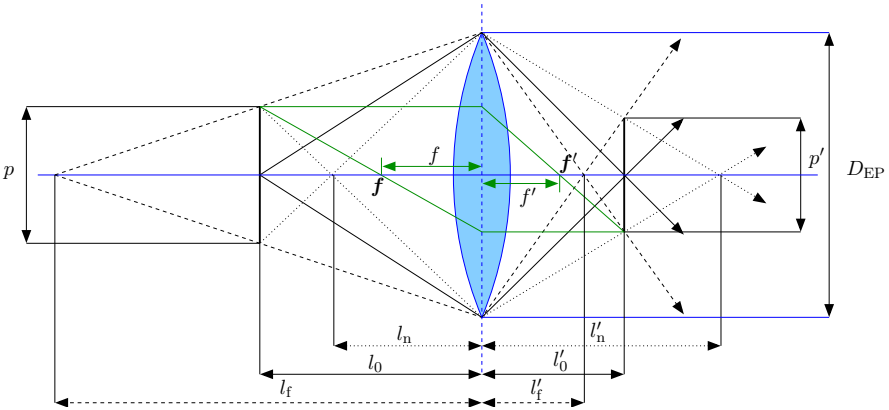


Fig. 2.15: Depth of field. Objects at a distance of l_0 are in focus, while the l_n gives the distance of the near plane and l_f the distance of the far plane the thresholds leading to the maximum allowed blur. Reproduced based on [163]. The green construction lines assert the distances of the object and image plane in focus.

Thus, when focusing a camera on an object at a distance of l_0 the range that is below the maximum allowed blur can be calculated. As l_0 is often large compared to the other values, an approximation omits the last term $\frac{p'k}{l_0 f'}$. The largest possible depth of field is achieved when the far distance is at infinity, i.e., when the lens is focused so that the maximum blur occurs at infinity. Evaluating $l_f \to \infty$ gives the corresponding distance at which an object is in focus:

$$l_0 = \frac{f'^2}{p'k} - f' \approx \frac{f'^2}{p'k}. \tag{2.10}$$

This distance is called the hyperfocal distance. An object at infinity has the maximum blur. Decreasing the object distance gives first sharper images until the hyperfocal distance is reached. After that decreasing the object distance leads to poorer images again until the maximum allowed blur at approximately $l_n = l_0/2$. [163]

2.2.7 Aberrations

All equations derived under the paraxial assumption are only approximations of the true underlying mathematics of optics. All derivations from these ideal conditions are called aberrations, that fall under two main categories [98]. Chromatic aberrations appear due to the fact that the refractive index is indeed a function of frequency or color, while monochromatic aberrations also occur with quasi-monochromatic light. Under the latter fall the five primary aberrations that can be grouped in two subgroups, aberrations that blur the image, such as spherical aberrations, coma and astigmatism as well as aberrations that deform the image, such as Petzval field curvature and distortions. Many of the aberrations caused by using spherical surfaces with finite

aperture outside of the paraxial region can be minimized by manipulating the power, shape, thickness, glass type, arrangement of lenses and locations of the stops.

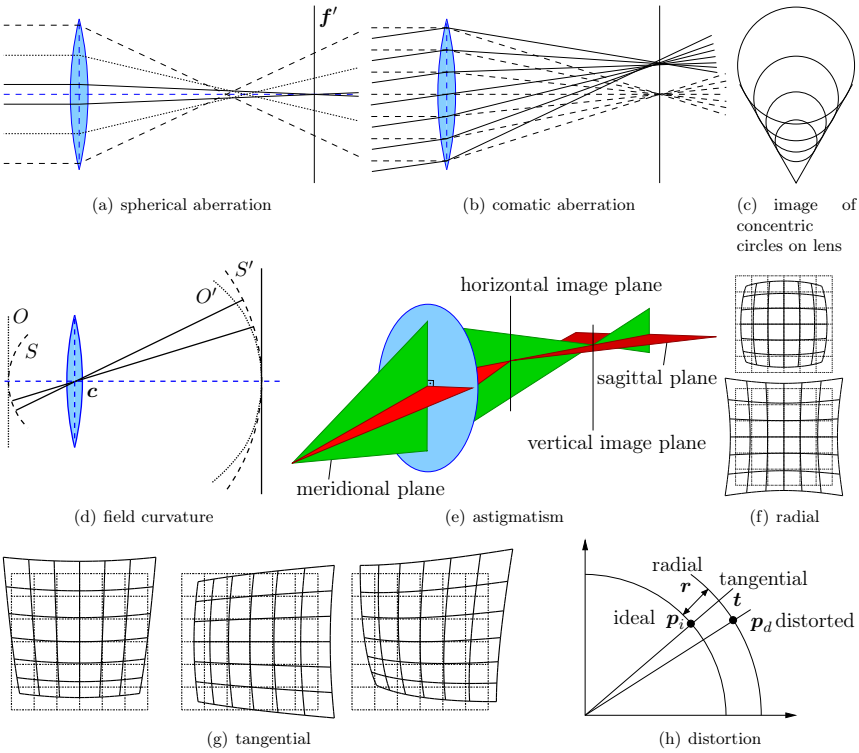
In the paraxial assumption the first-order theory was applied, where $\sin\varphi \approx \varphi$. The third-order theory gives an improved approximation, with $\sin\varphi \approx \varphi - \frac{\varphi^3}{3!}$. The difference between the third-order theory and the first-order theory is the sum of the five primary aberrations that are depicted in Fig. 2.16 and will be discussed in the following. The remaining difference to the results of exact ray tracing is the sum of higher order aberrations that are much smaller and will thus not be further considered here.

Spherical aberrations (Fig. 2.16(a)) are a direct cause from the curvature of the surface. As already pointed out in Fig. 2.8(b) that lead to the use of the paraxial assumption, rays at larger distance from the optical axis do not converge to the second principal focus due to approximations made for the equations derived from Fig. 2.7(b). For non-paraxial rays the focal length depends on the aperture leading to aberrations. For a converging lens marginal rays are bent too much and are focused in front of the paraxial rays. With growing distance to the optical axis the distance to the principal focus grows as well.

Comatic aberrations appear already for object points that are a small distance away from the axis. In the off-paraxial region the principal planes are rather curved surfaces leading to the effect visualized in Fig. 2.16(b). Considering a lens that is free of spherical aberrations a parallel bundle of rays (dashed lines) is refracted to focus in one point on the image plane. When rotating these rays slightly so that the bundle of rays becomes oblique the coma is evident. The effective focal length and magnification differs for rays traversing the off-axis region of the lens. As a consequence concentric circles on the lens are imaged in conic coma flair as depicted in Fig. 2.16(c) with a comet-like tail to which it owes its name. For a positive coma the distance of the comatic circle from the axis grows with the size of the circle on the lens. A large positive coma appears when using convex lenses while concave lenses cause a negative coma. [78, 98]

Given a lens without spherical and comatic aberrations one can assume a one to one relationship between object points and image points. In the paraxial region the image of a planar object that is perpendicular to the image axis is planar. For finite apertures, however, this assumption does not hold true anymore because the distance between object point and image point clearly grows for off-axis points, resulting in a curved surface. Given a spherical object S (cf. Fig. 2.16(d)) with the center of curvature at $\boldsymbol{c}$ given its image S' is again a spherical object with the center at $\boldsymbol{c}$ because off-axis points have a longer distance to cover. Moving each object point now towards the plane O means that also each image point is moved towards the lens resulting in the paraboloidal Petzval surface O'. For a positive refracting lens the Petzval surface is curved inward while for negative lenses the Petzval surface is curved outward. An appropriate combination of positive and negative lenses negates the field curvature.

Closely related to field curvature is the astigmatism that is based on the fact that oblique rays for off-axis points converge to different points in the tangential and sagittal plane causing two paraboloid image surfaces. This effect can be reduced through lens design. The tangential or meridional plane is spanned by the optical axis and the principal ray through the center of the aperture. The sagittal plane intersects the meridional plane perpendicular at the principal ray. It is actually not a single plane but changes slope when the chief ray changes direction at the surfaces of the lens system. For axial object points the cone of light is symmetrical. However, with growing angle between the optical axis and the principal ray, the difference between the

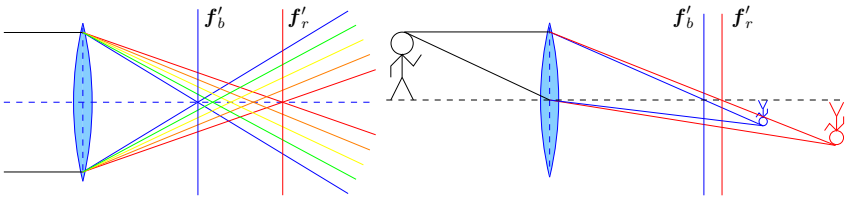


(a) spherical aberration (b) comatic aberration (c) image of concentric circles on lens

(d) field curvature (e) astigmatism (f) radial

(g) tangential (h) distortion

Fig. 2.16: The five primary aberrations. Reproduced based on [98] (a),(d),(f) [78] (b),(c), [143] (e) and [202] (h).



(a) Axial (b) Lateral

Fig. 2.17: Chromatic aberration. Reproduced based on [98]

focal lengths of a parallel bundle of rays increases. The focal lengths of the meridional rays decrease compared to the sagittal rays and thus the rays are tilted more with respect to the lens. This is demonstrated in Fig. 2.16(e). Somewhere between the horizontal and the vertical image planes there is a point where the cross section of the beam is circular. This point is the so-called circle of least confusion and the ideal image plane. The astigmatism leads to two paraboloidal image surfaces on the same side of the Petzval surface.

The last primary aberration is distortion. In lens systems imperfections in the manufacturing process lead to distortions that have both radial and tangential components. When deriving the magnification it was assumed to be identical over the entire image. In real systems, however, it is often a function of the off-axis image distance. This is due to the fact that different areas of a thick lens can have different focal length. This leads to distortions of the image at whole while each individual point is in focus. This concept is called radial distortion. The two types of radial distortion from the dashed original are depicted in Fig. 2.16(f). In the positive or pincushion distortion the off-axis magnification is greater than the optical axis magnification causing each point to move radially outward with growing distance. The negative distortion where the outer points move inward is also called barrel distortion. The direction of the distortion depends on the sign of the focal power, i.e., a positive lens suffers from positive radial distortion while a negative lens suffers from negative radial distortion, respectively. Apart from radial distortions tangential distortions occur in decentered lenses where the optical center of the lens is not identical to the mechanical axis. Tangential distortions are most easily envisioned by imagining an image plane that is slightly tilted with respect to the optical axis. The effects of tangential distortion are shown in Fig. 2.16(g), with examples of rotations around each of the axes individually and around both axes simultaneously. The combination of decentering effects as well as tilted elements within the lens system and the varying magnification leads to a point position $\boldsymbol{p}_d$ with distortion that differs from the ideal point position $\boldsymbol{p}_i$ as illustrated in Fig. 2.16(h) where both the radial distortion r and the tangential distortion t are indicated.

The five primary aberrations have an effect on the image quality even for monochromatic light. The last type of aberrations describes the effect of a lens on polychromatic light. Up to now the refractive index was considered constant for each lens. More precisely, however, the refractive index of a material is a function of the wavelength $i(\lambda)$. Thus rays of different wavelength, e.g., for a camera this corresponds to different color, are refracted differently and have different focal lengths as depicted in Fig. 2.17. Depending on the position of the image the edges of the blurred image appear in different colors. This effect is called axial chromatic aberration. For off-axis points the lateral chromatic aberration comes into effect. The different focal lengths based on the varying wavelength also cause a difference in transverse magnification. The different constituents of white light form thus a continuum of overlapping images that vary in size and color. As the human eye is most sensitive to yellow-green light the focus should be on the image of this color.

2.2.8 Radiometry

The focus of the previous sections is on the geometric image formation. This section is dedicated to the amount of electromagnetic radiation emitted by light sources and passing through an optical system, and thus to radiometry, the science of measurement of electromagnetic ra-

diation [78]. A point source emits a certain amount of radiant energy Q_e in J over time. The total amount of emitted energy is a fundamental quantity that is especially important when the amount of the output is not constant, as for pulsed lasers, or when the emitted energy needs to be quantified. The radiant flux ϕ_e describes the power of the source in J/s and thus the amount of energy radiated per unit time. An isotropic point source emits an identical amount of energy in all directions. For measuring the radiant energy, the interesting quantity is the light that is perceived by the sensor, thus the amount within a cone with the point source at its apex. The solid angle Ω in steradians (sr) describing this cone is the ratio between the area A of a detector and the distance r at which the detector is placed from the point source:

$$\Omega = \frac{A}{r^2}. \tag{2.11}$$

For sources that do not radiate uniformly in all directions the radiant flux per unit solid angle in a given direction gives the radiant intensity I_e in Wsr^{-1}:

$$I_e = \frac{\phi_e}{\Omega}. \tag{2.12}$$

Finally, the radiant flux per unit area of a detector at a distance r, namely the irradiance E_e that is in units of Wm^{-2} is expressed as:

$$E_e = \frac{\phi_e}{A}. \tag{2.13}$$

Substituting Ω from equation (2.11) and I_e from equation (2.12) yields the inverse square law:

$$E_e = \frac{I_e}{r^2}. \tag{2.14}$$

For a point source the irradiance is constant over a spherical surface. For a flat detector, however, the irradiance varies over the surface as light is no longer incident on the surface and the distance to the source changes. Fig. 2.18 illustrates the effects when a small patch on the sphere with area A is projected onto a flat surface at point $\boldsymbol{p}'$. The size of the area on the detector changes to

$$A' = \frac{A}{\cos\theta}, \tag{2.15}$$

where θ is the angle between the line passing through point $\boldsymbol{p}'$ and the normal of the flat surface passing through the point source $\boldsymbol{q}$. Combining equations (2.13) and (2.15) relates the irradiance E'_e at $\boldsymbol{p}'$ to the irradiance of the patch with size A tangent to the sphere by

$$E'_e = E_e \cos\theta.$$

Simultaneously the distance to the light source increases to

$$r' = \frac{r}{\cos\theta}$$

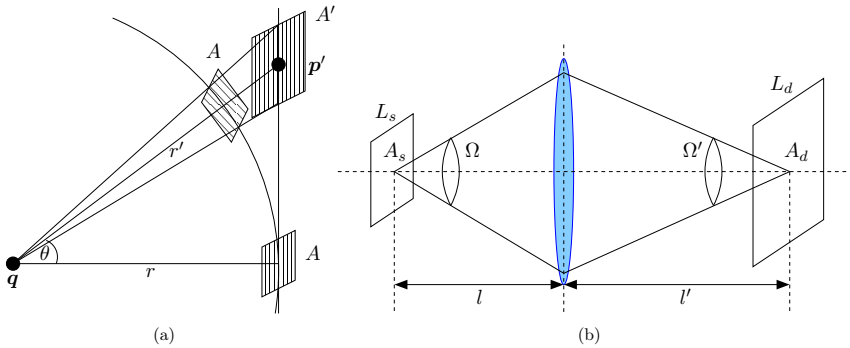


Fig. 2.18: (a) Variation of radiance for a flat surface at an incident angle θ. (b) Lens imaging an extended source. Reproduced based on [78].

and with equation (2.14) the $\cos^3$ law of radiometry is derived:

$$E'_e = E_e \cos^3 \theta.$$

Extending this concept to an extended source one needs to account for the entire source area and variations of the radiant intensity over this area. By considering the area as a collection of equally distributed point sources, the radiance L_e describes the radiant flux emitted into the unit solid angle per unit area of a source in a given direction in $\mathrm{W/m^2\,sr}$:

$$L_e = \frac{\phi_e}{A\Omega}. \tag{2.16}$$

Here A is the area of the source and Ω the solid angle subtended by the detector at the source. Viewing the source at an angle θ reduces the area to $A' = A\cos\theta$ and the point spacing in the respective direction becomes more dense yielding the equation

$$L_e(\theta) = \frac{L_e(0)}{\cos\theta}, \tag{2.17}$$

where $L_e(0)$ is the radiance along a known direction, typically normal to the source surface. This means that the radiance increases with growing angle θ, an implication that is rebutted by simple observation. For a Lambertian radiator the radiance is assumed to remain constant independent of the direction, thus

$$L_e(\theta) = L_e(0).$$

To compensate for decrease caused by the cos in equation (2.17) the radiant intensity is modeled inversely

$$I_e(\theta) = \frac{\phi}{\Omega}\cos\theta.$$

As practical sources often meet the Lambertian condition over a range of viewing angles the model is a good approximation.

When placing a screen at a distance r from the extended source a small patch of size dA of the source has an intensity of

$$I_e = \frac{\phi_e}{\Omega} = L_e dA$$

according to equations (2.12) and (2.16). Summing up all the small parts to cover the entire area A and inserting into the square law (2.14) gives an irradiance of

$$E_e = \frac{L_e A}{r^2} = L_e \Omega,$$

where Ω is the angle subtended by the detector from the source.

In linear optical systems radiance is conserved except usually negligible transmission losses. A simplified explanation is given using the positive thin lens depicted in Fig. 2.18(b). The lens is placed at a distance l from an extended light source and projects an image onto the detector at a distance l'. The subscript $_s$ denotes radiometric quantities related to the source while $_d$ denotes those related to the detector, respectively. With the solid angle $\Omega = A/l^2$ subtended by the lens with size A at the source the total radiant flux collected by the lens is derived from equation (2.16) to

$$\phi = A_s L_s \frac{A}{l^2}. \tag{2.18}$$

Similarly the radiance for the detector is given by

$$L_d = \frac{\phi}{\Omega' A_d}, \tag{2.19}$$

with $\Omega' = A/l'^2$. The area of the image relates to the area of the source given the magnification m by $A_d = A_s/m^2$. Combining equations (2.18) and (2.19) and canceling identical terms yields

$$L_d = L_s.$$

Based on equation (2.13) and substituting ϕ from equation (2.19) and Ω' the irradiance at the detector is given by

$$E_d = \frac{\phi}{A_d} = \frac{L_d \Omega' A_d}{A_d} = \frac{L_d \cdot \frac{A}{l'^2} A_d}{A_d} = L_d \frac{A}{l'^2}. \tag{2.20}$$

2.2.9 Vignetting

The vignetting effect, the darkening of an image towards the edges was already mentioned in Section 2.2.6. There are different factors that cause vignetting. The most important factor is the natural vignetting that is caused by the variation of the irradiance due to different viewing angles. In Fig. 2.18 the light source is on the optical axis. As seen before, deviations from the

perpendicularity lead to a fall off in irradiance. For an object at distance l to the lens an off-axis point $\boldsymbol{p}$ has a distance of $l_s = l/\cos\theta$ to the optical center of the lens. The lens has an area of $A = \pi\left(\frac{D}{2}\right)^2$ but seen from the source it is reduced to $A_s = A \cdot \cos\theta$. From equation (2.11) the solid angle Ω is given by

$$\Omega = \frac{A_s}{l_s^2} = \pi\left(\frac{D}{2}\right)^2 \cos\theta \left(\frac{\cos\theta}{l}\right)^2 = \frac{\pi}{4}\frac{D^2\cos^3\theta}{l^2}$$

When calculating the radiant flux according to equation (2.18) the reduction of the source area by a factor of $\cos\theta$ needs to be taken into account, yielding

$$\phi = A_s \cos\theta L_s \Omega = L_s \frac{\pi}{4}\cos^4\theta \frac{A_s}{l^2}.$$

Following equation (2.20) the irradiance is

$$E_d = \frac{\phi}{A_d} = L_s \frac{\pi}{4}\cos^4\theta \frac{A_s}{A_d l^2}.$$

Based on the magnification equation (2.7) the area of the source A_s and on the detector A_d and the squared object distance l^2 are substituted by the squared image distance l'^2:

$$\frac{A_s}{A_d} = \frac{1}{l'^2}.$$

For distant objects where $l \gg l'$ the fundamental paraxial equation shows that l' is approximated by the focal length f and using equation (2.8) the irradiance is expressed as:

$$E_d = L_s \cdot \frac{\pi}{4}\left(\frac{1}{k}\right)\cos^4\theta, \tag{2.21}$$

also known as the $\cos^4$ law. The term $\left(\frac{1}{k}\right)^2\cos^4\theta$ describes the vignetting effect. [78, 88, 198]

2.2.10 Color camera specifications

Used on Irma3D is a Canon EOS 1000D DSLR camera with a standard Canon EF-S 18-55 mm 1:3,5-5,6 IS II zoom lens [185]. The camera body has a size of 126.1 mm × 97.5 mm × 61.9 mm and weights 450 g. The camera has a 22.2 mm × 14.8 mm high-sensitivity high-resolution large single-plate CMOS image sensor with an effective pixel number of ≈ 10.10 megapixels. With an aspect ratio of 3:2 this means an image size of 3888 × 2592 pixels at maximum resolution. The 35 mm equivalent focal length is approximately 1.6 times the lens focal length. Data is stored in a 12-bit Canon original raw format using either the sRGB or the Adobe RGB color space. Within this thesis the former is used for all experiments. The electronically controlled focal-plane shutter allows for automatic shutter speeds ranging from 1/4000 s to 30 s. However, with full resolution the maximum image acquisition rate is 1.5 images per second. The camera has seven auto focus points but does not come with a distance indicator so that the distances need to be measured manually. For collecting photos without the robot a Canon EOS 1100D DSLR camera is used. It has a slightly larger body and an image size of 4272 × 2848 pixels.

Table 2.3: Hyperfocal distances in m of a Canon EOS 1000D camera with EF-S18-55 mm IS II lens [122]

	$f/4$	$f/4.5$	$f/5.6$	$f/8$	$f/10$
18 mm	4.26	3.79	3.05	2.13	1.71
24 mm	7.57	6.74	5.41	3.79	3.03
35 mm	-	14.33	11.51	8.06	6.45
55 mm	-	-	28.43	19.90	15.92

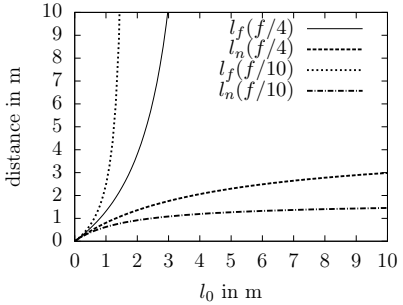


Fig. 2.19: Minimum and maximum distance for objects to cause the maximum allowed blur when focusing on l_0 using a Canon EOS 1000D camera with EF-S18-55 mm IS II lens.

The EF-S18-55 mm IS II lens consists of 11 elements in 9 groups. Adjusting the distance between these groups allows to adjust the focal length from 18 mm to 55 mm. The corresponding aperture limits range from $f/3.5$ to $f/22$ at 18 mm and from $f/5.6$ to $f/36$ at 55 mm. The lens has a diameter of ≈ 58 mm, maximum length of 70 mm and weighs 200 g. The closest focusing distance is at 25 cm from the image sensor plane. At this distance the field of view ranges from 207 mm × 134 mm at 18 mm focal length to 67 × 45 mm at 55 mm focal length. Table 2.3 gives the approximate hyperfocal distances for some of the zoom levels of the Canon EOS 1000D camera with the EF-S18-55 mm IS II lens calculated using the approximate equation (2.10). The maximum blur is defined by the pixel size that was experimentally evaluated to be 0.019 mm.

Fig. 2.19 illustrates the near and far focal distances when focusing on objects with varying range using the focal length of 18 mm. As neither the camera nor the lens come with a scale to set the focal distance, the only way to adjust the focus according to the necessary ranges is to focus on an object with known distance.

For experiments where the focus is on thermal data rather than on optical data a Logitech QuickCam Pro 9000 webcam is used with a field of view of 75° and a focal length of 2 mm. At the full resolution of 1600 × 1200 pixels the maximum video frame rate is 10 Hz. The photos for the Bremen City data set were captured with a Panasonic DMC-LS80 digital camera. The adjustable focal length ranges from 5.5 mm to 16.5 mm which corresponds to a equivalent focal length between 33 mm and 100 mm. The 1/2.5″ image sensor captures images up to 2048 × 1536

pixels. The two-step aperture varies between $f/2.8$ and $f/8$ for the wide angle and $f/5.1$ and $f/14$ in tele mode. [160]

2.3 Infrared cameras

Like the human eye photo cameras capture the visible light of a scene [178]. The visible spectrum is a small portion of the electromagnetic radiation. The radiation of invisible light covers not only a much larger part of the spectral band but also carries further information. Electric radiation is emitted from every body with a temperature above the absolute zero ($-273.15\,^{\circ}\mathrm{C} = 0\,\mathrm{K}$). This radiation is proportional to the bodies intrinsic temperature and can be observed with an optical measuring device similar to a photo camera. The part of the electromagnetic radiation used for this purpose lies in the infrared spectrum. The infrared radiation was discovered by William Herschel in 1800 [157]. His setup consisted of sun light led through a dispersive prism and a thermometer to measure the temperature along the colors of the spectrum. His discovery was not only the increase in temperature towards red but he measured the maximum far beyond the visible spectrum.

Fig. 2.20 shows the infrared spectrum within the electromagnetic spectrum. Enlarged are the areas with the most relevance for thermography. The visible spectrum, ranging from purple with a wavelength of $0.38\,\mu$m over blue, green, yellow, orange to red at $0.78\,\mu$m, is followed on the upper border by the infrared spectrum. The infrared spectrum is often divided into sub-ranges whose exact definitions vary in literature. Following the definitions from [178] four sub-divisions are made based on the applications. Near-infrared (NIR) between 0.78 and $3.0\,\mu$m is suitable for night-vision devices. Mid- (MIR) and long-wave (LIR) infrared, from 3.0 to $7.0\,\mu$m and 7.0 to $14.0\,\mu$m, respectively, comprise the thermal infrared range. The far-infrared range has little significance for thermography. The nature of thermal radiation allows imaging similar to cameras that capture the visible light (cf. Fig. 2.21). In thermography the emitted radiation penetrates the atmosphere and is focused by a special lens onto the detector of the sensor. A spectral filter selects the wavelength range relevant for the temperature measurement. The thus received signal is transformed into an electrical signal, amplified, processed and converted into a value representing the temperature [157, 178].

Next to pyrometers that measure the temperature of one spot precisely currently devices with increasing optical resolutions at decreasing prices are becoming available to perform thermography [178]. These devices partition the entire object scene into sequentially scanned pixels, thus allowing to capture several temperature values within a larger field of view at a given frame rate, yielding a temperature picture. Each pixel is mapped through a lens onto a CCD matrix where the electric charge for each pixel is aggregated and transmitted at video frame rate. Depending on their optomechanical scanning system these devices are categorized ranging from one-element detectors to full frame matrices. One-element detectors use an oscillating mirror and mirror polygons to project the object scene onto the receiver chip. The oscillating mirror movements limit the number of scanned pixels per time period. This drawback is compensated by the uniformity of the individual image pixels. Line scanning systems are independent of any mechanic motion unit. Consisting of a line of detectors they are a cost-efficient solution for surveying purposes. However, to capture a full scene the movement of the camera needs to be linked to the scene.

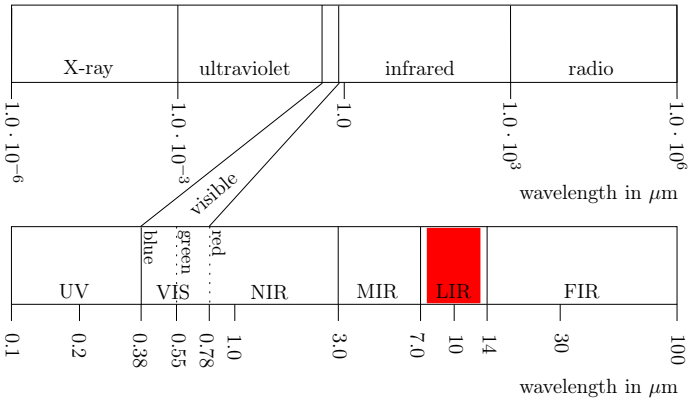


Fig. 2.20: Presentation of the infrared spectrum within the electromagnetic spectrum (reconstructed based on [178]).

To overcome this drawback various combinations of the two principles have been used, i.e., lines or matrices of detectors in combination with rotating mirror polygons were used to scan several pixels at a time. The rotation allows to scan along the lines while the inclination of the facets covers the rows. Thus, the resolution of the image is a multiple of the line or matrix detector elements. The fastest approach avoids mechanically moved parts. Here a full detector matrix covers the entire scene. At each time interval all pixel values are measured simultaneously. To achieve regularity between the individual pixel values a fine calibration needs to be guaranteed during manufacturing. [178]

To understand the functionality of a thermal camera the basic physical concepts of heat radiation need to be explained. The following is based on [87, 98]. In contrast to infrared

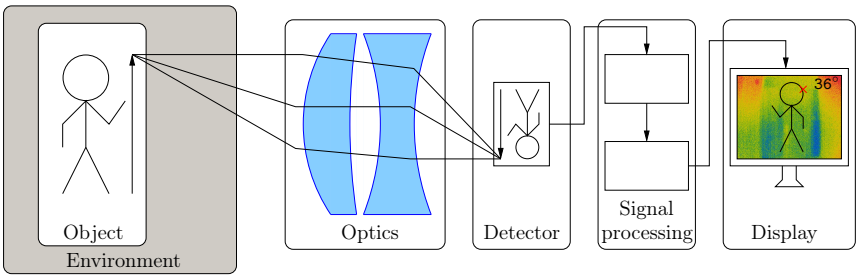


Fig. 2.21: The radiometric chain for thermal imaging.

photography, where only the reflected spectral portion in the near region of the infrared spectrum is measured which does not allow temperature interpretation, thermography aims at measuring the surface temperature radiation.

Thermal radiation is the electromagnetic energy emitted by all objects due to the motion of their constituent atoms at a wide range of frequencies. Relevant for thermography is the surface temperature radiation [87]. The ability to emit and absorb electromagnetic energy depends on the nature, e.g., the color, texture or material of the surface and is described by the emission coefficient ε_λ and the absorption coefficient α_λ. ε_λ describes the energy per unit area per unit time emitted in a tiny wavelength range around λ in $\mathrm{W}/(\mathrm{m}^2\mathrm{m})$ while α is the fraction of the incident radiant energy absorbed per unit time in the same wavelength range. The distribution function $I_\lambda = \frac{\varepsilon_\lambda}{\alpha_\lambda}$ describes the radiant energy per unit time and depends on the temperature T but is independent of material, color or size. In an isolated chamber at temperature equilibrium the absorption and emittance must be identical for each λ (Kirchhoff's radiation law). Otherwise the temperature would change. For a completely absorbing body with $\alpha_\lambda = 1$ that appears black, the distribution function becomes $I_\lambda = \varepsilon_\lambda$ and is identical to the radiant energy distribution that emerges from an isolated cavity at the same temperature. Planck's radiation law gives the equations for calculating the radiation curves of a black body based on the temperature:

$$I_\lambda(\lambda, T) = \frac{2\pi hc^2}{\lambda^5 e^{\frac{hc}{\lambda k_\mathrm{B} T}} - 1},$$

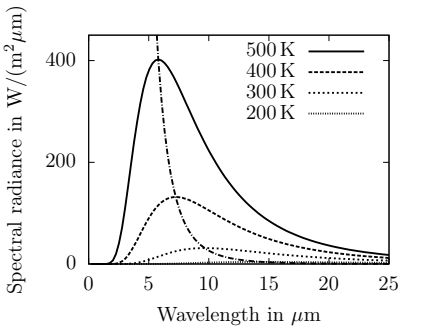
with the speed of light c, Boltzmann's constant $k_\mathrm{B} = 1.3806503 \times 10^{-23}\,\mathrm{m^2 kgs^{-2} K^{-1}}$, Planck's constant $h = 6.62606876 \times 10^{-34}\,\mathrm{Js}$. The spectral radiance as calculated from Planck's law is plotted in Fig. 2.22(a) for some temperatures. Integrating Planck's law yields the Stefan-Boltzmann law for black bodies

$$P = \sigma A T^4 \tag{2.22}$$

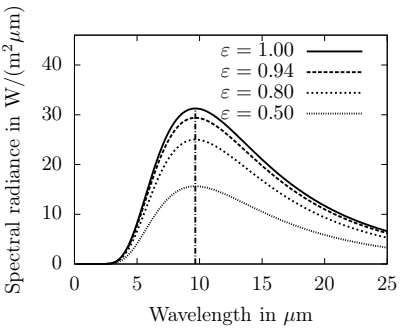
which relates the total radiant power at all wavelengths P to the area of the radiating surface A and the temperature, with the constant $\sigma = 5.67033 \times 10^{-8}\,\mathrm{W}/(\mathrm{m^2 K^4})$. It follows that the total power per unit area, i.e., the area under any one of the black-body radiation curves for a specific T is simply given by $\frac{P}{A} = \sigma T^4$. The fourth power in equation (2.22) shows that the radiation is highly sensitive to temperature changes [98]. For practical applications, however, the black-body assumption cannot be made as the emissivity is dependent on the temperature, the angle of measurement, the surface geometry (flat, convex, concave), the thickness of the material, the surface characteristics (polished, oxidized, rough, sandblasted), the spectral range of the measurement and the transmission characteristics [157]. Instead, the total emissivity ε, a material constant with $0 < \varepsilon < 1$ is used to relate the radiated power of an arbitrary surface to that of the black body

$$P = \varepsilon \sigma A T^4.$$

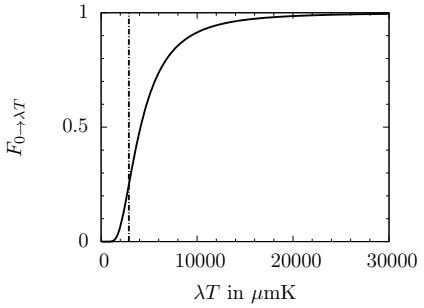
Some exemplary values for the emissivity of different materials at room temperature (300 K) are given in table 2.4 and some of the resulting radiation curves are shown in Fig. 2.22(b). Even though the total emissive power changes one important characteristic remains the same.



(a) Planck radiation law



(b) Effects of emissivity at room temperature



(c) Fraction function

Table 2.4: Total emissivity of selected materials at room temperature (300 K) [98]

Material	ε
Aluminum foil	0.02
Copper, polished	0.03
Copper, oxidized	0.5
Carbon	0.8
White paint, flat	0.87
Red brick	0.9
Concrete	0.94
Black paint, flat	0.94

Fig. 2.22: (a) Spectral radiance for a black body based on the Planck radiation law. (b) Spectral radiance for varying emission values. (c) The fraction function plotted as a finite series explained in [187].

According to Wien's displacement law each radiation curve has a maximum at a wavelength $\lambda_{\max}$. The product of the maximum wavelength and the temperature is constant:

$$\lambda_{\max} T = 2897.8\,\mu\text{mK}.$$

The maximum is shown as the parabola in Fig. 2.22(a) but remains the same for changing ε (cf. Fig. 2.22(b)). For selective radiators where the emissivity is a function of the wavelength $\varepsilon = f(\lambda)$, this characteristic holds not true anymore [87].

In practice Wien's displacement law is applied to determine the approximate wavelength region that corresponds to the measurement value. In this area the optical data of all the factors influencing the measurement must be well known to avoid measurement errors. The fraction function $F_{0-\lambda T}$ is derived by integrating the Planck law within the limits of 0 and λT. The

integral can be represented as a finite series that is plotted in Fig. 2.22(c) [187]. It describes the fraction of emission potential lying in the spectral region between 0 and λ. From the slope of the curve it becomes clear that the area around the maximum $\lambda_{\max}$ contributes the most to the total power. The power potential of a wavelength span is calculated by the difference of the values of two fraction functions. The fraction function applies to all temperatures and wavelengths and is important for practical application.

Apart from the emission ε and the absorption α three further quantities contribute to the radiation of an object, the reflection ϱ, the transmission τ and the extinction e, the fraction of light scattered by the object. The extinction is negligible for surface radiation and assuming $\alpha = \varepsilon$ at thermal equilibrium the types of radiation illustrated in Fig. 2.23(a) contribute to the total power with:

$$\varepsilon + \varrho + \tau = 1.$$

Thus equation (2.22) needs to be adjusted to include the ambient temperature

$$P = \sigma \cdot \left(T^4 - T_{\text{amb}}^4\right) \cdot \varepsilon.$$

In practical applications, where the exact absolute temperature is required, the emissivity is to be measured. Here it is to be considered that the radiation is actually only measured at a certain angle. The radiance

$$L = \frac{P}{\Omega \cdot \cos\varphi}$$

describes what portion of P is radiated within the cone of angle Ω with the angle φ to the surface normal. The emittance relates both the power as well as the luminance of a surface to the emittance and luminance of a black body:

$$\varepsilon = \frac{P}{P_{\text{BB}}} = \frac{L}{L_{\text{BB}}},$$

which forms the basis of measuring the emissivity of a surface. This actual emissivity is a function of many factors. Apart from the temperature and the wavelength the azimuthal and zenithal angle to the surface and the surface quality play a role. For thermal imaging however the acquisition speed and the temperature resolution are typically more important than the accuracy of the absolute temperature values of each pixel [87].

The first infrared detectors were cooled one-element infrared detectors. The oldest image-based infrared detectors used pyroelectric vidicons, i.e., the classical vidicon principle where the object scene is scanned via electron beams is sensitized for the infrared radiation using special material for the detectors. The main disadvantage was low thermal and spatial resolution. Due to their transparency in the thermal infrared range classical semiconductor devices consisting of materials such as germanium, silicon or gallium arsenide are not suitable for detector pixels [178].

The two main developments leading to affordable infrared cameras in the 20th century were the production of focal plane arrays (FPA) as detectors and the realization of micro-bolometers as FPA detectors [61, 87, 178]. The basic principle of a bolometer is that the absorbed infrared radiation heats up a thermistor, i.e. a thermally sensitive resistor that changes its resistance.

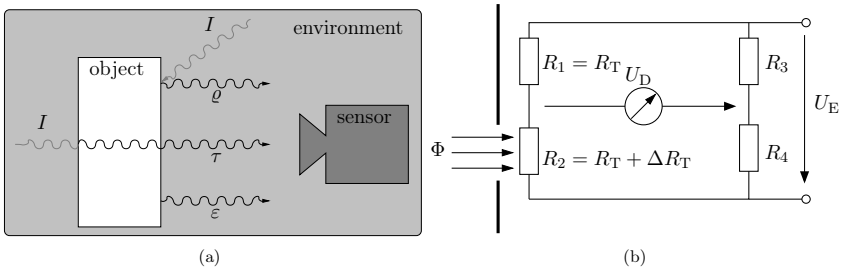


Fig. 2.23: (a) Surface radiation of an object. Reproduced based on [157]. (b) Principle of a bolometer. Reproduced based on [178].

A bolometer consists of a Wheatstone bridge with two thermistors as depicted in Fig. 2.23(b). The two thermistors R_T with identical temperature sensitivity are in the same leg of the bridge. When no radiation enters the bolometer, the bridge is balanced due to the resistors R_3 and R_4. When radiation Φ reaches the open resistor R_2 it changes its resistance by ΔR_T while the the resistance of R_1 remains the same leading to a change in voltage U_D. Changes of the ambient temperature without change of the incident radiation Φ have no influence on U_D as both R_1 and R_2 react to it identically. To calculate the effect of heat radiation precisely, the voltage U_E has to be kept at a constant value [178]. Bolometers work at room temperature but the ambient temperature needs to be kept stable. An optical chopper (shutter) is used to measure and compensate the zero drift [61].

Used within this thesis are two different models of the Optris PI Imager thermal camera, namely the PI160 and the PI400. Both cameras use micro-bolometer arrays and feature similar characteristics. Details are given in table 2.5. The main difference is the higher resolution and larger field of view (FOV) of the PI400 with 382×288 pixels within a FOV of $62 \times 49°$ as opposed to 160×120 pixels and $41 \times 31°$ for the PI160. A further benefit of the PI Imager cameras is their compact design. With a size of $45\,\text{mm} \times 45\,\text{mm} \times 62\,\text{mm}$, a weight of 195 g and a standard screw thread they can easily be attached to any hardware. The spectral range of both cameras is specified as 7.5 to 13 μm and thus lies within the long-wave infrared range. To enable absolute temperature measurements the camera constantly keeps track of three internal temperatures. A blackened piece of metal that is motor-driven in front of the image sensor serves as optical chopper in the Optris cameras [157]. Each pixel is referenced to the same known temperature. During this offset calibration no image data is acquired. For continuous measurements it is recommended to perform the calibration automatically. For taking individual images the option to trigger the calibration at a convenient time is given.

2.4 Laser scanning

The main sensor on Irma3D is a Riegl VZ-400 laser scanner. Laser range imaging uses directed light and the time-of-flight principle to acquire range measurements. Inside a vacuum or gaseous

Table 2.5: Specification of the Optris PI160 and PI400 thermal cameras

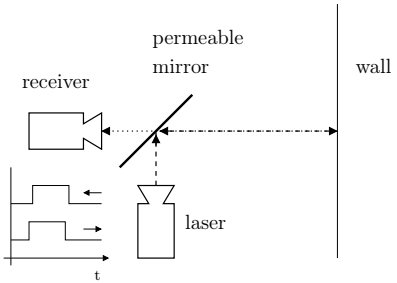
	PI160	**PI400**
Measurement range	$-20 \dots 100\,°\mathrm{C}$	
Ambient temperature	$0 \dots 50\,°\mathrm{C}$	
Relative humidity	$10 \dots 90\%$ non-condensing	
Size	$45 \times 45 \times 62\,\mathrm{mm}$	$45 \times 56 \times 80\,\mathrm{mm}$
Optical resolution	160×120 pixels	382×288 pixels
Weight	$195\,\mathrm{g}$	$320\,\mathrm{g}$
FOV	41×31, $f = 5.7\,\mathrm{mm}$	62×49
Temperature resolution	$0.1K$	$0.08K$
Frame rate	120 Hz	80 Hz
Spectral range	$7.5 \dots 13\,\mu\mathrm{m}$	
Accuracy at ambient temperature $23 \pm 5°C$	$\pm 2°C$ or $\pm 2\%$ whichever is greater	
Warm-up time	10 min	
Emission value	$0.100 \dots 1.000$	

environments light travels with a constant velocity of $c = 2.99792458 \cdot 10^8\,\mathrm{m/s}$, the speed of light. The consistency at which light spreads and the independence of the surrounding medium enable light to be used for distance measurements. Three basic methods are commonly used for time-of-flight based distance measurements [61, 136]. The concept is outlined in Fig. 2.24.

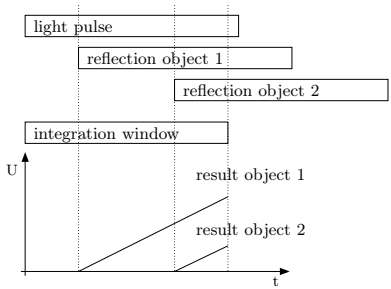
First, for direct time measurements a light pulse or a sequence of pulses is sent out. A receiver waits for the return of the reflected signal. The time between sending and receiving the light is measured and multiplied with $c/2$ to calculate the distance to the obstacle.

Second, impulse integration is often used in image based technology. Simultaneously with sending out the light pulse the sensor starts integrating the received light for a fixed exposure time. With increasing distance to the object the time before receiving the reflected signal increases thus decreasing the amount of light returned before the end of the integration window. In Fig. 2.24(b) the concept is sketched without the influence of the ambient light. For practical applications the procedure is repeated several times to increase the accuracy and to compensate for ambient light. By taking the difference of two images, one without and one with light impulses, the remaining intensity is proportional to the distance of the reflecting objects.

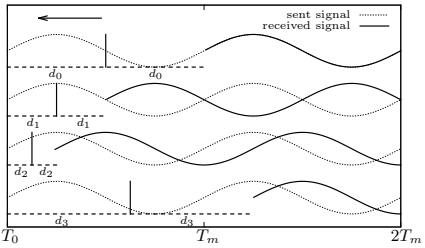
Third, in contrast to the two discontinuous methods measurements with modulated light are a means for continuous distance measurements. Instead of sending out individual pulses a continuous wave is sent out, i.e., the light source is modulated with a sinusoidal signal. The modulation is either a frequency modulation or an amplitude modulation. The frequency modulated continuous wave method allows to measure several targets in the same direction simultaneously but as the signal processing is complex, it is not commonly used, instead the amplitude is modulated. Assuming the signal travels through air the relation between the modulation wave λ_m and the



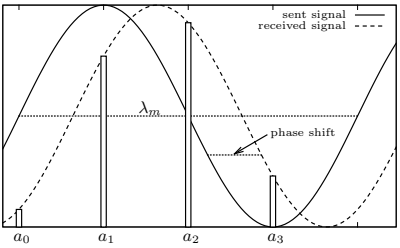
(a) direct time measurements



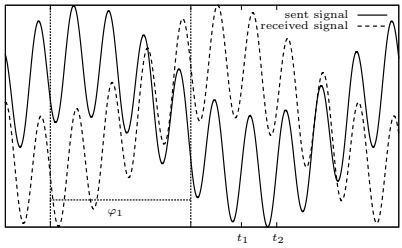
(b) pulse integration



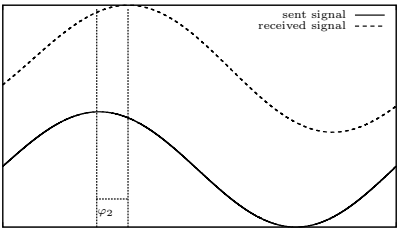
(c) phase shift for changing distances



(d) phase shift calculation



(e) two frequency phase shift ω_1



(f) two frequency phase shift ω_2

Fig. 2.24: Time of flight measurement principles. Source: based on [61]

period time T_m is given by $\lambda_m = c \cdot T_m$. The sensor receives the returned signal shifted by the runtime resulting in a phase shift $\Delta\varphi$ between the sent and the received signal. The distance d to the object is given by

$$d = \frac{\Delta\varphi}{2\pi} \cdot \frac{\lambda_m}{2} + n \cdot \frac{\lambda_m}{2}, \qquad n = 0, 1, 2, \ldots,$$

which means that $\Delta\varphi$ is repeated periodically with the distance to an object, thus yielding unambiguity only between 0 and 2π, i.e., up to a distance of $\lambda_m/2$. Fig. 2.24(c) illustrates both the sent and the received signal for varying distances d_i, where $d_0 = \lambda_m/2$, $d_1 = \lambda_m/4$, $d_2 = \lambda_m/8$ and $d_3 = 5 \cdot \lambda_m/8$, respectively. At time T_0 the signal modulation starts. For d_0 the signal reaches the obstacle from where it is reflected after $T_m/2$, thus the return signal is received at T_m, leading to a phase shift of λ_m, i.e., a full wavelength. For decreasing distances to the object the phase shift also decreases. When the distance to the object exceeds $\lambda_m/2$, as is the case for d_3, the phase shift is ambiguous, in the example, the phase shift for d_3 and d_2 is periodically identical.

To calculate the phase shift from the reflected signal the discrete Fourier transform is used. Four equally spaced samples are sufficient to reconstruct the return signal as demonstrated in Fig. 2.24(d). Let $t_0 = 0$, $t_1 = T_m/4$, $t_2 = T_m/2$, $t_3 = 3T_m/4$ and a_i the amplitude of the signal received at t_i, then the period φ of the phase is calculated as

$$\varphi = \arctan\left(\frac{a_0 - a_2}{a_1 - a_3}\right).$$

The modulation frequency determines directly the unambiguous range and thus the maximum range of the measurement device. With increasing wavelength the range measurement accuracy decreases. To obtain high accuracy at a long range the sent sinusoidal signal P_s is concurrently intensity modulated with two different frequencies ω_1 and ω_2 and the return signal P_r contains the two phase shifts φ_1 and φ_2 for both modulation frequencies:

$$P_s(t) = P_{s_0} \cdot (\cos(\omega_1 t) + \cos(\omega_2 t)).$$
$$P_r(t) = P_{r_0} \cdot (\cos(\omega_1 t + \varphi_1) + \cos(\omega_2 t + \varphi_2)).$$

Fig.s 2.24(e) and 2.24(f) show the concept. The component φ_1 is unambiguous over the entire predefined measurement range and is used for the coarse quantization to determine range window where the fine quantization component φ_2 is unique again while φ_2 is otherwise ambiguous over the effective range. Frequency selective combination of the two phase shift components yields precise and unique measurements over an increased measurement range.

Two of these measurement techniques are used in laser scanning technology. Pulsed laser scanners use the concept of direct time measurements by sending out a focused beam of light. A receiver waits for the return of the reflected light and measures the time between sending and receiving the light. Multiplied with the speed of light $c = 299,792,458\,\mathrm{m/s}$ this yields the distance to an obstacle. Second, phase-shift systems send out continuous waves and calculate the phase shift of the return signal to determine the distance of an obstacle. The two measurement techniques feature different advantages and disadvantages. While phase-shift systems enable high data rates due to the continuous measurements, pulsed laser scanners have to wait for each return signal, thus increasing the measurement time and decreasing the data rate. To increase the data rate modern pulsed systems such as the Riegl VZ-400i offer the option to have two pulses in the air simultaneously where ambiguities are resolved in post processing. The advantage of pulsed systems is their long range. While current phase-shift technology is limited in distance, pulsed laser scanners operate up to distances of several kilometers. A further advantage of pulsed laser scanners is the ability to perform full wave analysis. This is demonstrated in Fig. 2.25. Due

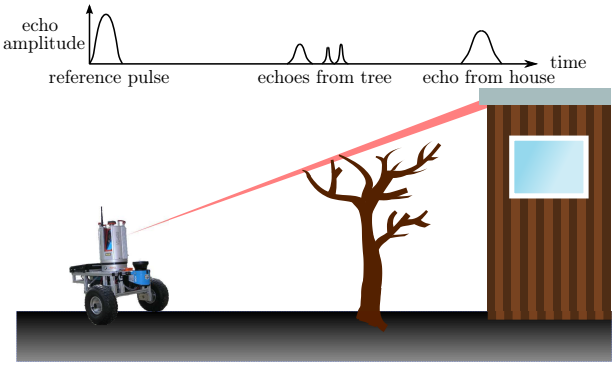


Fig. 2.25: The principle of full wave analysis. Multiple echoes are returned when the beam is split up at edges. Source: [64]

to the beam divergence the beam splits up when hitting an edge. Part of the beam is directly reflected towards the receiver while the other part continues until it hits another target from where it is reflected. This results in several smaller peaks in the return signal and the possibility to measure multiple points per beam. Analyzing the full wave form allows for reasoning about the perceived environment, e.g., detection of vegetation [64].

Apart from range values most laser scanners also measure the reflectivity of the target. By calculating the proportion between intensity of the measured scattered light and the sent signal the reflectivity of the object is directly derivable. The reflectivity depends on the material and the color of the object, but the intensity is also influenced by the distance and the incidence angle at which the laser beam hits the surface. To allow for a 2D laser line scan, 2D laser scanners contain a rotating mirror that directs the sent out laser beam into all directions enabling a field of view of up to 360° at fast rates. The pulsed SICK LMS100 laser scanner that is mounted at the front of Irma3D operates at a rate of 50 Hz and measures the distance to obstacles within its 270° field of view at a resolution of 0.5°. The mobile robot Kurt3D is equipped with a SICK LMS200 laser scanner that features an opening angle of 180° with a resolution between 1.0° and 0.25°. The systematic error is denoted with 15 mm and the statistical error with 5 mm.

The Riegl VZ-400 is a pulsed laser scanner. It operates on power supplies between 11 and 32 V DC. On Irma3D two of the 12 V lead-acid batteries are reserved for the 3D scanner. With a diameter of 18 cm, a height of 30,8 cm and a weight of 9.6 kg its size and weight are small enough to be carried by the robot. The VZ-400 features a vertical scan angle range from −40° to 60° and a horizontal field of view of 360°. Vertical scanning is achieved by rotating a multi-facet mirror. With a step width between 0.0024° and 0.288° this leads to a scan speed between 3 and 120 lines per second. Horizontal scanning is performed by rotating the scanner head. The rotation allows to scan up to 60° per second at a step width of 0.5°. The finest step width is 0.0024°. Two scanning modes are supported. In High Speed Mode the scanner detects targets up to 350 m at a measurement rate of 122,000 points per second. In long range mode the rate

drops to 42,000 points per second at a maximum distance of 600 m. The minimum range is set to 1.5 m. For common mapping tasks the High Speed Mode is sufficient. At a scan resolution of 0.04° both vertically and horizontally a scan takes approximately 3 minutes leading to a point cloud with 22.5 million points, not counting multiple echoes.

The Riegl VZ-400 measures calibrated reflectivity for all targets in dB. The intensity information is the relative reflectance in dB which corresponds to the ratio of the actual amplitude of the return signal to the amplitude of a white flat target at the same distance aligned orthogonally to the laser beam. Thus, such a white target has 0 dB as intensity. The reference for 0 dB is a diffuse target. A relative reflectance higher than 0 dB results from targets reflecting with a directivity different from that of a Lambertian reflector. Such targets may be reflecting foils, corner-cube arrays, license plates, traffic signs or mirror-like objects as, e.g. window panes or blank metal aligned perpendicular to the measurement beam. This is used to create special targets from reflective foil that are identifiable in the scan based on their reflectivity. Placing them in the environment allows for georeferencing the point clouds. To capture color information Riegl offers a camera mount through which certain cameras can be controlled via the scanner to capture images after the point acquisition. [191, 193]

(a) Riegl VZ-400 (b) Faro Focus3D

Fig. 2.26: Riegl offers a camera mount for the VZ-400 [192] while the Faro Focus 3D has an integrated camera to capture color information [71]. In a post-processing step the color information is assigned to the points.

The Faro Focus3D (Fig. 2.26(b)) is a phase-based laser scanner that captures up to 976,000 points per second with a vertical field of view of 320° at a maximum range of 120 m. Apart from intensity values the scanner is capable of capturing color information for each point with an integrated camera. The images are captured after the point cloud acquisition. [71]

2.5 Triangulation

As an alternative to time of flight measurements a second class of optical distance measurement devices use the concept of triangulation. The explanation follows [61] and [118]. Triangulation is based on the reflection of rays and the principle is sketched in Fig. 2.27 [61]. Consider a small bundle of rays that is emitted from a sender. At a distance of d from the principal plane of the optical lens of the sender it is reflected at a surface. A receiver is positioned at a distance B from the optical center of the sender. For simplicity the principal planes of the sender and receiver optic are coplanar and the detection plane of the receiver is parallel to this plane at a distance F. Now the ray that is reflected at the object passes through the optical center of the receiver lens and reaches the detection plane with an offset of x, called disparity. From similar triangles

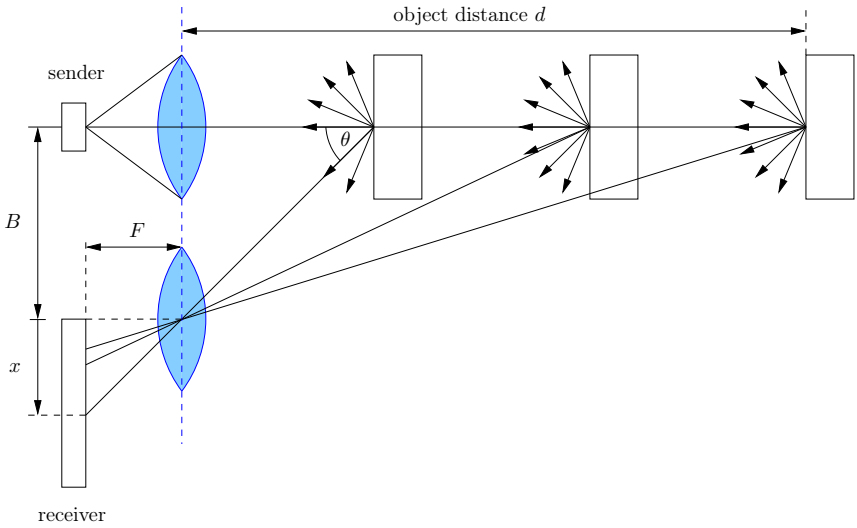


Fig. 2.27: Principle of triangulation. Reproduced based on [61].

the relation

$$\tan\theta = \frac{B}{d} = \frac{x}{F}$$

is derived, where θ is the angle between the incident and the reflected ray. When the object moves further away, the angle decreases and the offset becomes smaller according to the relation. Measuring x with a position sensitive detector allows for direct calculation of the object distance d given that F and B are known:

$$d = \frac{B \cdot F}{x}. \tag{2.23}$$

B is called the baseline and influences the measurable range. The larger B the larger is the measurement range for the object distances. The importance of the baseline is also evident when looking at the sensitivity of the distance measurements, i.e., how much the distance changes depending on the changes of the offset x and subsequently how precise distance measurements can be made [118]. By differentiating equation (2.23) and substituting x results the quadratic sensitivity

$$\frac{dd}{dx} = \frac{-d^2}{B \cdot F}.$$

From this it automatically follows that the precision of a sensor changes quadratically with distance. If the resolution of the offset measurement x allows for a precision of 1 mm at a distance

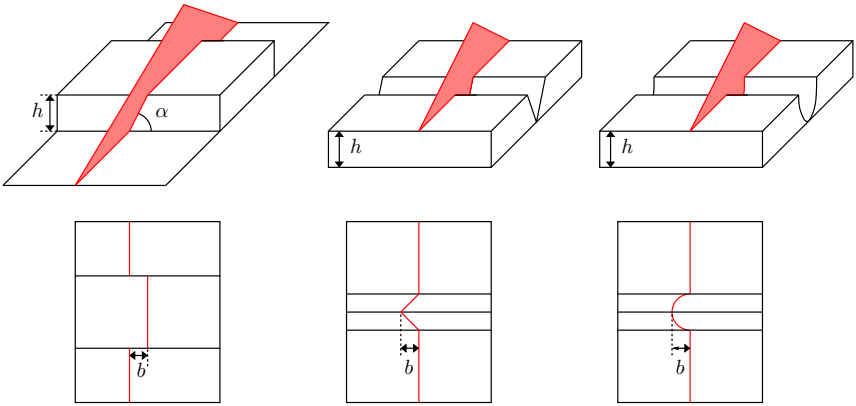


Fig. 2.28: Principle of line scanning. Reproduced based on [61].

of 1 m the precision grows to 4 mm at a distance of 2 m. At a distance of 10 m the precision reaches only 10 cm. Increasing the baseline or the distance F between optic and detection plane with a factor Δ, thus decreasing the field of view, improves the precision by a factor $\frac{1}{\Delta}$. This shows that triangulation measurements have limitations for long distances. In configurations with a laser diode typical devices have ranges between 20 mm and 5000 mm [61].

The concept of triangulation is not limited to single point measurements. Two main categories exist, namely stereo cameras and structured light sensors. Instead of using a sender and a receiver stereo cameras use two cameras that are fixed with a baseline. Their field of view changes slightly and consequently the resulting images differ. To determine the disparities correspondences in both images are established by detecting and matching features such as corners. This is explained in more detail in Section 4.2. Section 8.5 explains how to recover 3D geometry from photo collections, a process often referred to as bundle adjustment or structure from motion (SfM). Stereo cameras suffer from two problems. First, in regions with little texture no features are detected leading to a disability to obtain distance measurements. Second, features are typically not individual pixels but small patches, which decreases the resolution of the resulting depth measurements. [118]

In the second category one of the cameras is replaced by a projector. Here a distinction can be made based on the types of projectors. The first type uses line laser projectors to project lines or grids on the scene while the second type uses a full image projector to project a known pattern onto the scene. Fig. 2.28 explains the principle of a line laser. Assuming a camera is set up above the scene with the optical axis perpendicular to the ground plane. For a line laser the laser ray is diverged using a cylindrical lens to form a laser plane, which when hitting a surface appears as a line. If the line laser is placed at a distance B from the camera and projects a line onto the scene with the incident angle α a change in the height h within the scene leads to an offset b in the image. For different slope types the shape of the line changes as visualized in

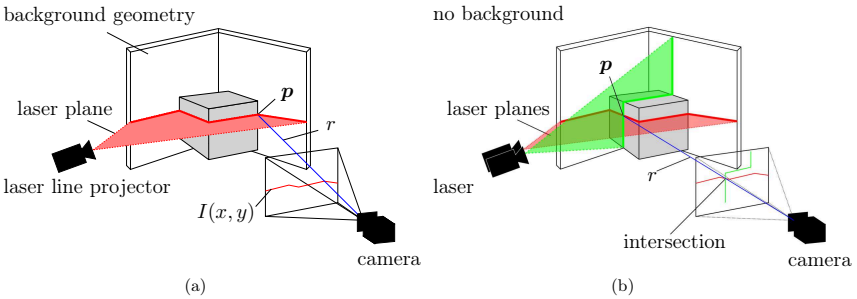


Fig. 2.29: Laser line projector systems using one (a) or two (b) laser planes.

Fig. 2.28. The general relation between the offset and the height is given by

$$h = b \cdot \tan \alpha.$$

Detecting the laser line in the camera image is fairly simple as long as the light is not absorbed by the object. Using a static camera one image is taken with and one without the laser line. The difference image $I_d = I - I_l$ leaves only the pixels that are changed due to the laser line. However, to get a full model, the laser needs to be swiped over the scene and the relative transformation between the camera and the line laser need to be known to determine the angle α. A common solution to this problem is to use a fixed camera and to attach markers on the back plane. In [214] a reference double-frame is placed as a calibration target around the object. The laser is calibrated using the four intersections of the laser plane with the double-frame. The system proposed by [204] is more general as it is independent of a precisely designed double frame. It is similar to the one depicted in Fig. 2.29(a). The main advantage is the increased number of points for calculating the laser plane. The idea is to use the known geometry of the background, e.g., a corner with known angle, for calibrating the laser line.

Consider a static calibrated camera in a scene with known background geometry [204]. Camera calibration is explained in detail in chapter 7. For the line scanner both intrinsic and extrinsic calibrations are needed. This means that not only is it known how a point in world coordinates is projected onto the image but also the position in world coordinates is known, i.e., the relative translation between the optical center of the camera and the background geometry. Using a corner like the one depicted in Fig. 2.29(a), this is achieved by attaching calibration patterns to each of the back planes. When directing the line laser at the object the laser plane intersects with the unknown object surface as well as with the known background geometry. The background points are used to calculate the parameters of the laser plane. From the difference image I_d the line pixels $Y(x)$ are calculated. Assuming a horizontal line, a single y coordinate is determined by weighted averaging over the points in column x. By intersecting the ray through the pixel $Y(x)$ with the background geometry a 3D point candidate is generated for each line pixel. A RANSAC [72] procedure filters the object points from the background points. Three linearly independent background points suffice to determine the laser plane. The remaining points are

used to evaluate the guess. For the remaining object points the intersection of the ray with the laser plane gives the 3D point coordinates. Repeating this procedure while the line laser sweeps over the object yields the full visible surface. With an average or median filter multiple distance estimates for individual pixels obtained from different images are incorporated to get a better final result. A full 3D model of the object requires scans from different angles.

An alternative solution that works without a special background configuration uses a cross laser [21,82]. Here the setup consists of a fixed calibrated camera and two line lasers that produce orthogonal laser planes as sketched in Fig. 2.29(b). Each image shows two laser curves, giving $2{\cdot}N$ lines in N images. The intersections of these lines from multiple images are used to determine the plane parameters for each laser plane up to a scale. Care has to be taken that laser planes with only colinear intersection points are omitted. Furthermore a geometric consistency check helps to reject inaccurately estimated laser planes. The calculation of the plane parameters is a two-step procedure. First solving a linear equation system using the coplanarity constraint, i.e., using the intersection points, yields the plane parameters up to a 4-DOF indeterminacy. The exact plane is then calculated by considering the orthogonality constraint between the laser pair from each image. The intersection of the thus calculated laser planes with the image rays yield the 3D point coordinates as in the single line laser case.

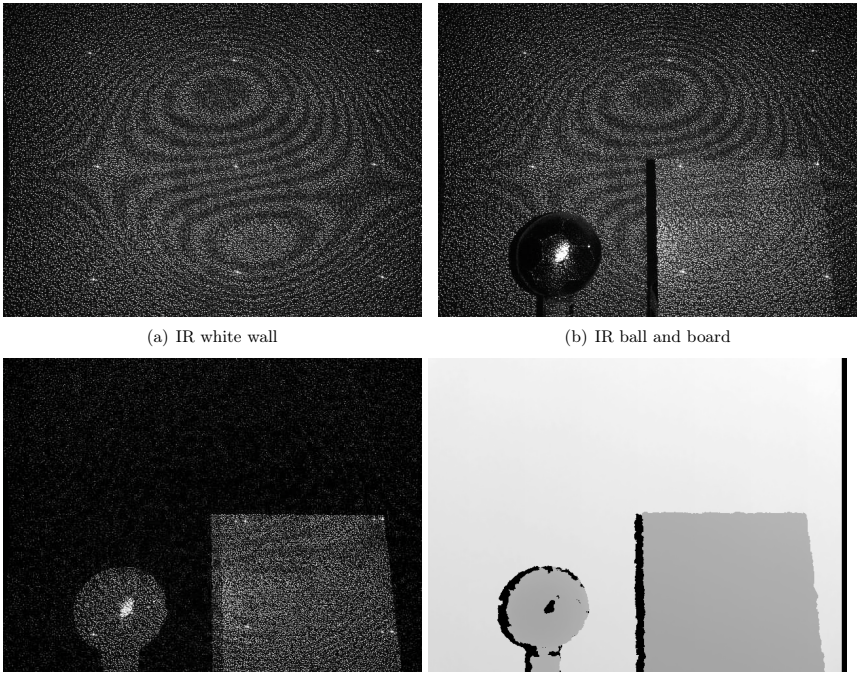
Both solutions, using one or two lasers have one major disadvantage. Each individual image captures only a small number of points. To acquire a full 3D model many images are necessary. An image projector overcomes this issue. The optics of a projector are similar to those of a camera. Instead of detecting the light it emits light rays. Instead of projecting one or two lines into the scene it projects a known pattern that covers the entire scene. Typical patterns are black and white stripes. The depth can be decoded at each pixel or at small patches of pixels, especially at edges. To improve resolution, detectivity and 3D data quality often several images with varying patterns are projected. Gray codes and sinusoidal phase-shifted patterns are popular [118].

Fig. 2.30 shows examples of four scenes with a one line, a two line and a stripe pattern projector. For comparison the scenes are identical, even though only the single line projector requires a known background. The images reveal some drawbacks of the structured light methods. The detection is highly dependent on the lighting conditions. The black and white pattern from the projector is sensitive to high contrast, i.e., changes in lighting and in background color. In dark areas, such as the shorts and the shirt of the dwarf the lines disappear as well as on white parts of the polar bear and the mug. The best results are achieved on the uniformly colored body of the mannequin. A further challenge is to focus the pattern correctly on the desired distance while keeping the projector calibrated and achieving sharp edges even with the limited resolution of the projector. In this setup both the projector and the camera are static. With the requirement of a baseline this limits the field of view further as areas that are visible in the camera are occluded for the projector, an effect which is more present in crowded scenes but is also visible on the side of the mannequin. With the line projectors this effect can be eliminated by moving the projector as demonstrated for the mannequin with the cross line setup. However, with 26 images the body is still far from being completely covered hinting to the amount of images and thus amount of time needed to cover the model completely. The focused light of the line projectors makes the lines more distinguishable even in changing lighting conditions but also more vulnerable to errors due to reflections. Depending on the surface color, the colors of the

Fig. 2.30: Structured light in various scenes using a line projector (left), a cross-line projector (center) and a projected light pattern (right). Last row, center: The accumulated lines of 26 images. As the camera and the scene remain stable the intersection detection is performed with all images.

laser lines change, as demonstrated on the dwarf and on the tennis ball. When using the cross setup this causes problems when trying to distinguish between the two lines. The sensitivity to light makes structured light systems difficult to use outdoors.

The previous methods have in common that they require a static scene where at least one of the hardware components remains in the same position over the entire acquisition time. A second similarity is the usage of visible light which might be disturbing in an environment with humans. The PrimeSense technology [85] overcomes both these issues by combining an infrared projector with an infrared camera in one device. The most prominent device using this technology is the Microsoft Kinect that was developed for motion tracking on the Microsoft Xbox video game



(a) IR white wall

(b) IR ball and board

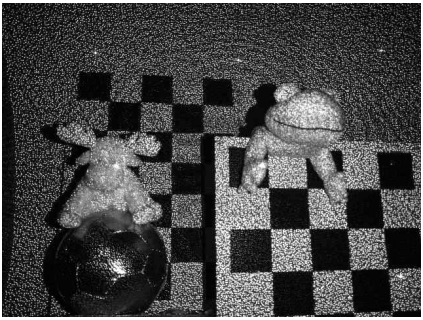
(c) Difference image

(d) Depth image

Fig. 2.31: Images taken with the Asus Xtion Pro Live RGB-D camera. (a) The infrared image captured by the infrared camera from a white wall. (b) The infrared image with a ball and a board in front of the wall. (c) The difference image between (a) and (b). The depth image corresponding to (b).

console. Used within this thesis is an alternative that is especially popular due to its ability to draw power and communicate through USB, the Asus Xtion Pro Live RGB-D camera. Both cameras have similar specifications and consist of an infrared projector, an infrared camera and an additional RGB camera.

In the general concept a beam of light in the near IR range is send through a diffusor. The interference among the different components of the diffused beam creates a speckle pattern that is projected into a scene. Several reference images for different surface distances are stored on the device. When capturing an image of a surface in the measurement area, this image is compared to the reference images to determine the distance. This is done by correlating a block of 19×19 pixels to the reference images along a horizontal line [118]. Locations with local peaks for the cross-correlation indicate the displacement of the points on the object surface, i.e., the disparity



(a) IR image

(b) Depth image

(c) RGB image image

(d) The point cloud colored with the RGB data

Fig. 2.32: A scene captured with the Asus Xtion Pro Live RGB-D camera.

that corresponds to the angle with respect to the projector and thus the change in distance compared to the reference image as given by the triangulation equations. The method requires the speckle pattern to be relatively invariant over a range of distances and unique over a horizontal area leading to strong cross-correlation values only in the range of the 3D measurement. [85, 118]

The Asus Xtion Pro Live RGB-D camera that is used in this thesis operates best at a distance of 0.8 to 3.5 m at an opening angle of $58° \times 45°$. The depth image has a resolution of 640×480 pixels, the RGB image a resolution of 1280×1024 pixels with 30 frames per second. The baseline between the IR projector and the IR camera is approximately 7.5 cm. The RGB camera is between the two at a distance of approximately 2.5 cm from the IR camera. With a depth precision of 11 bits the actual depth decision is limited by the resolution. Due to the smearing effects of the block correlation the spatial precision is lower than suggested by the resolution of the depth image. The pattern expands and contracts based on temperature leading to disparities along the vertical axis. This variation is compensated based on measurements with

a temperature sensor but it is visible when examining subsequent images captured by the IR camera closely. [85, 118]

Fig. 2.31 shows the speckle pattern as seen by the IR camera of the Asus Xtion Pro Live RGB-D camera. The first image shows the speckle pattern on a white wall. At a closer look the pattern appears to be arranged into nine smaller rectangles each of which contains one brighter white circle. In the second image a soccer ball and a board are placed in front of the wall. The change in the speckle pattern is apparent. The reflective characteristics of the ball let the pattern appear much darker while in the center specular reflectance causes a white spot. More evident is the change when looking at the difference image. In the parts where the wall is still visible in the second image, there are small differences caused by noise and by the temperature variation, but the ball and the board clearly stick out. The depth image demonstrates that the specular reflections on the ball prevent the device from making depth measurements. The individual data for a more complex scene is depicted in Fig. 2.32. The speckle pattern is clearly visible in the image. Again the reflectance on the ball but also the fabric of the two animals cause problems for the depth determination and holes in the depth image and consequently in the point cloud. Where the moose is sitting on the ball reflections are visible on the ball surface that correspond to the large hole in the depth image. On the black squares of the chessboard pattern it is hard to detect the speckles with the human eye but the depth image shows that the device succeeded. Nevertheless, materials that absorb the infrared light are a further source for dropouts in the data.

2.6 Position and tracking system

iSpace is a high-precision position and tracking system from Nikon Metrology [144]. The optical laser based system consists of several transmitters. These are mounted on a wall or on tripods to cover the experimental area both indoors and outdoors. The rotating head of each transmitter emits two perpendicular fan-shaped laser beams at a unique distinguishable frequency near 40 Hz. The vertical opening angle of the laser beams is limited to 40 degrees and the detectable range lies between 2 to 55 meters. Several sensor frames can be located within the system. A sensor frame consists of at least one detector, a photo diode with a horizontal opening angle of 360 degrees and a vertical opening angle of 90 degrees. A small radio frequency module transmits the sensor data wirelessly to the base station of the iSpace system, a PC running the iSpace control software. A sensor frame with one detector is sufficient to acquire 3D position information. To measure also the rotation and to increase the accuracy of the position data the sensor frame used on the robot within this thesis has a total of four detectors. A sensor frame with two detectors, the hand-vector bar, is used to measure points in the environment to evaluate the quality of the resulting 3D model. The iSpace system differs from other position and tracking systems as the transmitters do not actively observe the position of the sensor frames. Instead, each sensor frame receives the laser data from the transmitters and sends the information on to the control PC. The control PC calculates the elevation and azimuth angles between all detectors for a sensor frame and each visible transmitter based on the received data defining a straight line between transmitter and detector. Given the relative transformation between the transmitters the length of the lines is calculated using triangulation. To determine the position of the transmitters a calibration

procedure using a few hundred points from a special sensor frame is applied. An optimization process calculates the position of all transmitters in a self-defined coordinate system. Three points, the origin, a point on the x-axis and a point on the y-axis allow the user to define its own coordinate system. In typical environments the iSpace system is able to perform measurements at a sampling rate of 40 Hz with a maximum error of [$\pm$0.25]mm. In practice environmental factors such as size, reflection of the surface and occlusions of the transmitters have to be taken into consideration. The iSpace system is used in this thesis to evaluate the quality of the mapping process.

2.7 Summary

The basis for the sensors used in this thesis is electromagnetic radiation in the range of visible and infrared light. Modelling the propagation of light using wavelets and rays, its behavior comes down to general geometric rules. This explains why it is possible to measure radiation using sensor arrays, as done in color and infrared cameras, resulting in 2D image representations. Using th principles of time-of-flight or triangulation it is even possible to recover the 3D geometry as a point cloud representation of the environment.

Next, several data sets are presented that were collected using optical sensors partially with or without a mobile robot (cf. Section 2.1). All data sets contain 3D point clouds. These are collected using a Riegl VZ-400 or a Faro Focus3D terrestrial laser scanner, SICK LMS 2D laser scanners mounted on a rotation unit, or an Asus Xtion Pro Live RGB-D camera. In most scenarios one or more cameras are calibrated to the 3D range sensor (cf. Chapter 7). These cameras are an Optris PI160 or PI400 thermal camera, a Logitech QuickCam Pro 9000 webcam or a Canon EOS 1000 DSLR camera. Additionally, some photos are captured that are used to recover the 3D geometry from feature matching (cf. Sections 4.2, 8.5). The specifications for the hardware are given in Sections 2.4, 2.2.10, 2.3 and 2.5.

Chapter 3

Data sets

Throughout the past few years a lot of data was collected using the technology described in the previous chapter to present and evaluate the methods described in the following chapters. This chapter describes the environment, the hardware setup and the acquisition procedure of selected data sets that are used in this thesis to demonstrate the results. The AVZ, Hannover1, Bremen City, Arena and Bräutigam data sets are available from the 3D scan repository [153].

3.1 Zagreb

A data set for calibration only was recorded at the Faculty of Electrical Engineering and Computing in Zagreb. Fig. 3.1 shows the calibration pattern with small light bulbs in the corridor used for data collection. The calibration procedure is explained in Chapter 7. Mounted on top of the Riegl VZ-400 laser scanner is the Optris PI400 thermal camera. Used within this thesis is a part of the data set consisting of 70 pairs of laser scans and thermal images.

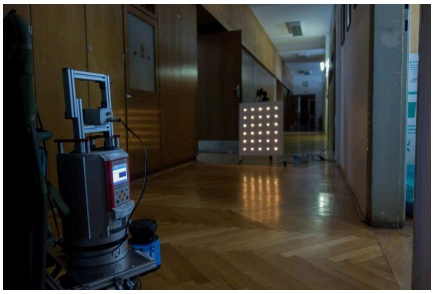


Fig. 3.1: The setup for calibration in Zagreb.

In the narrow hallway the scan is restricted to cover only an angle of 50° horizontally. The process is started by taking a thermal image at the central scanning angle. The following scan with a resolution of 0.08° results in more than 500,000 points per scan. Then the calibration pattern is moved to a different position and the process is repeated.

3.2 Jacobs Thermo

A series of experiments was carried out on the Campus of Jacobs University Bremen, Germany with the aim to build a complete 3D model of the environment based on 3D scans with thermal

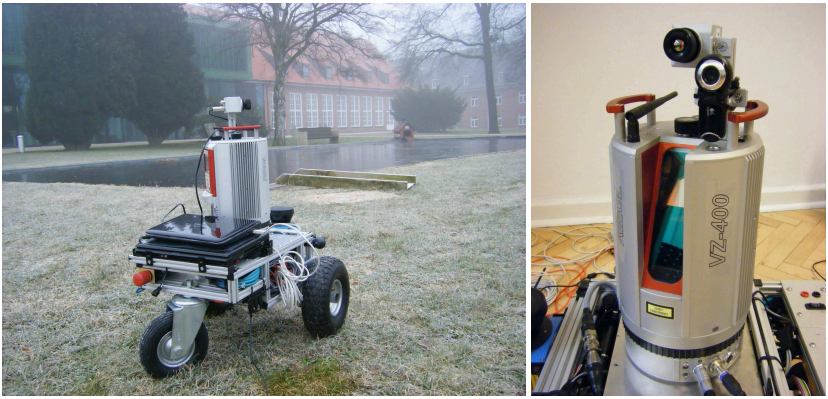


Fig. 3.2: The robot Irma3D on the Campus of Jacobs University Bremen, Germany (left). The laser scanner with the thermal camera and the webcam mounted on top (right).

information. The setup for simultaneous acquisition of 3D laser scan data and thermal images is the robot Irma3D with an Optris PI160 thermal camera and an optional Logitech QuickCam Pro 9000 webcam (see Fig. 3.2). The laser scanner acquires data with a field of view of $360° \times 100°$. To achieve the full horizontal field of view the scanner head rotates around the vertical scanner axis when acquiring the data. The setup takes advantage of this feature when acquiring image data. Since the cameras are mounted on top of the scanner, they are also rotated. Acquiring 10 images per camera during one scanning process covers the full 360°. To avoid blurring and the problems that arise from the need to synchronize the images are not taken while scanning. Instead the scanner performs a full 360°rotation for scanning and rotates back with stops at the image positions. A further advantage of this strategy is that the cameras can be connected with regular USB cables because the cable is unwound after each rotation.

For the first data sets the robot was remotely controlled. Data set Jacobs Day consists of laser scans, photos, and thermal images recorded at 13 different positions outside on campus in the early afternoon. A second data set, Jacobs Night, was recorded at night consisting of laser scans and thermal images from 15 different positions in the same area.

Several experiments for the Automation Lab data set were carried out in a research building at Jacobs University. The environment consists of one laboratory and two offices that are connected by a hallway. The data was collected in a stop-and-go fashion. At each scan position a laser scan and nine or ten images per camera were collected. Of special interest are two data sets that were collected autonomously by the robot Irma3D. The first part was collected during 2D exploration and nine images per position were captured and is used for the experiments in Sectioni 7.2.3. The second part with ten images per rotation was acquired during 3D exploration and is explained in detail in Chapter 6.

Fig. 3.3: Irma3D in front of the Cathedral (left) and the Roland statue in front of the City hall in Bremen City.

3.3 Bremen City

Data set Bremen City consists of laser scans and thermal images recorded at night at 11 different positions around the Bremen City Hall. Fig. 3.3 shows Irma3D on the market square next to the Roland statue in front of the City Hall while acquiring the data. The City Hall and the Roland have been a UNESCO world heritage site since 2004 as they symbolize the unique development of autonomy and sovereignty of the people of Bremen. The Roland, was built in 1404 as a symbol for liberty and market rights. With a height of 5.5 m it is one of the oldest and most representative Roland statues in Central Europe. It refers to a paladin of Charlemagne and stands for the rights given to the people of Bremen by the emperor. The City Hall, built between 1405 and 1409, has never been destroyed and is until today an auhentic example of Gothic stlye. Further remarkable buildings nearby are the Bremen Cathedral, the Schütting and the statue of the Bremen Town Musicians. [180]

The thermal images were recorded at night with temperatures below 0° C. The robot was remotely controlled. Scanning with a resolution of $0.04 \times 0.04°$ and taking nine thermal images per scanning position with the Optris PI160 resulted in a total acquisition time of approximately 90 minutes and a total of 177.6 million points. A video showing the entire scene with thermal information is available at [11].

Additionally, 143 photos were taken independently during the day on the market square with a Panasonic DMC-LS80 camera. Bundler [184] is used to reconstruct the environment from these photos. The Patch-based Multi-view Stereo Software (PMVS) [84] is used to produce a dense reconstruction of the bundler results. The photos cover only a part of the environment mapped with thermal information.

3.4 Ostia Antica

Ostia Antica is a large archaeological site, close to the modern suburb of Ostia (Rome). Due to the exceptionally well preserved state of the city, Ostia is of immense interest for the study of the Roman empire. According to archaeologically unverified tradition, Ostia was founded during the second half of the 7th century B.C. at the mouth of the Tiber river. This region was said to be strategically important for Rome because of its salt marshes. The existence of the settlement

Fig. 3.4: Irma3D at the garden house in Ostia Antica. Top left: At the last scan position, in front of the garden house. Top right: Overview of the room structure. Bottom left: Inside the large hall. Bottom middle: Calibration image from the first day. Bottom right: calibration image from the last day.

at the mouth of the Tiber as early as the beginning of the 4th century B.C. is supported by evidence. Initially only a military camp meant to defend Rome towards the sea, Ostia grew to be an autonomous harbor city of about 50 hectares with a population of approximately 50,000 in the second half of the 2nd century A.D. In the following decades began the decline of the city. Large parts of the town were abandoned following historical events. The last inhabitants left the city after raids of the Saracen pirates from the sea around the 9th century A.D. [158].

The experiments took place in one of the Hadrianic garden houses in Ostia Antica. Historians suspect that these garden houses were a large building project consisting of many houses built in series production that were rented out afterwards. Of special interest are the architecture and accouterments, i.e., the wall paintings and the floor mosaics. The garden house in question contains one large semi-open area and several smaller rooms that are connected by narrow hallways. Thus, scanning the entire area completely without holes requires a large number of scan positions.

Scanning took place over the course of five days. A total of 59 scans with a resolution of

Table 3.1: Work progress in Ostia Antica

	Day 1	Day 2	Day 3	Day 4	Day 5
# scans	6	16	16	15	6
Exposure time	1/30	1/125	1/200	1/500	1/500
Aperture value	$f/10$	$f/4$	$f/4$	$f/4$	$f/4$
Focal length	18.0	18.0	18.0	20.0	21.0

0.03° was acquired with the mobile robot Irma3D. Irma3D was equipped for these experiments with the Riegl VZ-400 laser scanner and the Canon EOS 1000D DSLR camera (Fig. 3.4). This was the first field test of this setup. A video explaining the data collection can be seen at [10].

Every morning the exposure time of the camera had to be adjusted according to the lighting conditions. Calibration was performed using a chessboard pattern as depicted in Fig. 3.4. A set of images for intrinsic calibration was taken every morning. The data for the extrinsic calibration, i.e., a photo and a laser scan with the calibration pattern, was only collected on the first and on the last day. The number of scans acquired per day is listed in Table 3.1. The first day was mainly used to prepare the environment for data collection. Especially the floor had to be cleaned from dust to allow for the floor mosaics to be discerned. The first scans in the afternoon of day 1 were taken to check the functionality of all components. On day 2 the full day was reserved for scanning and the number of scans could be increased. On day 3 a technical defect caused a break in the data acquisition phase but the defect could be resolved and the scanning resumed. At the end of day 4 all accessible rooms of the garden house had been captured. Day 5 was left to take some additional scans in areas that were hard to access and from the outside to complete the model. Table 3.1 also presents some of the camera parameters listed in the EXIF file. Noticeable are the varying aperture values. For the extrinsic calibration the aperture value was set to $f/8$. As known from the previous chapter, the optical characteristics are influenced by a change in aperture. The zoom lens including the focus ring was fixed with tape. As seen in the EXIF data, the focal length changed after the third day. As a result the calibration pattern in Fig. 3.4 becomes blurry. As most of the data processing was carried out after leaving the acquisition site this had a negative impact on the calibration accuracy. This is discussed in Sections 8.4 and 8.5.

3.5 Würzburg Residence Palace

The Würzburg Residence is a unique baroque palace in the city center of Würzburg, Germany that was labeled a UNESCO World Cultural Heritage site in 1981. Being built from 1720 to 1744 with the interior finished in 1780 it is now one of Europe's most renowned baroque castles [209]. It was laboriously reconstructed after being heavily damaged during World War II. The existence of a 3D color model would have made a realistic reconstruction easier. Not destroyed during the war were the large unsupported trough vault above the main staircase designed by architect Balthasar Neumann, the Garden hall with ceiling paintings by Johann Zick, the white hall with the impressive stucco work by Antonio Bossi and the Imperial Hall with frescos by Giovanni

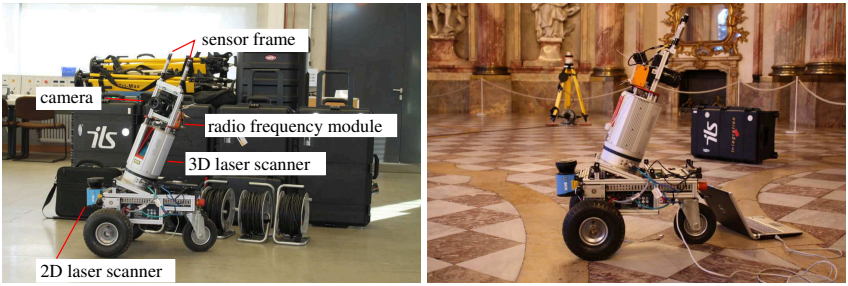


Fig. 3.5: Irma3D. Left: With the setup used in the Residence. Right: In the Imperial Hall during calibration. In the background is one of the transmitters for the iSpace localization system and a box with two reflective markers for the calibration procedure.

Battista Tiepolo. With its large colorful paintings by the Venetian painter Giovanni Battista Tiepolo and fine stucco work by stuccoist Antonio Giuseppe Bossi in many of the almost 400 rooms the Würzburg Residence is a unique example of baroque style.

Experiments were carried out in both the White Hall and the Imperial Hall, two large halls with impressive 3D structure. The colorful paintings in the Imperial Hall and he 3D structure of the environment can only be captured simultaneously by the combination of two technologies, i.e., laser scanning and photography. Thus the data was acquired with the mobile robot Irma3D.

The iSpace positioning system was set up in the Imperial Hall of the Residence Palace to evaluate the results of the localization. The Würzburg Residence was ideally suited for the experiments as the floors were accessible for the wheeled robot and the halls were large enough for the iSpace system to be set up so that the entire environment could be observed by this high precision localization system. Irma3D was equipped for these experiments with the Riegl VZ-400 laser scanner, the Canon EOS 1000D DSLR camera, and an iSpace sensor frame. To account for the small opening angle of the camera, the scanner was mounted atilt on the robot to enable the view of the ceiling (Fig. 3.5).

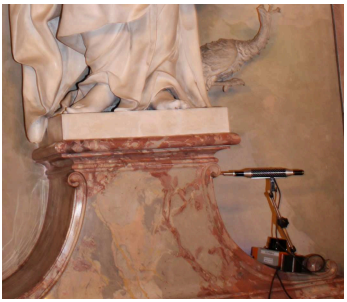


Fig. 3.6: The hand-vector bar is used to measure distinctive points in the environment that are used to evaluate the quality of the final model.

To capture the entire environment, data had to be collected at several locations. This was especially crucial due to the restricted field of view of the camera. Accordingly, the robot was manually driven to a scanning location and stopped there for data collection. The scan resolution was 0.04° both horizontally and vertically at a field of view of $360° \times 100°$. After

Fig. 3.7: Excavation of the rediscovered Stephanitorzwinger in Bremen, Germany.

the scan was taken, the scanner head was rotated in discrete steps to take 12 pictures with a resolution of 3888×2592 pixels at each scanning location. In the White Hall nine scanning positions were used. Starting from one position near the entrance, the other scanning positions were chosen in the corners of the hall. Due to the tilt of the laser scanner two scans were taken in each corner, one facing the corner, and the other facing the open hall.

To achieve a quantitative evaluation of the methods the iSpace system was set up before data collection in the Imperial Hall. Six transmitters were mounted on tripods and calibrated to define a coordinate system as depicted in Fig. 3.5. iSpace was then able to measure the robot position precisely using the sensor frame attached to the top of the laser scanner. Furthermore, seven distinct points in the environment were measured using the hand-vector bar to evaluate the quality of the final model (cf. Fig. 3.6). In the Imperial Hall, 11 scanning positions were used. At each scanning position the pose (position and orientation) of the robot was recorded using the iSpace system. Due to the limited strength of the iSpace signal the idea of taking two scans in each corner could not be pursued. Instead, more scans were recorded in various positions near the center of the hall to reduce occlusions, especially from the tripods with the transmitters, as much as possible.

3.6 Bräutigam

The Bräutigam (German: groom) data set was collected in Bremen, Germany at the excavation site of the rediscovered Stephanitorzwinger, which is depicted in Fig. 3.7. Built from 1525 to 1545 this tower was part of the town fortification of Bremen. With a height of 40 m and its position halfway in the river Weser it was the first landmark for ships reaching the city of Bremen. It was destroyed in the 17th century when lightning set the stored munitions on fire [42].

The data set consists of eight terrestrial 3D scans and ten images per scan acquired with the Riegl VZ-400 laser scanner and the Canon EOS1000D DSLR camera mounted on a tripod.

Fig. 3.8: The scanner setup in three selected positions. The Scanner was carried with the backpack seen in the left image and mounted on the tripod.

The exposure settings were adjusted once and kept fixed for all images. The measurements took several hours. Consequently, the exposure of the captured images was influenced by the position of the sun as well as clouds obscuring the sun.

3.7 Helgoland

Located in the German Bight of the North Sea with a distance of 70 km to the mainland, Helgoland is Germany's only deep-sea island. Since 1720 it consists of two islands as a storm tide destroyed the natural connection between the main island and the dune. The main island has a size of approximately 1 km^2. Its cliff coast reaches a height of 61 m above the water. While the smaller dune with a size of 0.7 km^2 has sandy beaches the main island is mainly composed of the characteristic red Buntsandstein, chalk and Muschelkalk seen in Fig. 3.8 and offers the perfect ground for a unique flora and fauna. More than 400 bird species have been recorded on Helgoland and in spring time more than 30,000 seabirds come to Helgoland during breeding season. The most famous landmark on Helgoland is the *Lange Anna*, a free standing rock column depicted in Fig. 3.8.

The goal of the HELGOLAND scanning project was to survey the *Lange Anna* over a period of several years. Therefore laser scans were collected in April of 2014, 2015 and 2016 around the rock column. The scanning positions are shown in Fig. 3.9. In 2014 nine scan positions were chosen, four on the 'Oberland', the upper part of the island, and four on the rocky tidal flats that are only accesible during low tide. In the following years the positions were increased to 13 scanning positions. The hardware setup is depicted in Fig. 3.8. The Riegl VZ-400 laser scanner was carried to the scanning positions with a Tatonka load carrier that is equipped with two 12 V lead batteries and the necessary cables for power and data connection. At each position the scanner was mounted on a tripod and two scans were taken, one full scan with low resolution and one scan with higher resolution that covers only the *Lange Anna*, and 10 photos. A compass and GPS were used to record the scanning poses.

Fig. 3.9: GPS positions of the scans from 2014 (blue), 2015 (yellow), 2016 (red). Satellite image source: [91].

3.8 Projector

The PROJECTOR data set consists of individual parts with identical setup for testing a system that combines thermal imaging with AR similar to the one recently presented by Palmerius and Schönborn [159]. By co-calibrating a thermal camera with a projector the visual representation of the temperature information is directly reprojected into the scene. The system for spatial projection of thermal data for visual inspection consists of three main hardware components. A thermal camera perceives the infrared radiation emitted from observed objects that corresponds to their temperature. A depth camera records the 3D geometry of the scene while the projector projects the observed information back into the scene to make the temperature values visible to the user. Each component defines its own coordinate system. The essential point is to determine the transformation between these coordinate systems and to transform the data accordingly. Here the small test scenarios are described that are used to evaluate the approach proposed in Section 8.3 where more information about the methodology and the workflow is given.

In all test scenarios the hardware is identical. For 3D measurements an Asus Xtion Pro Live RGB-D camera is used that is mounted on a tripod with an Optris PI400 thermal camera. The Panasonic PT-VZ575N is a portable projector capable of a resolution of 1920×1200 pixels at 4,800 lm. The hardware (depicted in Fig. 3.10) is set up in the work cell of a KUKA KR 16 industrial robot. Due to the high frame rate of the depth camera no fixture for the calibration pattern is required.

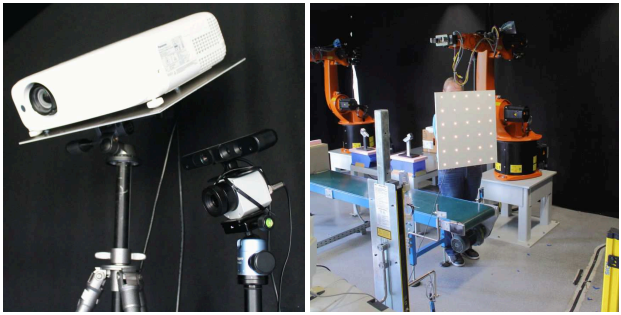


Fig. 3.10: Left: Hardware used in the experiments, a Panasonic PT-VZ575N projector, an Asus Xtion Pro live RGB-D camera and an Optris PI 400 thermal camera. Right: The calibration procedure.

3.9 Additional data sets

Some of the experiments and examples in this thesis were carried out on smaller data sets or with individual scans from data sets where the recording procedure is not important. These data sets are described in this section. Their common ground is the focus on the point cloud data.

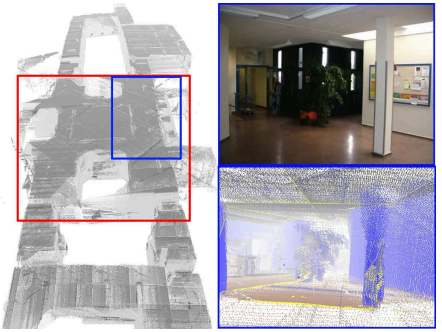
The AVZ data set was collected with the mobile robot Kurt3D in the AVZ building at the University of Osnabrück. It consists of a robot run with 76 scans of 81,360 points each from the fourth floor of the University building as well as several scans with different resolutions of an empty seminar room. The red box in Fig. 3.11(a) marks a hall where eight hallways meet. The wave-like systematic errors seen on the floor are caused by inaccurate control of the servo motors of the pan-tilt unit for the 2D laser scanner.

The HANNOVER1 data set consists of of 468 laser scans from a robot run recorded by Oliver Wulf at the Leibniz Universität Hannover while driving a distance of approximately 750 m. Each scan consists of 14,000 to 18,000 scan points.

The ARENA data set was recorded with a Riegl VZ-400 laser scanner on a tripod in the Mobile Robot Test Arena at Jacobs University (cf. Fig. 3.11(b)). It contains 27 laser scans that have been registered manually using a marker-based approach. The registration has been visually inspected and verified. Each scan has a resolution of 0.08° both horizontally and vertically at a field of view of 360° × 100°.

The SEMINAR ROOM and the BASEMENT data sets were both collected in the Research I building at Jacobs University Bremen with the mobile robot Irma3D. The scan settings were a resolution of 0.04° both horizontally and vertically using the maximum field of view leading to approximately five million points per scan. The two rooms are depicted in Fig. 3.11. Most of the furniture shown in the photo of the seminar room was removed for scan acquisition to enable the scanning of the architectural structures. The pipes and ceiling structure in the basement occlude the walls behind as shown exemplarily in the scan depicted in Fig. 3.11(d).

The CHURCH GROHN data set was recorded with the Faro Focus3D laser scanner. Approximately 28 million points were measured at each of the 12 scanning positions in a church in



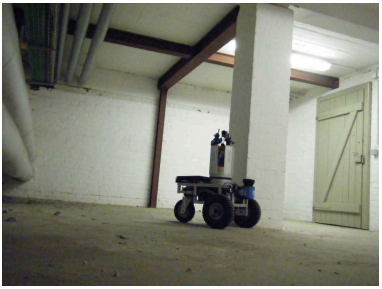
(a) AVZ

(b) Arena

(c) Seminar room

(d) Basement scan

(e) Basement



(f) Basement

Fig. 3.11: (a) The complete 3D map of AVZ as seen from above. The red box marks the area used for the experiments in Section 4.3.4. The blue box marks the area shown on the right. (b) The Riegl-VZ on a tripod during data acquisition in the Arena. (c) Photo of the Seminar room and (e)+(f) Basement environment. (d) A scan showing approximately the same area as (f).

Table 3.2: Summary of the datasets

Data set	Robot	Riegl	Faro	SICK	Thermo	Camera	Photos	SfM
Zagreb		✓			✓			
Jacobs Day	✓	✓			✓	✓		
Jacobs Night	✓	✓			✓			
Jacobs Lab	✓	✓			✓	✓		
Bremen City	✓	✓			✓	✓	✓	✓
Ostia	✓	✓				✓		✓
Imperial Hall	✓	✓				✓		
White Hall	✓	✓				✓		
Bräutigam		✓				✓		
Helgoland		✓				✓		
AVZ	✓			✓				
Hannover1	✓			✓				
Arena		✓						
Seminar room	✓	✓						
Basement	✓	✓						
Church Grohn			✓				✓	✓
Universum	✓	✓				✓		

Bremen, Germany. Apart from the distance, the Focus3D measures reflectance information. With an internal camera it also determines an RGB value for each point. Additionally, 121 photos were captured with a Canon EOS 1100D for reconstruction with VisualSfM [205, 206].

The Universum data set was recorded at an interactive science exhibition in the Universum Bremen. For this experiment Irma3D is equipped with the Riegl VZ-400 laser scanner and the Canon EOS 1100D camera. A scan with a resolution of 0.03° and nine photos were recorded at 16 scanning positions. For large parts of the experiment the robot operated in autonomous mode with the method described in Chapter 6.

3.10 Summary

The data sets presented in this chapter vary in acquisition procedure and hardware setup. Table 3.2 summarizes important characteristics. The first column lists whether a robot was used for data acquisition followed by the laser scanner setup and the type of camera calibrated to the laser scanner. The last two entries refer to photos that are taken manually and the reconstruction using SfM. The somewhat unique setup for the Projector data set is not listed in Table 3.2. It consists of a RGB-D camera, a thermal camera and a projector used to project the perceived information back into the scene.

Chapter 4

3D point cloud processing

A point cloud acquired with a laser scanner contains accurate measurements of the environment. The values of each measurement are guaranteed a certain precision and accuracy by the manufacturer. Despite this characteristic, handling the huge amount of raw data from a point cloud poses limits to many applications. Starting from simple applications as determining the bounding box of a scene over collision detection to visualization tasks the computational costs are tremendous if one has to touch every single point. Consider an unorganized point cloud taken with a Riegl VZ-400 laser scanner that contains 22.5 million points at a resolution of 0.04° both horizontally and vertically. The acquisition time for the point cloud is approximately three minutes. To be able to perform any point operation on all points of the scan sequentially within one second the operation must be performed in less than 44.4 ns per point. Calculating, for example, the euclidean distance between two points in C++ on a standard laptop computer with an Intel Core i7 quad-core CPU with 2.1 GHz takes approximately 100 ns. This simple example shows that processing point clouds heavily asks for parallelization and data structures that allow for performing certain tasks without the need to access all points from the point cloud but instead enable the direct access of points within the vicinity of the area of interest. This chapter summarizes essential methods for low level point cloud processing. In the first section data structures that enable fast processing of point clouds are presented.

To reduce processing time even further it is often beneficial to add semantic knowledge to the data. Instead of considering each point belonging to an object, reducing the computation to the primitive shapes the object is composed of, such as planes, cylinders or spheres, often decreases the computational costs. The most simple primitive shape in 3D is a plane. Man-made structures consist in large parts of planar regions. Thus, determining the most prominent planes in a point cloud extracts valuable knowledge from the data. This is of interest when trying to recover the full geometry of building interiors (cf. Section 8.1) or when trying to find a distinct object in the scene as needed for camera to laser scanner calibration (cf. Chapter 7). Calculating the distance between planes also reduces the measurement error to systematic errors by canceling the noise of individual point measurements. Section 4.3 gives an overview over plane detection methods. Related to plane detection is the calculation of surface normals that is useful for validating thermal measurements (cf. Section 7.2.5).

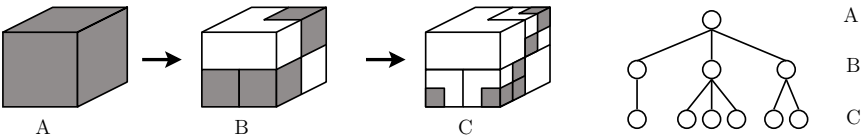


Fig. 4.1: The octree structure divides the data into cuboids (Reproduced based on [69]).

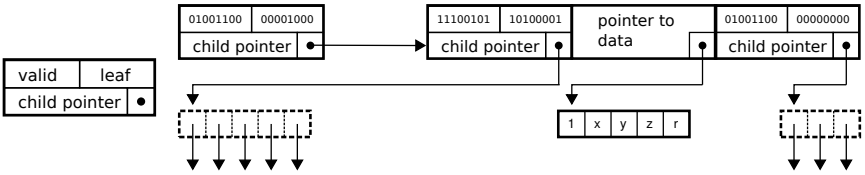
4.1 Data structures

Point cloud processing is inherently sped up by the use of two types of data structures, namely data structures that exploit the spatial structure of the data or those that simulate the characteristics of the acquisition process. The first category divides the point cloud into spatially connected parts. In the following, two tree structures are presented that are ideally suited for point cloud storage, namely the the octree and the kD-tree. The points are stored in the leaf nodes. On each level a check determines which child nodes need to be exploited thus reducing the required number of accessed points drastically. Even though point cloud data is three-dimensional, it is not truly volumetric. Laser range finders scan only the surface of objects. They strobe their surrounding in a sweeping fashion. 3D cameras even acquire a regular grid. The second category makes uses of this concept by projecting the acquired data onto a 2D plane. Section 4.1.3 describes panorama generation methods and their use for 3D point cloud processing.

4.1.1 Octree

The octree is a hierarchical data structure for the organization of 3D points. The concept behind the octree is quite intuitive. It divides the space containing the point cloud into cubes or cuboids. Starting with the bounding box that encloses all points in the data set the octree splits each cuboid recursively into eight equally sized cuboids. Fig. 4.1 shows how the data is stored in a tree of height three. On the left are the cuboids, on the right the corresponding tree. Cuboids that contain at least one point are colored in gray while empty cuboids are depicted in white. The root node represents the bounding box of the entire point cloud. Each node consists of a cuboid and has eight child nodes corresponding to the eight cuboids that are created by splitting the cuboid of the parent node along the center parallel to the planes of the coordinate system. If a cuboid contains no further points, the node remains empty and the branches of that subtree tree are no further expanded. This leads to only three nodes on the level B of the tree corresponding to the three gray voxels in the voxel representation B. The six nodes on level C represent the six voxels seen in octree subdivision C.

To store the data efficiently the representation from [69] is chosen. The root node contains some information about the octree. All other nodes are limited to the essential information to limit redundancy, as depicted in Fig. 4.2(a). Nodes are either a leaf node or an intermediate node. An intermediate node has up to eight children. One bit per child determines whether the child node is valid, i.e., it exists because it contains point data. A second bit declares whether the child is a leaf or an intermediate node. Additionally, a pointer to the first child is stored. Using the validity information and this pointer, each child node is directly accessible. A leaf



(a) Representation of a node

(b) Memory representation

Fig. 4.2: A small example of an octree as it is stored using the proposed encoding (Source: [69]). The node in the upper left has three valid children, one of which is a leaf. Therefore, the child pointer only points to three nodes stored consecutively in memory. The leaf node in this example is just a pointer to an array which stores both the number of points and the points with all their attributes.

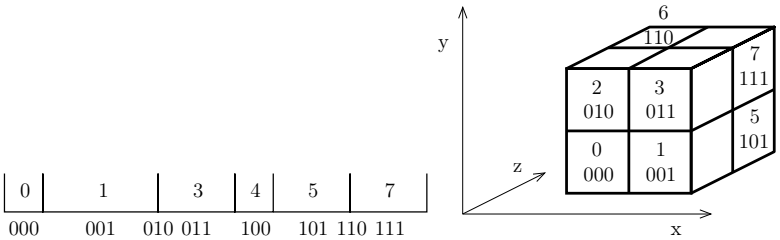


Fig. 4.3: For easy access the nodes are indexed and the points are stored consecutively. Empty voxels (index 2 and 6 in the example) have the same end pointer as their predecessors.

node consists only of a pointer to the data. The first entry of the data stream stores the number of points, followed by the point data, i.e., the x, y and z coordinates of the points as well as any other attributes the points may have. Fig. 4.2(b) illustrates the concept. The root node has three valid children, one of which is a leaf. The child pointer references the first child, which has five children of its own. The second node is the leaf that contains only the pointer to the data. The third child is again an intermediate node with three children. In [69] it is shown that this efficient representation of the octree allows to store up to one billion points in 4 GB of main memory.

The octree divides the space containing the 3D points into cubes of equal size. To reduce the overhead that is generated from a large amount of empty nodes three stopping rules are commonly used during the creation of an octree. First, the creation of the octree is stopped as soon as a predefined height of the tree is reached. Second, if a node contains less than a predefined number of points e, it is no further split. Third, if the cube represented by the nodes reaches a minimum size v, the branching stops.

In the implementation described in Algorithm 1 the third stopping rule is used while the second one is optional. The octree is created from a list of points m. The point type, i.e., whether the point contains further information, such as color, reflectivity or temperature, is

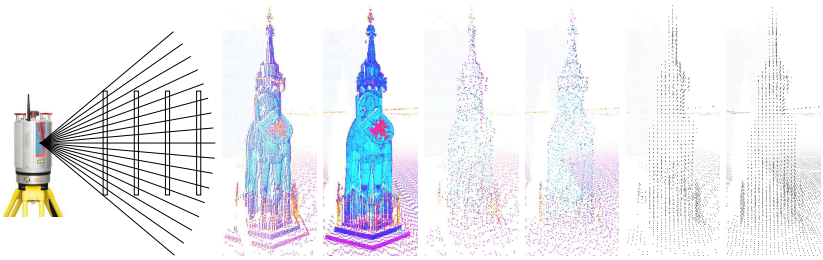
stored in the root node. During the initialization the stopping rules are defined and the center as well as the size of the axis aligned bounding box containing all points is calculated. To have a cubic bounding box, the size of the bounding box in the direction of the largest extent of the point data is chosen. The branching of the octree is done recursively.

Algorithm 1 Creating an octree

Input: point cloud m with N points, the minimum voxel size v and an early-stop option
Output: the constructed octree
Octree(m, N, v, $e = 1$)
store point type
calculate axis aligned cubic bounding box: center, size
voxel size $\leftarrow v$
early-stop $\leftarrow e$
Branch(m, N, size, center)

Branch(m, N, **size, center**)
blocks $\leftarrow$ pointer to first point of m
sort list of points according to split-axes, first based on z, then y, last x
store pointer to each end of each block into blocks
for $i = 0$ **to** 7 **do**
  if blocks[$i + 1$] - blocks[i] > 0 **then** # block is not empty
    set ith bit of parent.valid to 1
  end if
end for
create node for each valid child and link child pointer of parent
child size = 0.5· size
calculate child center
for all valid child nodes i **do**
  if child size $<$ voxel size **or** blocks[$i + 1$] - blocks[i] $\leq$ early-stop **then**
    set ith bit of parent.leaf to 1 # leaf node
    write N to beginning of data
    append points to data
  else
    set ith bit of parent.leaf to 0 # inner node
    Branch(m[blocks[i]], blocks[$i + 1$] - blocks[i], child size,child center[i])
  end if
end for

The original point data is sorted based on its assignment to the child cubes as depicted in Fig. 4.3. For this purpose the points are first sorted based on the z value. Points with a z coordinate smaller than the center of the cube are moved to the beginning of the list while those with a z coordinate larger than the center are moved to the end. Then both parts are divided based on the y coordinate and the resulting four parts again based on the x coordinate in the same manner, resulting in the order depicted in Fig. 4.3. The end pointers to each section are



(a) Schematic (b) `scan000` (c) `scan012` (d) `scan000` $v = 10$ (e) `scan012` $v = 10$ (f) `scan000` $v = 10$, c (g) `scan012` $v = 10$, c

Fig. 4.4: (a) Schematic of the decreasing point cloud density based on the distance to the sensor. Due to the angular sweeping of the laser beam with increasing distance the density decreases, i.e., a wall of identical size is sampled with fewer points. (b)+(c) The Bremen Roland seen from two positions in the Bremen City data set. `scan000` has a distance of $\approx 65\,\text{m}$ to the statue while `scan012` has a distance of $\approx 33\,\text{m}$. (d)+(e) The scans reduced to a voxel size of 10 cm. One point per voxel is randomly chosen. (f)+(g) The center of each occupied voxel is chosen during reduction.

stored for further processing. If a cube contains no points, the end pointer for the corresponding section is identical to the end pointer of the previous section. For all cubes containing points the corresponding bit of the valid pointer in the parent node is set to 1. The nodes for these cubes are generated in a sequential space in memory and linked to by the parent node. Their size equates to half the size of the parent node and their new center points are calculated by performing three steps from the center of the parent cube, one step in the direction of each axis of the coordinate system. Each step has a length of half the size of the child cube. All combinations of steps in the positive and negative direction for the three axes result in the eight child centers. The branching continues for each of the child nodes with the points from the corresponding section. Once a child node meets one of the stopping criteria, i.e., the section in the point list contains fewer points than the threshold e or the cube size falls below the voxel size v no further inner nodes are created and the child is marked as a leaf node. The leaf node contains the number of points in the leaf followed by the actual point data containing all information as specified by the point type.

Due to its equal sized cubes the octree is ideally suited for certain tasks where the spatial structure of the data is important. Two examples are important within this thesis, the reduction and the visualization of point cloud data [69]. Laser scanners capture up to one million points per second. Processing this data in reasonable time is often not possible. For many tasks it is not necessary to consider each individual point. Furthermore, the recording method inherently samples surfaces with varying density dependent on their distance to the laser scanner. Surfaces which are close to the scanning position have a significantly higher density than those at a larger distance. The concept is schematically depicted in Fig. 4.4. Due to the angular sweeping of the laser beam with increasing distance the density decreases, i.e., a wall of identical size is sampled with fewer points. The concept is shown on two scans (position 0 and position 12 of

Fig. 4.5: The effect of fog onto the depth perception on one scan from the BREMEN CITY data set. Left: The points cloud without fog. Right: Exponential white fog with a density of $D = 0.015\,\mathrm{m}$.

the BREMEN CITY data set) that both contain the Bremen Roland. `scan000` has a distance of $\approx 65\,\mathrm{m}$ to the statue, while `scan012` has a distance of $\approx 65\,\mathrm{m}$, leading to a difference in density. While the densely sampled statue in Fig. 4.4(b) looks almost like a surface, the points in Fig. 4.4(c) are clearly distinguishable. When thinning out the scans using a voxel size of $\leq 10\,\mathrm{cm}$, i.e., an effective voxel size of $8.35977\,\mathrm{cm}$, the difference in density is not noticeable anymore. In Fig. 4.4(f) and (g) the center of the occupied voxels is chosen for the reduction leading to equidistant points. However, the resulting points are not original data points. For noisy data, this option smoothes the data points. For high precise data, on the other hand, it reduces the accuracy by shifting all points to the center of the voxel. Given a wall that is aligned to a plane of the coordinate system, in the worst case all points and thus the entire plane will be shifted by half the voxel size towards the center of the octree cubes. Alternatively, in Fig. 4.4(d) and (e) one random point per voxel is chosen as representative.

The visualization of point cloud data is computationally expensive as millions of points need to be transformed into the camera coordinate system and drawn on the screen. Due to the opening angle of the camera the field of view is limited. To reduce computational burdens the visible space is also commonly limited by a back plane, i.e., far away objects are not considered. For point cloud visualization this is essential as far way points would otherwise completely overshadow closer points. To achieve a depth perception in the texture-less point cloud fog is commonly used [95]. Points with increasing distance δ to the camera origin are slowly faded out with a density D according to

$$\mathrm{RGB}_{\mathrm{new}} = \mathrm{RGB}_{\mathrm{old}} \cdot e^{-D\delta} + (1 - e^{-D\delta}) \cdot \mathrm{RGB}_{\mathrm{fog}}$$

Fig. 4.5 shows the effect for white fog on black points, with the original point color $\mathrm{RGB}_{\mathrm{old}} = (0, 0, 0)$, the fog color $\mathrm{RGB}_{\mathrm{fog}} = (1, 1, 1)$ and the density $D = 0.00015\,\mathrm{cm}$. Compared to the image without fog, the three-dimensional structure of the scene appears when applying the fog to it. Given an 8 bit color representation, the last color bit is faded at a distance of $\delta_{\max} = \frac{\ln(2^{-8})}{D}$ due to fog. Thus $\delta_{\max}$ automatically becomes the back plane distance.

In computer graphics, the visible space defined by the opening angle of the virtual camera, the back plane and a front plane is called frustum. For a perspective projection the volume enclosed by the frustum is a truncated rectangular pyramid consisting of six planes. Fig. 4.6 illustrates

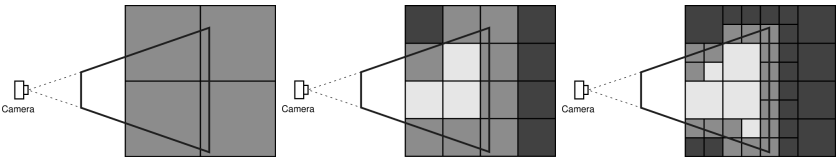


Fig. 4.6: Principle of recursive frustum culling. On each level of the octree it is decided if nodes lie completely inside (light gray), partially inside (gray), or completely outside of (dark gray) the frustum. Only for the nodes that are partially inside the frustum the culling is continued. (Source: [69])

how an octree is used to reduce the computational burden by frustum culling. Frustum culling refers to ignoring all objects that lie outside of the frustum and therefore need not to be drawn. The algorithm descends recursively into the octree. For each node a collision check with the six oriented planes of the frustum is performed. If a plane intersects with the node (gray voxels), it needs to be further analyzed, if the node lies completely outside the frustum (dark gray), it is discarded and if the node lies completely inside (white), the collision check is abandoned and all points within this voxel are drawn.

The octree is a simple data structure and it is intuitive how the data is stored. However, a lot of nodes will be empty due to the fact that only surfaces are captured. This issue is overcome by the kD-tree. In [70] different search tree implementations are compared with respect to their speed in finding nearest neighbors. It is shown that both kD-tree and octree perform well for the task, but the implementation plays an important role for the performance.

4.1.2 kD-tree

The kD-tree is a binary tree that was first developed by Bentley [14]. Its charm lies in the fact that it stores data of arbitrary dimension with little overhead and allows for efficient search strategies. Let k be the dimensionality of the data. Each level of the tree discriminates the data based on one of the k keys. Fig. 4.7 demonstrates the kD-tree using 2D points as example. Given a point cloud $D = \{\boldsymbol{p}_i = (x_i, y_i) | i = 0, 1, \ldots, N-1\} \subset \mathbb{R}$ with N points the keys correspond to the coordinates of the point, namely x_i and y_i. At level 0, the root node, and on all even levels of the tree the data set is split up based on the x coordinates, on level 1 and all subsequent uneven levels the y coordinate is the discriminator. Each node contains a point $\boldsymbol{p}_i$. The left subtree contains all points that are smaller than $\boldsymbol{p}_i$ with respect to the discriminator, the right subtree points that are larger, respectively. If x is the discriminator, then the x axis is called the split axis. The space is divided into two subspaces by a hyperplane. In the 2D case this is a line that is perpendicular to the split axis, in the 3D case a plane, respectively. An optimal kD-tree does not degenerate, i. e., the level of all leaf nodes differs by at most one. Thus the maximum path length to each node in an optimal tree containing N points is $\lfloor \log_2 N \rfloor$. The kD-tree shown in Fig. 4.7 is optimal. To build such an optimal tree, each node is assigned the point that is the median with respect to the discriminator key of the remaining points. In the example, starting from all points 'A' has the median x coordinate and is therefore inserted into the root node. 'B', 'D', 'E', and 'H' have smaller x-coordinates than 'A' and are therefore inserted into the left

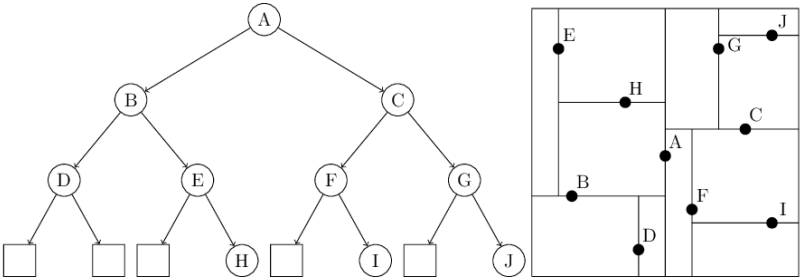


Fig. 4.7: kD-tree.

subtree, while 'C', 'F', 'G', 'I', and 'J' build the right subtree. The discriminator on the next level is the y coordinate. 'B' is the median of the left tree, whereas 'C' is the median of the right subtree, respectively. Thus these points are stored in the nodes on level one. Level two is split up based on the x coordinates again and the procedure is repeated until no points are left. Creating optimal kD-trees is costly in run time, as the median needs to be found for each node. However, if a lot of search operations are required, the speedup for these makes up for the additional creation time. Alternatively the center of the bounding box surrounding the subtree can be chosen. The split line of each node marks one side of the bounding box of its subtree. The bounding boxes are open at the border of the input space. Due to alternating splitting rules the subtrees in the 2D case need at least four ancestors to have a fully defined bounding box, e.g., the left subtree of 'H' is fully constrained while the right subtree is open to the top. The left subtree of 'I' is even open to the bottom and to the right. It is thus recommended to store the bounding volume of the entire data to speed up the search speed.

The overhead of the kD-tree depends on the number of nodes and the data that is stored in these nodes. The original concept suggests to store one point per node and the reference to the two child nodes. The split axis is derived from the depth in the tree. This means, that the number of nodes is equal to the number of points resulting in a large overhead for large point clouds. Various implementations exist that differ from the original concept proposed by Bentley [14]. The original implementation was intended to enable the retrieval of a specific data entry. In [70] different implementations are evaluated with respect to their performance of the nearest neighbor search for the iterative closest point algorithm (ICP) which will be explained in detail in Chapter 5.1. In this thesis the implementation from *The 3D Toolkit* (3DTK) [146, 149] is used, which was shown to perform well for 3D points. In this implementation the actual data is stored in the leaf nodes only. Internal nodes store the center, the side lengths and the radius of the circumscribed circle of the bounding box surrounding the contained points, the split axis and pointers to the child nodes. The plane perpendicular to the axis with the largest extent is chosen as the split axis during creation. Algorithm 2 shows the creation of such a kD-tree.

The creation of the kD-tree is recursive. The constructor is called with the list of points and the number of points. If less than 10 points are left, the current node is a leaf and all remaining points are stored in this leaf. The algorithm determines the split axis by first calculating the

Algorithm 2 Creating a kD-tree

Input: point cloud m with N points
Output: the constructed kD-tree
KDtree(m)

```
1: if N ≤ 10 then                                   # leaf node
2:    copy points to point list
3:    nrpts ← N
4: else                                             # inner node
5:    nrpts ← 0
6:    determine bounding box: center, d_x, d_y, d_z, r^2
7:    split-axis ← max(d_x, d_y, d_z)
8:    splitvalue ← center.splitaxis
9:    for all points m_i in m do
10:      if m_i.splitaxis < splitvalue then
11:         add m_i to left
12:      else
13:         add m_i to right
14:      end if
15:   end for
16:   child_left ← KDtree(left)
17:   child_right ← KDtree(right)
18: end if
```

bounding box of all points, by finding the minimum and maximum values of each coordinate. The axis with the largest extent is chosen as split axis and the center coordinate of this axis is the split value. The point cloud is then divided into two subsets, based on the split value, that are passed on to create the left and right sub trees. The asymptotic runtime for the creation of an optimal kD-tree is $\mathcal{O}(kN \log N)$. The depth of the tree is $\log N$. On each level, all N points are examined linearly to determine the center of the bounding volume for each of the k dimensions of the tree. In the root node, all points are in the same bounding box. On lower levels, the search is divided into subtrees, but the total number of points stays the same [147].

To find an entry in a kD-tree that stores all the data in the leaf nodes one has to decent into the leaf that stores the area surrounding the point. Storing the bounding box in all the inner nodes facilitates this procedure. Only the points in the corresponding leaf need to be checked. Finding the closest point, the so called nearest neighbor search (NNS), is an adaption of the simple retrieval search. In this thesis the NNS is always restricted to a certain range, i.e., for the radius nearest neighbor search only points that lie within a radius $d_{\max}$ are nearest neighbor candidates. The adaptation of the NNS is outlined in Algorithm 3.

The algorithm starts with the root node of a kD-tree and a query point $\boldsymbol{p}_q$. It descends into the tree until it reaches the leaf node v_l that would contain $\boldsymbol{p}_q$. The closest point from v_l is chosen as candidate $\boldsymbol{p}_{\text{closest}}$ and the search distance d_2 is reduced to the distance between the closest point $\boldsymbol{p}_{\text{closest}}$ and $\boldsymbol{p}_q$. By backtracking the remaining branches are examined using the ball-within bounds test, i.e., they are only considered if the sphere of radius d_2 around query

Algorithm 3 Radius NNS in a kD-tree

Input: Root t of a kD-tree, query point $\boldsymbol{p}_q$, maximum allowed distance $d_{\max}$
Output: closest point $\boldsymbol{p}_{\text{closest}}$ in the kD-tree

```
NNS()
 1: d_2 ← d_max^2
 2: p_closest ← 0
 3: t →FindClosest()
 4: return p_closest

FindClosest()
 5: if t is a leaf node then
 6:   for all points p_i in leaf do
 7:     d ← ||p_i − p_q||
 8:     if d < d_2 then                    # point is closer than previously found point
 9:       p_closest ← p_i                  # update closest point
10:       d_2 ← d
11:     end if
12:   end for
13: else                                   # inner node
14:   if sphere of radius d_2 around p_q overlaps with bounding box of current tree then
15:     if p_q lies in left subtree then
16:       child_left→FindClosest()
17:       if sphere of radius d_2 around p_q overlaps with bounding box of right subtree then
18:         child_right→FindClosest()
19:       end if
20:     else
21:       child_right→FindClosest()
22:       if sphere of radius d_2 around p_q overlaps with bounding box of left subtree then
23:         child_left→FindClosest()
24:       end if
25:     end if
26:   end if
27: end if
```

point $\boldsymbol{p}_q$ overlaps with the bounding box of the node, thus limiting the search path to those nodes that could contain a point closer to the current candidate. Thus the algorithm performs in logarithmic time [79].

Algorithm 4 Fixed range search between two points in a kD-tree

Input: Root t of a kD-tree, start point $\boldsymbol{p}_s$, end point $\boldsymbol{p}_e$, maximum allowed distance $d_{\max}$
Output: list of points $\boldsymbol{D}_{\text{closest}}$ between $\boldsymbol{p}_s$ and $\boldsymbol{p}_e$ in the kD-tree

FixedRangeSearchBetween2Points()

$d_2 \leftarrow d_{\max}^2$
$\boldsymbol{D}_{\text{closest}} \leftarrow 0$
$\boldsymbol{v}_{s,e} \leftarrow \frac{\boldsymbol{p}_e - \boldsymbol{p}_s}{\|\boldsymbol{p}_e - \boldsymbol{p}_s\|}$
$d_{s,e} \leftarrow \|\boldsymbol{p}_e - \boldsymbol{p}_s\|$
$t \rightarrow$RecursiveFixedRangeSearchBetween2Points()
return $\boldsymbol{D}_{\text{closest}}$

RecursiveFixedRangeSearchBetween2Points()

$r_t \leftarrow$ radius of circumscribed sphere of the node
if t is a leaf node **then**
  for all points $\boldsymbol{p}_i$ in leaf t **do**
    $\boldsymbol{v}_{i,s} \leftarrow \boldsymbol{p}_s - \boldsymbol{p}_i$
    $d_{i,s} \leftarrow \|\boldsymbol{v}_{i,s}\|$
    $d_i \leftarrow d_{i,s}^2 - (\boldsymbol{v}_{i,s} \cdot \boldsymbol{v}_{s,e})^2$
    if $d_i < d_2$ **then** # point lies in the cylinder of radius $d_{\max}$ connecting $\boldsymbol{p}_s$ and $\boldsymbol{p}_e$
      add $\boldsymbol{p}_i$ to $\boldsymbol{D}_{\text{closest}}$
    end if
  end for
else # inner node
  $\boldsymbol{v}_{t,s} \leftarrow \boldsymbol{p}_s - \boldsymbol{c}_i^t$ # vector node center to start point
  $d_{t,s} \leftarrow \|\boldsymbol{v}_{t,s}\|$
  $d_t \leftarrow d_{t,s}^2 - (\boldsymbol{v}_{t,s} \cdot \boldsymbol{v}_{i,s})^2$
  if $d_t > (r_t + d_{\max})^2$ **then** # circumscribed sphere of t does not intersect with cylinder
    return
  end if
  $\boldsymbol{v}_{t,e} \leftarrow \boldsymbol{p}_e - \boldsymbol{c}_t$ # vector from node center to end point
  $d_{t,e} \leftarrow \|\boldsymbol{v}_{t,s}\|$
  if $d_{t,e} > d_{s,e} + r_t$ **then** # circumscribed sphere of t is further away from $\boldsymbol{p}_e$ than $\boldsymbol{p}_s$
    return
  end if
  if $d_{t,s} > d_{s,e} + r_t$ **then** # circumscribed sphere of t is further away from $\boldsymbol{p}_s$ than $\boldsymbol{p}_e$
    return
  end if
  child_left$\rightarrow$RecursiveFixedRangeSearchBetween2Points()
  child_right$\rightarrow$RecursiveFixedRangeSearchBetween2Points()
end if

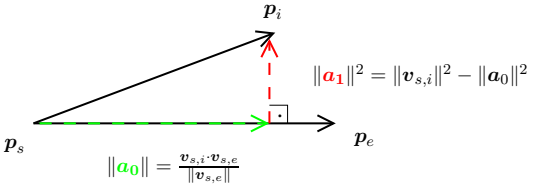


Fig. 4.8: Orthogonal projection $\boldsymbol{a}_0$ of $\boldsymbol{v}_{s,i}$ on $\boldsymbol{v}_{s,e}$ and rejection $\boldsymbol{a}_1$ of $\boldsymbol{v}_{s,i}$ from $\boldsymbol{v}_{s,e}$.

A laser scanner sees only a small glimpse of the entire environment. In all directions only the closest surface is visible. Everything behind it is occluded by the surfaces that lie in front. Consequently, it is indispensable to join data from several positions into one coordinate system to capture the scene completely. For various applications, e.g., to detect dynamic changes in the environment or to combine point cloud data with camera data, the question arises whether a point in the scene was visible from the sensor position. This is determined by checking whether there are points between the query point $\boldsymbol{p}_e$ and the sensor position $\boldsymbol{p}_s$. As point clouds are noisy and represent individual points rather than surfaces the search has to be extended to a cylinder with radius $d_{\max}$ connecting the start point $\boldsymbol{p}_s$ and the end point $\boldsymbol{p}_e$. This comes down to a special case of the search problem and is therefore solved efficiently using tree structures like the kD-tree, as outlined in Algorithm 4.

Let

$$\boldsymbol{v}_{s,e} = \frac{\boldsymbol{p}_e - \boldsymbol{p}_s}{\|\boldsymbol{p}_e - \boldsymbol{p}_s\|} \tag{4.1}$$

be the normalized direction vector from $\boldsymbol{p}_s$ to $\boldsymbol{p}_e$ and $\boldsymbol{v}_{s,i} = \boldsymbol{p}_s - \boldsymbol{p}_i$ the vector from the start point $\boldsymbol{p}_s$ to the query point $\boldsymbol{p}_i$. The squared distance d_i between $\boldsymbol{p}_i$ and the direction vector is calculated as the rejection of $\boldsymbol{v}_{s,i}$ from $\boldsymbol{v}_{s,e}$, i.e., the orthogonal projection of $\boldsymbol{v}_{s,i}$ onto the plane orthogonal to $\boldsymbol{v}_{s,e}$:

$$d_i = \|\boldsymbol{v}_{s,i}\|^2 - (\boldsymbol{v}_{s,i} \cdot \boldsymbol{v}_{s,e})^2 \,. \tag{4.2}$$

Fig. 4.8 visualizes the geometric concept. Thus, the point lies in the cylinder of infinite length if $d_i < d_{\max}^2$. A quick check is introduced to limit the search to those nodes whose circumscribed sphere intersects with the cylinder and which lie between $\boldsymbol{p}_s$ and $\boldsymbol{p}_e$. This is ensured by checking whether the distance $d_{t,s} - r_t$ between the sphere and the start point $\boldsymbol{p}_s$ is larger than the distance between start and end point and the distance $d_{t,e} - r_t$ between the sphere and the end point is larger than the distance between start and end point, respectively. $d_{t,s}$ and $d_{t,e}$ represent the distance to the center while r_t is the radius of the sphere.

4.1.3 Panorama images

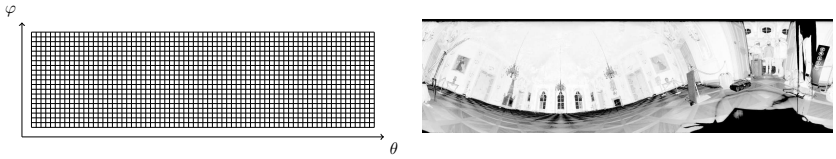
Range data as acquired by a 3D camera or a laser scanner is often considered not to be truly three-dimensional, but to have 2.5 dimensions instead. This is due to the fact that only the surface of objects are represented. For depth cameras this is quite obvious, as one essentially

Fig. 4.9: The scanned environment corresponds to a partial sphere, restricted by the field of view of the scanner. The scanner is located in the center.

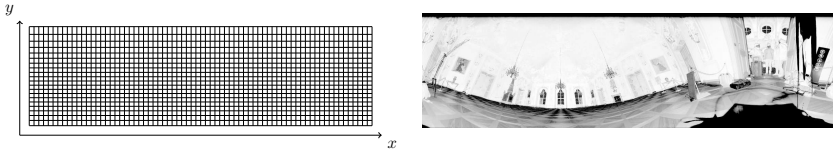
has a 2D image of depth values. The same holds true for laser scanners, however. Essentially the laser scanner sweeps its environment spherically as depicted in Fig. 4.9. For each spherical point the depth value is determined. Unwrapping such a sphere is the same problem as drawing a world map. For this purpose different panorama projection methods have been proposed. In [107] and [106] several methods are presented and evaluated for the representation of 3D laser scans. Fig. 4.10 shows the three projection methods used in this thesis.

The advantage of panorama representation methods is that 3D data is represented in an easily accessible 2D data structure that reflects the data acquisition process. This way neighborhood relationships are immediately available enabling fast calculations, e.g., for normal calculations or jump edge detection. The normal of a point is calculated by taking into account the neighboring points and by determining the direction of the surface spanned by these points. Parts of the scanned surface that are not visible by the scanner lead to a discontinuity in the visible parts. The virtual edges between those disconnected parts are called jump edges. This is exploited in the Region Growing approach for plane detection explained in Section 4.3.2. Furthermore image based methods, such as feature detection (cf. Chapter 4.2 and 5.3), are thus applicable to the 2D representation of the laser scan.

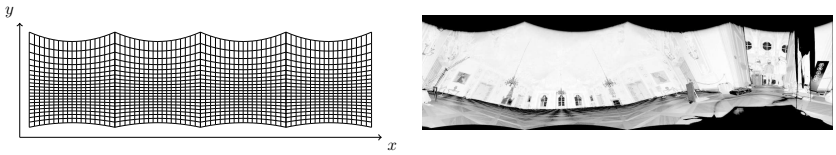
The simplest projection method is the equirectangular or rectangular projection. Due to its simplicity it is commonly used. The latitude and longitude of the sphere are directly transformed into the horizontal and vertical coordinates of a grid. No further scaling or transformations are applied. This leads to a deformation of the original data. Areas close to the poles are stretched compared to those close to the equator. Accordingly also horizontal lines, except the equator, become curved whereas vertical lines remain vertical and straight. The equirectangular projection can map the full sphere, i.e., 360° horizontally and 180° vertically. Fig. 4.9 illustrates the equirectangular projection using the field of view of the Riegl VZ-400 Laser scanner, that is restricted to 100° horizontally. For visibility the angular resolution is set to 5° both horizontally and vertically. The longitude θ and the latitude φ in spherical coordinates are transformed into



(a) Equirectangular



(b) Mercator



(c) Pannini

Fig. 4.10: Example of the different projection methods. On the left are the latitude and longitude lines as they appear in the projection with a resolution of 5°. The field of view is $0° \leq \theta \leq 360°$ horizontally and $-40° \leq \varphi \leq 60°$ vertically as for the Riegl VZ-400. On the right is the first scan from the White Hall data set as panorama image. Note that the scanner was tilted for better view of the ceiling, thus the edge between wall and floor is curved in all projections.

images coordinates according to the transformation equations:

$$x = \theta$$
$$y = \varphi.$$

The equirectangular projection leads to strong distortions. This implicates that the features vary based on the location in the image. The Mercator projection shows less pronounced distortions. It is a conformal isogonic projection methods, i.e., it preserves angles and the local scale in every direction around any point is constant. The vertical stretching is reduced by using a logarithmic scaling for the latitude mapping:

$$x = \theta$$
$$y = \ln\left(\tan\varphi + \frac{1}{\cos\varphi}\right).$$

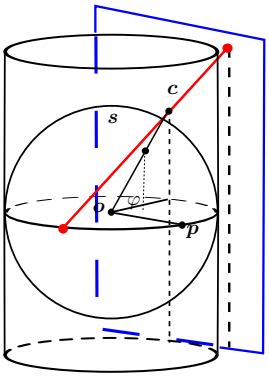


Fig. 4.11: Schematic of the construction rule for the Pannini projection. The sphere point $\boldsymbol{s}$ is first mapped onto the point $\boldsymbol{c}$ on the cylinder and then projected onto the tangent plane. This corresponds to the rectilinear projection of the cylindrical projection of the sphere. $\boldsymbol{p}$ is the point corresponding to the spherical coordinates $(\theta = 0, \varphi = 0)$. (Source: [107])

The Mercator projection is capable of projecting the full 360° horizontal field of view. It is only recommended to be used for a vertical field of view of up to 150°, thus not ideal for all 3D laser scanners. For example, the Z+F Imager 5010 with a FOV of 360° × 320° exceeds this requirement. Note, that the effective vertical field of view is only 160°.

The Pannini projection is a mathematical rule for constructing perspective images, derived from 18th century paintings, where artists painted wide architectural scenes without visible distortions [181]. One of the most famous *vedutisti* (view painters) is Giovanni Paolo Pannini in whose memory the projection method is named. Other common names are *Verdutismo* or *Recti-Perspective*. The construction consists of two steps that are depicted in Fig. 4.11. First, a spherical image is projected onto a cylinder. This projection is then again projected onto a tangent plane. The center of the first projection is in the center of the sphere. The center of the second projection is on the view axis at a distance $d \geq 0$ from the cylinder axis. The construction method is given as:

$$\begin{aligned} x &= \frac{(d+1)\sin(\theta-\theta_0)}{d+\sin\varphi_1\tan\varphi+\cos\varphi_1\cos(\theta-\theta_0)} \\ y &= \frac{(d+1)\tan\varphi\left(\cos\varphi_1-\sin\varphi_1\left(\frac{1}{\tan\varphi}\right)\cos(\theta-\theta_0)\right)}{d+\sin\varphi_1\tan\varphi+\cos\varphi_1\cos(\theta-\theta_0)}, \end{aligned} \tag{4.3}$$

where θ and φ define the point that is to be projected whereas θ_0 and φ_1 define the projection. θ_0 and φ_1 are the polar coordinates of the inflection point, i.e., the point where the tangent plane touches the cylinder. Varying values for d give different projections. $d = 0$ yields the rectilinear projection while $d \to \infty$ is a cylindrical orthographic projection. $d = 1$ is defined as

the Pannini projection. It produces images with a single central vanishing point thus appearing as the correct perspective even for a large field of view. Straight vertical lines as well as radial lines through the center of the image remain straight. Horizontal lines become curves. As the recommended field of view of the Pannini projection is less than 150° for the horizontal and the vertical direction the data has to be divided into smaller subsets horizontally to cover the entire 360°, thus several values for θ_0 are applied in equation (4.3).

In Fig. 4.10 the three aforementioned projection methods are used on the same scan from the White Hall data set. As the scanner was mounted with a tilt on the robot to enable the view of the ceiling the effects of the different methods are clearly visible. The horizontal edge between walls and floor is curved in all projections, however the effect is the most pronounced in the equirectangular projection and the least pronounced in the Pannini projection, respectively. For the Pannini projection the data is split into four parts. The more parts the weaker is the distortion on the edges. However, there is always a discontinuity between the parts. Even though the Pannini projection looks the most realistic within one subimage, the discontinuities on the edges are a major drawback. This will be discussed in more detail for image features in the next section.

4.2 Image features

One of the most important aspects in image analysis is feature detection. Features are the key to recognition tasks such as object detection, scene classification or place recognition. Consequently, they are also essential for joining multiple images to get a more complete view of a scene. There are two related tasks that deal with this problem. First, in panorama generation multiple images are joined to increase the field of view of the camera. To avoid parallax errors, all images should be taken at the same position while the orientation changes. Second, multi-view stereo refers to the reconstruction of the 3D geometry based on triangulation (cf. Chapter 2.5). Instead of using a sender and a receiver several camera positions are used to determine the disparities. Due to the fact that the baseline, i.e., the translation between the individual images is essential for determining the 3D geometry, the term structure from motion (SfM) is often used to refer to this technique. Instead of a fixed setup with known rotational and translational offsets between the hardware components, in this scenario the different poses of the same camera, the offsets, are determined by discovering and matching features that appear in more than one image.

Various image features exist based on shape, color, size or more complex attributes. For matching features reliably they have to be distinguishable and retrievable under changing conditions. Color is a feature that is not only sensitive to differences in illumination and depends on the processing in the camera but lacks a clear measure of similarity. Euclidean distance in the RGB space and distance measures based on the HSV color space yield different results. The size varies with distance to an object. The shape is distorted when changing the viewpoint. Commonly used features such as lines and corners are simple but lack distinctiveness. For feature recognition invariance to image scaling and rotation as well as to changes in illumination and viewpoint are indispensable. For this reason Lowe introduced the Scale Invariant Feature Transform (Sift) that produces a large number of features that are well localized and highly distinctive [129]. The approach uses gray-scale images and consist of four main steps, namely scale-space extrema

(a) Original (b) $L(x, y, 1.6)$ (c) $L(x, y, \sqrt{2} \cdot 1.6)$

(d) DoG: (a) - (b) (e) DoG: (b) - (c) (f) Sift features

Fig. 4.12: Difference of Gaussian (DoG) as foundation for Sift features. The original 972×972 gray-scale image (a) is convoluted with a Gaussian kernel with $\sigma = 1.6$ (b) and $\sigma = \sqrt{2} \cdot 1.6$ (c). The gray-scale of the two difference images (d) and (e) is scaled for better visibility. The darker the lines, the larger the difference. With the variation in the smoothing parameter σ the edges are differently pronounced in the difference images.

detection, key-point localization, orientation assignment, and key-point descriptor generation.

In the first step key-point candidates are determined by detecting extrema over the entire image at various scales. To achieve scale invariance a difference-of-Gaussian (DoG) pyramid is generated from the gray-scale image. First the input image $I(x, y)$ is convoluted with a variable-scaled Gaussian

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

to yield the convoluted image

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

where $*$ is the convolution operation. This is essentially a smoothing operation. Using different scales σ for the Gaussian convolution kernel yields different results especially in areas with high image gradients. Thus, computing the difference of two convoluted images

$$D(x, y, \sigma, k) = L(x, y, k\sigma) - L(x, y, \sigma)$$

results in an image where the edges of the original image are pronounced. This is depicted in Fig. 4.12. From two examples it is already clear that also the difference images vary significantly depending on σ. To build a DoG pyramid at each level or octave of the pyramid the original image is convoluted several times. Let s be the number of intervals in each octave, then $s + 3$ smoothed images are required. The first image I_0 is the original image. The remaining images I_i with $i \in [1, \dots s + 2]$ are generated using $G(x, y, k_i\sigma)$ with $k_i = 2^{\frac{i-1}{s}}$. The adjacent images are used to create the $s + 2$ DoG images. For the next octave of the pyramid the second Gaussian image to the top of the current octave with the smoothing factor 2σ is re-sampled. Taking every second pixel per row and column yields the initial image for the next octave.

Once the DoG pyramid is built local extrema over all scales are detected. The top and bottom layer of each octave is ignored so s images are examined. A pixel is a maximum if it is larger than all its eight neighbors on the same level, all nine neighbors on the level above and all nine neighbors on the level below or a minimum if it is smaller than all 26 neighbors, respectively.

The maxima and minima over all scales are selected as key-point candidates. These candidates are filtered as explained in detail in [129] to achieve higher stability. The location with subpixel accuracy is determined by using the Taylor expansion. The derivatives are approximated based on intensity differences with neighboring points. The same function allows to reject unstable extrema in areas with low contrast. As seen in Fig. 4.12 edges are strongly pronounced in the DoG function. However, edges are weak features as small noise causes their position to shift along the edge. Thus, to eliminate edge responses the principal curvature of a feature is computed using a Hessian matrix. A high ratio between the Eigenvalues of the Hessian matrix characterizes edge responses.

Once the exact position of the key-point is known and weak candidates have been removed the main orientation of the key-point needs to be found to ensure rotation invariance. To this end the gradient orientation of a circular window around the key-point is evaluated. For each key-point the Gaussian smoothed image with the scale that is closest to the scale of the key-point is chosen. For all sample points (x, y) within the circular window around the key-point the magnitude $m(x, y)$ and the orientation $\theta(x, y)$ of the gradient are pre-computed using pixel differences:

$$m(x, y) = \sqrt{(L(x + 1, y) - L(x - 1, y))^2 + (L(x, y + 1) - L(x, y - 1))^2}$$
$$\theta(x, y) = \tan^{-1}\left(\frac{L(x + 1, y) - L(x - 1, y)}{L(x, y + 1) - L(x, y - 1)}\right).$$

The image gradients are illustrated in Fig. 4.13. To calculate the main orientation of the key-point the orientations of the sample points are weighted by their magnitude and a Gaussian weight based on the distance to the key-point and inserted into a orientation histogram. The σ for the Gaussian weight equals 1.5 times the scale of the key-point and the histogram contains 36 bins covering 360°. For the highest peak in the histogram and all peaks with a value of more than

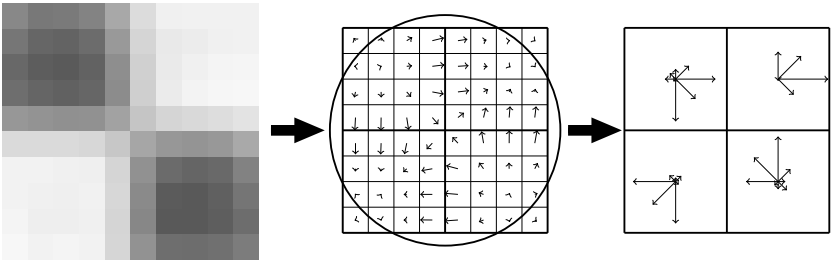


Fig. 4.13: Illustration of the creation of a Sift descriptor from a gray scale image using local image gradients. The key-point is located at the center of the patch. For clarity the alignment to the main gradient orientation has been omitted.

80% of the highest peak a key-point is created at that location with the orientation calculated from interpolating the peak bins and their neighbors.

After determining the scale and the orientation of the key-point a description of the key-point is necessary that enables fast and reliable feature matching. The gradient information is the key concept of the Sift features and needs to be represented in the descriptor. Fig. 4.13 shows the concept of the descriptor. First, the patch around the key-point is rotated to align with the main orientation of the key-point. This step is neglected in Fig. 4.13 for clarity. Then the area is divided into sample regions. Here a 2×2 sample region is chosen. From the sample points that fall into each region an orientation histogram is created. Here 8 orientation bins are used. The gradients of the sample points are inserted into the bins using tri-linear interpolation, i.e., their magnitude is added to the two closest bins weighted by the distance to the orientation represented by the bin and a Gaussian weighting function with σ equal to half the width of the window to increase the influence of points near the key-point. The sample points are the pixel centers. In practice both the subpixel location of the key-point as well as the alignment to the main orientation need to be taken into consideration when generating the key-point descriptor. The descriptor data is stored in a $w \times h \times a$ dimensional feature vector, where $w \times h$ describes the division of the sample regions and a describes the subsampling of the orientation histogram. For the example shown in Fig. 4.13 a $2 \times 2 \times 8$ feature vector is used while for the original implementation a $4 \times 4 \times 8$ feature was found to be a good trade-off between computational costs and performance. By normalizing the feature vector it becomes invariant to changes in illumination. Reconsidering the projection methods in the previous section, it becomes clear, that the discontinuities at the edges for the Pannini projection influence the Sift descriptor when gradients change directions.

An example of Sift feature locations computed using OpenCV is given in Fig. 4.12. The size of the circle indicates the scale while the small line gives the main orientation of the feature. The application of these features for matching and recovering the geometry of a scene is described in section 5.3 and 8.5. The next section describes an important 3D feature, namely planes.

4.3 Plane detection

3D point cloud acquisition and processing is subject to sensor noise and systematic errors. Scan registration is subject to matching errors. The structure of indoor environments usually comprises a large amount of planar surfaces. This can be exploited to reconstruct the real structure of the environment. Finding a planar structure in the data helps to accomplish several tasks. In this thesis it is used to determine the geometric calibration between laser scanners and cameras (cf. Chapter 7) and for the reconstruction of interiors in Section 8.1.

Plane extraction, or plane fitting, is the problem of modeling a given 3D point cloud as a set of planes that ideally explain every data point. The RANSAC algorithm is a general, randomized procedure that iteratively finds an accurate model for observed data that may contain a large number of outliers [72]. Schnabel et al. have adapted RANSAC for plane extraction and found that the algorithm performs precise and fast plane extraction, but only if the parameters have been fine-tuned properly [177]. For their optimization they use knowledge that is not readily available in point cloud data, such as normals, neighboring relations and outlier ratios. Bauer and Polthier use the radon transform to detect planes in volume data [13]. The idea and the speed of the algorithm are similar to that of the Standard Hough Transform. Poppinga et al. propose an approach to find planes with a combination of region growing and plane fitting [161]. Other plane extraction algorithms are highly specialized for a specific application and are not in widespread use. Lakaemper and Latecki use an Expectation Maximization (EM) algorithm to fit planes that are initially randomly generated [121], Wulf et al. detect planes relying on the specific properties of a sweeping laser scanner [207] and Yu et al. developed a clustering approach to solve the problem [213].

Approaches that work on triangle meshes ask for pre-processing of the original point data. Attene et al. fit geometric primitives into triangle meshes [9]. The proposed prototype works for planes, cylinders and spheres but is easily extensible to other primitives. Starting from single triangles they extend the cluster into the direction that is best represented by one of the primitives.

In [69] we presented a RANSAC approach for detecting planes using the octree data structure presented in chapter 4.1.1. In [32] we evaluated different variants of the Hough Transform [103] against the plane detection methods from [161] and [9]. After an introduction to the mathematical description of planes this chapter describes first Region Growing (RG), then RANSAC and finally the Hough Transform approach [28, 32].

4.3.1 Plane description

Planes are commonly represented by the signed distance ρ to the origin of the coordinate system and the slopes m_x and m_y in direction of the x- and y-axis, respectively:

$$z = m_x x + m_y y + \rho.$$

To avoid problems due to infinite slopes when trying to represent vertical planes, vector descriptions are used. A line in parametric form is given as

$$L(t) = \boldsymbol{c}_m + t\boldsymbol{d},$$

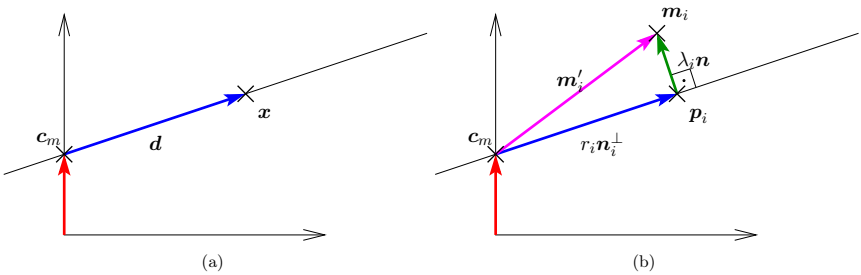


Fig. 4.14: From line equation (a) to distance to the plane (b).

where $\boldsymbol{c}_m$ is a point on the line and $\boldsymbol{d}$ the direction of the line as illustrated in Figure 4.14(a). For the parametric plane equation in 3D a second direction vector $\boldsymbol{e}$ is added:

$$P(t, s) = \boldsymbol{c}_m + t\boldsymbol{d} + s\boldsymbol{e}. \tag{4.4}$$

The Hesse normal form uses normal vectors. A plane is thereby given by a point $\boldsymbol{p}$ on the plane, the normal vector $\boldsymbol{n}$, a unit length vector, that is perpendicular to the plane and the distance ρ to the origin

$$\rho = \boldsymbol{p} \cdot \boldsymbol{n} = p_x n_x + p_y n_y + p_z n_z. \tag{4.5}$$

The normal vector is calculated from the direction vectors as

$$\boldsymbol{n} = \frac{\boldsymbol{d} \times \boldsymbol{e}}{|\boldsymbol{d} \times \boldsymbol{e}|}.$$

Considering the angles between the normal vector and the coordinate system in spherical coordinates, the coordinates of $\boldsymbol{n}$ are factorized to

$$p_x \cdot \cos\theta \cdot \sin\varphi + p_y \cdot \sin\varphi \cdot \sin\theta + p_z \cdot \cos\varphi = \rho, \tag{4.6}$$

with θ the angle of the normal vector on the xy-plane and φ the angle between the xy-plane and the normal vector in z direction as depicted in Fig. 4.15. To define a plane based on points a minimum of three points is necessary. The plane equation for the plane spanned by the three points $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ is calculated as

$$\rho = \boldsymbol{n} \cdot \boldsymbol{p}_1 = \frac{(\boldsymbol{p}_3 - \boldsymbol{p}_2) \times (\boldsymbol{p}_1 - \boldsymbol{p}_2)}{|(\boldsymbol{p}_3 - \boldsymbol{p}_2) \times (\boldsymbol{p}_1 - \boldsymbol{p}_2)|} \cdot \boldsymbol{p}_1. \tag{4.7}$$

The polar coordinates are derived from the unit length normal vector $\boldsymbol{n}$ as follows. φ is calculated from n_z:

$$\varphi = \cos^{-1} n_z. \tag{4.8}$$

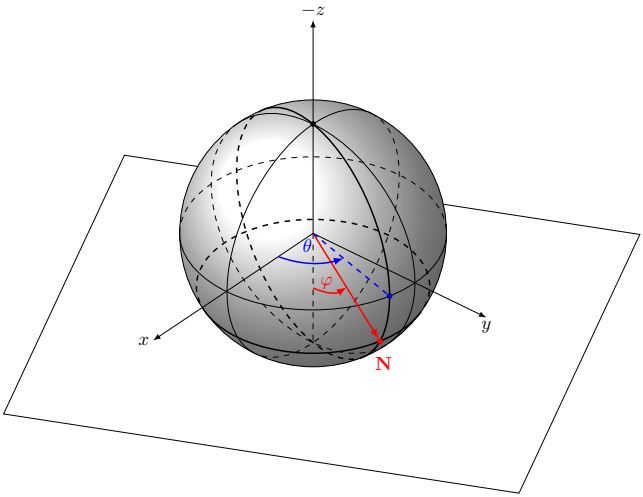


Fig. 4.15: Normal vector described by polar coordinates.

Next, the border cases are handled:

$$\theta = \begin{cases} 0, & \text{if} \quad \varphi = 0 \\ 0, & \text{if} \quad \varphi = \pi \\ 0, & \text{if} \quad \frac{n_x}{\sin(\varphi)} = -1 \\ \pi, & \text{if} \quad \frac{n_x}{\sin(\varphi)} = 1. \end{cases} \tag{4.9}$$

For the remaining cases a distinction of cases handles the disambiguity of the cosine to calculate:

$$\theta = \begin{cases} \cos^{-1}(\frac{n_x}{\sin(\varphi)}), & \text{if} \quad \sin(\theta) = \frac{n_y}{\sin(\varphi)}, \\ 2\pi - \cos^{-1}(\frac{n_x}{\sin(\varphi)}), & \text{else.} \end{cases} \tag{4.10}$$

Point cloud data is noisy. Thus calculating a plane from three points from a point set will not yield the best representation of the data. Instead, plane fitting is required. Plane fitting is the problem of determining the best fit plane from a set of points, i.e., the plane for which the sum of squared distances of all points is minimal.

Combining equations (4.4) and (4.5) yields that for each point $\boldsymbol{p}_i$ on the plane

$$\boldsymbol{n} \cdot (\boldsymbol{p}_i - \boldsymbol{c}_m) = 0$$

holds true.

Let $\boldsymbol{m}_i$ be the sample points that can be described as a linear combination of the point on the plane $\boldsymbol{c}_m$, the normal vector $\boldsymbol{n}$ with the perpendicular distance to the plane $\lambda_i = \boldsymbol{n} \cdot (\boldsymbol{m}_i - \boldsymbol{c}_m)$ and a unit length vector $\boldsymbol{n}_i^\perp$ that is perpendicular to $\boldsymbol{n}$, i.e. lies on the plane with the appropriate coefficient r_i as illustrated in Fig. 4.14(b):

$$\boldsymbol{m}_i = \boldsymbol{c}_m + r_i \boldsymbol{n}_i^\perp + \lambda_i \boldsymbol{n}$$

Defining $\boldsymbol{m}_i' = \boldsymbol{m}_i - \boldsymbol{c}_m$, $\lambda_i \boldsymbol{n}$ is the orthogonal projection of $\boldsymbol{m}_i'$ on the normal vector, i.e., the vector between $\boldsymbol{m}_i'$ and its orthogonal projection $\boldsymbol{p}_i$ on the plane. Thus the squared distance

$$\lambda_i^2 = (\boldsymbol{n} \cdot \boldsymbol{m}_i')^2$$

is used to define the square error of all points:

$$\begin{aligned}
E &= \sum_{i=1}^{N} \lambda_i^2 = \sum_{i=1}^{N} (\boldsymbol{n} \cdot \boldsymbol{m}_i')^2 \\
&= \sum_{i=1}^{N} n_x^2 m_{x,i}'^2 + n_y^2 m_{y,i}'^2 + n_z^2 m_{z,i}'^2 + 2 n_x m_{x,i}' n_y m_{y,i}' + 2 n_x m_{x,i}' n_z m_{z,i}' + 2 n_y m_{y,i}' n_z m_{z,i}' \\
&= \sum_{i=1}^{N} \boldsymbol{m}_i'^T (\boldsymbol{n}\boldsymbol{n}^T) \boldsymbol{m}_i' = \sum_{i=1}^{N} \boldsymbol{n}^T (\boldsymbol{m}_i' \boldsymbol{m}_i'^T) \boldsymbol{n} \\
&= \underbrace{\boldsymbol{n}^T \left(\sum_{i=1}^{N} \boldsymbol{m}_i' \boldsymbol{m}_i'^T \right) \boldsymbol{n}}_{1.} = \underbrace{\boldsymbol{n}^T \boldsymbol{H}(\boldsymbol{c}_m) \boldsymbol{n}}_{2.}
\end{aligned}$$

Using the first form of the error function E, the partial derivative

$$\frac{\partial E}{\partial \boldsymbol{c}_m} = \sum_{i=1}^{N} \begin{pmatrix} -2n_x^2 m_{x,i}' - 2n_x n_y m_{y,i}' - 2n_x n_z m_{z,i}' \\ -2n_y^2 m_{y,i}' - 2n_x n_y m_{x,i}' - 2n_y n_z m_{z,i}' \\ -2n_z^2 m_{z,i}' - 2n_x n_z m_{x,i}' - 2n_y n_z m_{y,i}' \end{pmatrix} = -2 \left[\boldsymbol{n}\boldsymbol{n}^T \right] \sum_{i=1}^{N} \boldsymbol{m}_i'$$

is zero, whenever $\sum_{i=1}^{N} \boldsymbol{m}_i' = 0$ holds true. Thus $\boldsymbol{c}_m$ is derived from

$$\sum_{i=1}^{N} \boldsymbol{m}_i' = \sum_{i=1}^{N} (\boldsymbol{m}_i - \boldsymbol{c}_m) = -N\boldsymbol{c}_m + \sum_{i=1}^{N} \boldsymbol{m}_i \stackrel{!}{=} 0 \Leftrightarrow \boldsymbol{c}_m = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{m}_i, \tag{4.11}$$

i.e., is the average of the sample points.

Given $\boldsymbol{c}_m$ the matrix $\boldsymbol{H}(\boldsymbol{c}_m)$ in the second form of E is determined. $\boldsymbol{H}$ is the symmetric 3×3 matrix covariance that is constructed as:

$$\boldsymbol{H}(\boldsymbol{c}_m) = \sum_{i=1}^{N} \boldsymbol{m}_i' \boldsymbol{m}_i'^T = \begin{pmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{pmatrix} \tag{4.12}$$

where

$$S_{xx} = \sum_{i=1}^{N} (m_{x,i} - c_{x,m})^2,\ S_{xy} = \sum_{i=1}^{N} (m_{x,i} - c_{x,m}) \cdot (m_{y,i} - c_{y,m}), \ldots. \tag{4.13}$$

Finding the minimum of the quadratic form

$$f(\boldsymbol{n}) = \boldsymbol{n}^T \boldsymbol{H}(\boldsymbol{c}_m)\boldsymbol{n}$$

is an optimization problem subject to the equality constraint $\boldsymbol{n}^T\boldsymbol{n} = 1$. Constraint minimization is solvable using Lagrange multipliers. To minimize a function $f(x, y)$ subject to $g(x, y) = c$, where both f and g have continuous first partial derivatives the Lagrange function $\mathcal{L}(x, y, \lambda)$, using the Lagrange multiplier λ is defined by:

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda \cdot (g(x, y) - c),$$

If $f(x_0, y_0)$ is an extremum of f for the original constraint problem, then there exists λ_0 such that (x_0, y_0, λ_0) is a stationary point for the Lagrange function, i.e., the partial derivatives of $\mathcal{L}$ are zero. Note, that not all stationary points yield a solution of the original problem. To find $\min f(\boldsymbol{n})$ subject to $\boldsymbol{n}^T\boldsymbol{n} = 1$ the Lagrange function is

$$\mathcal{L}(\boldsymbol{n}, \lambda) = f(\boldsymbol{n}) - \lambda(\boldsymbol{n}^T\boldsymbol{n} - 1)$$

with the partial derivatives

$$\begin{aligned} \frac{\partial \mathcal{L}(\boldsymbol{n}, \lambda)}{\partial \boldsymbol{n}} &= 2\boldsymbol{H}(\boldsymbol{c}_m)\boldsymbol{n} - 2\lambda\boldsymbol{n} \stackrel{!}{=} 0 \\ &\Rightarrow \boldsymbol{H}(\boldsymbol{c}_m)\boldsymbol{n} = \lambda\boldsymbol{n}, \end{aligned}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{n}, \lambda)}{\partial \lambda} = 0 \Leftrightarrow \boldsymbol{n}^T\boldsymbol{n} = 1.$$

$\boldsymbol{H}(\boldsymbol{c}_m)\boldsymbol{n} = \lambda\boldsymbol{n}$ holds true for the Eigenvectors $\boldsymbol{n}_i$ of $\boldsymbol{H}(\boldsymbol{c}_m)$ with the corresponding Eigenvalues λ_i. It follows

$$\frac{\partial \mathcal{L}(\boldsymbol{n}, \lambda)}{\partial \boldsymbol{n}} = 0 \Rightarrow f_{\text{sol}}(\boldsymbol{n}) = \boldsymbol{n}^T\lambda\boldsymbol{n} = \lambda\boldsymbol{n}^T\boldsymbol{n} = \lambda.$$

Thus the solution $\boldsymbol{v}_m$ to $\min f(\boldsymbol{n})$ is the Eigenvector of $\boldsymbol{H}(\boldsymbol{c}_m)$ with the smallest Eigenvalue. The Eigenvectors and Eigenvalues are, for example, determined using the method of Jacobi [110]. Let $\boldsymbol{v}_0$, $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ be the Eigenvectors of $\boldsymbol{H}$ and λ_0, λ_1 and λ_2 be the corresponding Eigenvalues. Then the normal vector $\boldsymbol{n}$ is the Eigenvector $\boldsymbol{v}_m$ corresponding to the Eigenvalue $\lambda_m = \min(|\lambda_0|, |\lambda_1|, |\lambda_2|)$ with the minimum absolute value. The plane distance is calculated as $\rho = \boldsymbol{n}^T \cdot \boldsymbol{c}_m$. A quick-check validates the non-planarity c by calculating the ratio of the smallest Eigenvalue λ_m and the sum of Eigenvalues $c = \frac{\lambda_m}{\sum_{i=0}^{2} \lambda_i}$. The higher c, the less planar is the point cloud.

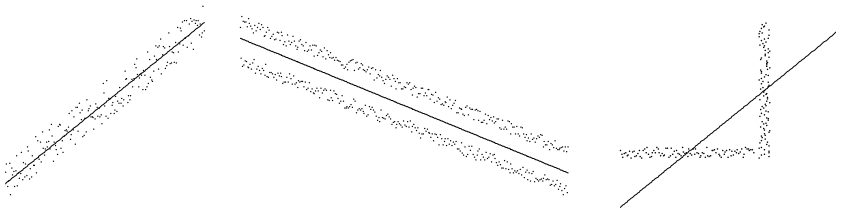


Fig. 4.16: Examples of line fitting using Eigenvectors.

Plane fitting works well if the point cloud contains a single plane. Typical point cloud data, however, represents environments with several planar structures. Fitting a plane into such data leads to undesired results as demonstrated in the line fitting examples in Figure 4.16. The fitting procedure will find the best approximation regardless of what the true shape of the data is. Thus, the results will not lead to a correct representation of the data. Consequently, one first needs to segment the data, i.e., those points that represent a planar part of the data and then fit a plane through it. There are three general approaches to this task and in the following, algorithms from all three categories will be described. First, region growing approaches like [161] start locally and add points to the current planar set while always checking for planarity on the run. Second, RANSAC-like [72, 177] approaches generate hypotheses based on a small set of points and then verify these hypotheses. Third, the evaluation of all possible plane configurations is done in the Hough Transform.

Plane refinement

Region growing approaches inherently yield connected planar patches. RANSAC as well as the Hough Transform, however, simply result in the plane parameters that are best represented in the point cloud. To calculate the borders of the plane that correspond to the actual data the following method is suggested. Given a point cloud M and a plane $P_i = (\boldsymbol{n}_i, \rho_i)$, a subset M_h of M that contains all the points with a distance of less than a threshold t to P_i. Then the refined plane parameters P_i^r are calculated by fitting a plane to M_h as described in section 4.3.1. If the planarity is sufficient the projection plane is determined. The projection plane is the plane of the coordinate system that is perpendicular to the axis corresponding to the most dominant element of the normal vector. Then the new point set M_r is created, containing all points from M with a distance of less than t to P_i^r. Now all points from M_r are first orthogonally projected onto P_i^r and then onto the projection plane. From these projected points we now want to find the largest connected region. This means we want to find the largest subset M_l in M_r so that for each point $\boldsymbol{p}_i$ in M_l there exists at least one point $\boldsymbol{p}_j$ in M_l for that the distance to $\boldsymbol{p}_i$ is below a threshold. This is achieved by generating an occupancy grid and applying a two-pass clustering algorithm [55] to the selected points. The first pass starts in the upper left corner to check 4-connectivity. For each occupied pixel it looks at the upper and the left neighbor. If they are both empty, a new region is started. If one of them is occupied, the current pixel is

assigned to the same region as the occupied neighbor. If both of them are occupied, the two regions are joined. To speed up the process the merging of regions is done by maintaining a list of linked regions. In the second pass the number of cells of each region is counted and then the counter of the linked regions are summed up. Last, all points belonging to any of the regions from the cluster are marked and removed from the point set. If the variance of the fitted plane is above a threshold, the plane is neglected. Otherwise, the convex hull of the largest cluster is calculated using the Jarvis' March convex hull algorithm [164] and the points are removed from M. The convex hull of a point set M_c is the smallest convex set containing M_c, i.e., a polygon P consisting of the minimum number of points from M_c where all points from M_c are within the bounds of P [164]. Speaking figuratively, if a rubber-band is spanned around all points from the set on a plane it will take on the shape of the convex hull of these points. Note that for this purpose the 2D projection on the plane is used. Using the convex hull has the advantage that occlusions in the point cloud are not represented in the planar reconstruction, however, it might introduce surfaces that were not present in the environment. If this is an issue, the concave hull should be calculated instead. However, as opposed to the concave hull, the convex hull is not uniquely defined. For an example refer to Section 8.1.

4.3.2 Region growing

Region growing approaches start locally and add points to the current planar set while always checking for planarity on the run. To work efficiently they require a data structure that enables the easy access of points in the neighborhood. This section describes two different approaches, one using range images [161], and one using triangle meshes [9].

The approach by [161] works for point clouds in Cartesian coordinates, which are generated from range images. It combines region growing and incremental plane fitting followed by a polygonization step. The algorithm starts with a randomly selected point from the point cloud and its nearest neighbor. Iteratively this set is enlarged by adding points that have the smallest distance to the current region, are close to the optimal plane through the points of the region, and do not lift the mean square error of the optimal plane above a threshold. Regions with too few points are discarded. To speed up the algorithm the 8 neighbors in the range image of each newly added point are inserted into a priority queue from which the next point is chosen. An update strategy for the plane fitting avoids expensive recalculations in each step.

For the update strategy the construction of the covariance matrix $\boldsymbol{H}$ needs to be analyzed and broken into its individual terms. According to equation 4.11 to 4.13 the entry S_{xy} of $\boldsymbol{H}$ is constructed as

$$\begin{aligned} S_{xy} &= \sum_{i=1}^{N} (m_{x,i} - c_{x,m}) \cdot (m_{y,i} - c_{y,m}) \\ &= \sum_{i=1}^{N} (m_{x,i}m_{y,i} - c_{x,m}m_{y,i} - c_{y,m}m_{x,i} + c_{x,m}c_{y,m}) \\ &= \sum_{i=1}^{N} m_{x,i}m_{y,i} - c_{x,m} \cdot \sum_{i=1}^{N} m_{y,i} - c_{y,m} \cdot \sum_{i=1}^{N} m_{x,i} + N \cdot c_{x,m}c_{y,m}. \end{aligned} \tag{4.14}$$

Substituting with S_x, S_y and S_z, where $S_x = \sum_{i=1}^{N} m_{x,i}$, the update strategy for deriving S_{xy}^{N+1} from S_{xy}^{N} is as follows:

$$\begin{aligned} S_{xy}^{N+1} =& S_{xy}^{N} + m_{x,N+1} m_{y,N+1} \underbrace{+ c_{x,m}^{N} S_y^{N} + c_{y,m}^{N} S_x^{N} - N \cdot c_{x,m}^{N} c_{y,m}^{N}}_{N} \\ & \underbrace{- c_{x,m}^{N+1} S_y^{N+1} - c_{y,m}^{N+1} S_x^{N+1} + (N+1) \cdot c_{x,m}^{N+1} c_{y,m}^{N+1}}_{N+1}. \end{aligned} \tag{4.15}$$

In addition to adding the product of the components $m_{x,N+1} m_{y,N+1}$ the terms depending on the mean c_m^N are subtracted and substituted with the terms for the new mean c_m^{N+1}. This reduces the costly recalculation of the sum over all points to an update with constant runtime. To enable this update strategy it is necessary to maintain the matrix $\boldsymbol{H}^N$ as well as the sum of the components S_x^N, S_y^N and S_z^N. The mean square error of the points is updated in a similar iterative manner. For this the product matrix is stored additionally:

$$\boldsymbol{P} = \sum_{i=1}^{N} \boldsymbol{m}_i \boldsymbol{m}_i^T = \begin{pmatrix} P_{xx} & P_{xy} & P_{xz} \\ P_{yx} & P_{yy} & P_{yz} \\ P_{zx} & P_{zy} & P_{zz} \end{pmatrix} \tag{4.16}$$

with

$$P_{xx} = \sum_{i=1}^{N} m_{x,i} m_{x,i},\ P_{xy} = \sum_{i=1}^{N} m_{x,i} m_{y,i}, \ldots. \tag{4.17}$$

The mean square error is calculated as

$$\begin{aligned} \mathrm{MSE}_N =& \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{n}_N \boldsymbol{m}_i - \rho_N)^2 \\ =& \frac{1}{N} \sum_{i=1}^{N} \left(\left(\sum_{a,b \in \{x,y,z\}} n_{a,N} p_a n_{b,N} p_b \right) - \left(\sum_{a \in \{x,y,z\}} 2 \rho n_{a,N} p_a \right) + \rho_N^2 \right) \\ =& \sum_{a,b \in \{x,y,z\}} \left(n_{a,N} n_{b,N} \sum_{i=1}^{N} p_a p_b \right) - \sum_{a \in \{x,y,z\}} \left(2 \rho n_{a,N} \sum_{i=1}^{N} p_a \right) + \rho_N^2 \\ =& \sum_{a,b \in \{x,y,z\}} n_{a,N} n_{b,N} P_{ab,N} - \sum_{a \in \{x,y,z\}} 2 \rho n_{a,N} S_a + \rho_N^2. \end{aligned} \tag{4.18}$$

Resorting reveals that the calculation can be simplified by use of the product matrix $\boldsymbol{P}$ and the component products S_x, S_y and S_z. Thus the computation is done in constant time.

For the detection of a calibration pattern in a point cloud in Chapter 7.1.2 the region growing approach is adopted by replacing the priority queue with a method similar to the method used for finding the largest connected region in the previous section. The advantage of region growing approaches is that they consider the connectivity of points. While this causes problems when

parts of the plane are occluded it helps to find smaller planar regions in cluttered areas. The point cloud is converted into a panorama range image using the equirectangular projection (cf. 4.1.3). From these projected points we now want to find connected planar regions. This is similar to clustering from image processing [55]. The algorithm starts in the upper left corner to check 4-connectivity. For each pixel that contains a point the upper and the left neighbors are evaluated according to two criteria. First, if the neighbor is empty or the range of the occupied neighbor differs from the range of the current pixel by more than a threshold, it is not further considered. Second, if by adding the point to the region of the neighbor, the mean square error (4.18) exceeds a threshold, the region is not further considered, either. If both neighbors are discarded, a new region is started. If one of them passes the evaluation, the current pixel is added to that region. If both of them are valid, the two regions are joined, if the resulting MSE remains below the threshold. Otherwise the point is added to the region with the lower MSE.

To enable joining of regions the previously defined iterative update operations are extended. From equation (4.14) and (4.15) the update strategy for the covariance matrix is calculated as

$$\begin{aligned} S_{xy}^{N+M} =& S_{xy}^{N} \underbrace{+c_{x,m}^{N} S_{y}^{N} + c_{y,m}^{N} S_{x}^{N} - N \cdot c_{x,m}^{N} c_{y,m}^{N}}_{N} \\ &+ S_{xy}^{M} \underbrace{+c_{x,m}^{M} S_{y}^{M} + c_{y,m}^{M} S_{x}^{M} - N \cdot c_{x,m}^{M} c_{y,m}^{M}}_{M} \\ &\underbrace{-c_{x,m}^{N+M} S_{y}^{N+M} - c_{y,m}^{N+M} S_{x}^{N+M} + (N+M) \cdot c_{x,m}^{N+M} c_{y,m}^{N+M}}_{N+M}. \end{aligned} \tag{4.19}$$

$\boldsymbol{P}$ and the new MSE_{N+M} are directly calculated. The inverse operations for removing a point and removing a region are derived from equation (4.15) and (4.19).

A second state-of-the-art method is the mesh segmentation by fitting of primitives (HFP – hierarchical fitting primitives) by [9]. The algorithm works on triangle meshes. In the beginning each triangle is considered as a cluster. Each cluster is assigned a primitive. The clusters are subsequently enlarged into the direction that is best represented by a primitive. The prototype presented in [9] supports planes, spheres and cylinders. Neighboring sets of triangles are merged into one cluster if the resulting cluster can be approximated by a primitive. The cost of merging two sets is the error of the approximation, i.e., the distance of the vertices from the primitive. Sorting the neighborhood relationships into a priority queue based on the merging costs a hierarchy is created. In each step the merging operation with the lowest cost is performed, the costs for the new cluster are recalculated and updated in the queue.

Calculating a triangle mesh from a point cloud is costly. Section 4.1.3 describes how to generate a range image from a point cloud. However, unordered point clouds that are generated from several viewing positions are not convertible into range images without loss of information.

4.3.3 RANSAC

RANSAC stands for RANdom SAmple Consensus [72]. It describes a set of randomized algorithms that finds a good configuration from a large set of possibilities. It is commonly used when a large number of outliers is present in the data. For plane detection this means that RANSAC tries to find a plane configuration that describes the data best, i.e., a plane that a sufficiently

large amount of points from the point cloud lie on. The algorithm begins by first drawing randomly a minimum sample from the data M. A minimum sample contains as many entries from the data set as necessary to determine the configuration definitely, thus three points $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ are drawn for plane detection. Then the plane $P_h = (\boldsymbol{n}_h, \rho_h)$ spanned by these points is calculated using equations (4.7) to (4.10). In the verification step, the algorithm iterates over the entire data set and creates a subset M_h that contains all the points from M with a distance of less than a threshold t to P_h. If the size of M_h is sufficiently large P_h is accepted. If this is not the case, the procedure is repeated for a finite number of sample sets and finally the hypothesis with the highest point count is chosen as representative of the data. The final result is then calculated by fitting a plane into the consensus set M_h.

This RANSAC procedure has three parameters, the required number of points to accept a hypothesis, the threshold t and the number of times new samples are randomly drawn from M. Fischler and Bolles [72] present some thoughts on how to determine the best parameters for a specific problem. They suggest to use one or two standard deviations of the average error as error tolerance t. For point cloud data the necessary t depends on the accuracy of the measuring device and is derived from the specifications. However, as stated by Fischler and Bolles, the expected error is usually relatively small compared to the error caused by outliers, thus a single approximate measure suffices for most scenarios. The number of attempts to find a consensus set is based on the expected number of trials required to find a subset that represents the plane and accordingly on the probability that a point $\boldsymbol{p}_i$ from M is within the error tolerance of a plane in the data. As this number is hard to determine for an arbitrary environment, it is advised to choose a higher number. The same holds true for the number of points that the model must consist of. Thus in the implementation from [69] this early stopping criteria is omitted.

The verification step is the part of the RANSAC algorithm that influences the runtime the most as all points in the data need to be touched. Thus, it is the step where modifications lead to the largest improvement. One way to reduce the number of points considered in the verification step drastically is presented in [69] using the octree data structure described in chapter 4.1.1. Once the plane $P_h = (\boldsymbol{n}_h, \rho_h)$ is calculated from the three points $\boldsymbol{p}_1$, $\boldsymbol{p}_2, \boldsymbol{p}_3$ the counting procedure recursively descends into the octree. While doing so, it checks if the cuboid surrounding each node intersects with the plane. Since the voxels are axis aligned, this is done by simple comparisons. Only voxels that intersect with the plane are further considered. Fig. 4.17 illustrates the effect with a simple example. On each depth level some of the voxels are discarded and in the end only for those points in the vicinity of the plane the point to plane distance has to be calculated. For a larger scene consisting of many structures and thus many outliers with respect to a single plane the improvement is even more evident.

After finding the plane P_m with the highest counter the corresponding points M_m are removed from M using the method described in Section 4.3.1 and the procedure is repeated on the remaining points.

4.3.4 The 3D Hough Transform

The Hough Transform [103] is a method for detecting parameterized objects, typically used for lines and circles. However, we focus on the detection of planes in 3D point clouds [28, 32]. Even though many Hough Transform approaches work with pixel images as input this is not

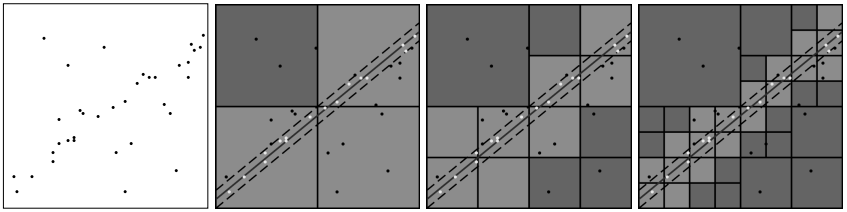


Fig. 4.17: Illustration of the speedup achieved by using an octree. Only the points in the light gray voxels are checked in the verification step. The dark gray voxels do not intersect with the line that is enlarged by a threshold. Therefore all points in these voxels are not considered in the verification step. (Source [69])

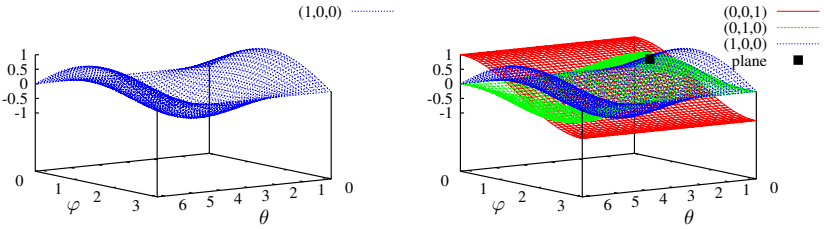


Fig. 4.18: Transformation of three points from $\mathbb{R}$ to Hough space (θ, φ, ρ). The intersection of the curves (marked in black) depicts the plane spanned by the three points.

a necessity. In our scenario a set of unorganized points in $\mathbb{R}^3$ is used as input and the output consists of parameterized planes.

φ, θ and ρ from equation (4.6) define the three-dimensional Hough Space (θ, φ, ρ) such that each point in the Hough Space corresponds to one plane in $\mathbb{R}^3$.

To find planes in a point set, one calculates the Hough Transform for each point. Given a point $\boldsymbol{p}$ in Cartesian coordinates, we have to find all planes the point lies on, i.e., find all the θ, φ and ρ that satisfy equation (4.6). Marking these points in the Hough Space leads to a 3D sinusoid curve as shown in Fig. 4.18. The intersections of two curves in Hough Space denote the planes that are rotated around the line built by the two points. Consequently, the intersection of three curves in Hough Space corresponds to the polar coordinates defining the plane spanned by the three points. In Fig. 4.18 the intersection is marked in black. Given a set P of points in Cartesian coordinates, one transforms all points $\boldsymbol{p}_i \in P$ into Hough Space. The more curves intersect in $\boldsymbol{h}_j \in (\theta, \varphi, \rho)$, the more points lie on the plane represented by $\boldsymbol{h}_j$ and the higher is the probability that $\boldsymbol{h}_j$ is actually extracted from P.

Hough methods

Standard Hough Transform For practical applications [58] propose discretizing the Hough Space with ρ', φ' and θ' denoting the extend of each cell in the according direction in Hough Space. A data structure is needed to store all these cells with a score parameter for every cell. In the following, incrementing a cell refers to increasing the score by +1. This data structure, called the accumulator, is described in more detail from page 102 onwards. For each point $\boldsymbol{p}_i$ we increment all the cells that are touched by its Hough Transform. The incrementation process is often referred to as voting, i.e., each point votes for all sets of parameters (ρ, φ, θ) that define a plane on which it may lie, i.e., if the Euclidean distance to the plane represented by the center of the cell is less than a threshold. The cells with the highest values represent the most prominent planes, the plane that covers the most points of the point cloud.

Once all points have voted, the winning planes are selected. Due to the discretization of the Hough Space and the noise in the input data it is advisable to search not only for one cell with a maximal score but for the maximum sum in a small region of the accumulator. Kiryati et al. use the standard practice for peak detection [116]. In the sliding window procedure a small three-dimensional window is defined that is designed to cover the full peak spread. The most prominent plane corresponds to the center point of a cube in Hough Space with a maximum sum of accumulation values. The steps of the procedure are outlined in Algorithm 5.

Algorithm 5 Standard Hough Transform (SHT)

```
1: for all points p_i in point set P do
2:   for all cells (ρ, φ, θ) in accumulator A do
3:     if point p_i lies on the plane defined by (ρ, φ, θ) then
4:       increment cell A(ρ, φ, θ)
5:     end if
6:   end for
7: end for
8: Search for the most prominent cells in the accumulator, defining the detected planes in P
```

Due to its high computation time (cf. page 104ff) the Standard Hough Transform is rather impractical, especially for real-time applications. Therefore numerous variants have been devised. Illingworth et al. give a survey on the early development and applications [108]. Kälviäinen et al. compare modified versions of the Hough Transform aiming to make the algorithm more practical [112]. Some of those procedures are described in the following subsections. This section concludes with an evaluation of these methods with the goal to find the optimal variation for the task of detecting a previously unknown number of planes in a 3D point cloud.

Probabilistic Hough Transform The Standard Hough Transform is performed in two stages. First, all points $\boldsymbol{p}_i$ from the point set P are transformed to Hough Space, i.e., the cells in the accumulator are incremented. This needs $\mathcal{O}(|P| \cdot N_\varphi \cdot N_\theta)$ operations, where N_φ is the number of cells in direction of φ, N_θ in direction of θ and N_ρ in direction of ρ, respectively when calculation a single ρ corresponding to the plane on which a point $\boldsymbol{p}_i$ is, given $\boldsymbol{p}_i$, φ and θ. Second, in a search phase the highest peaks in the accumulator are detected in $\mathcal{O}(N_\rho \cdot N_\varphi \cdot N_\theta)$. Since the size of the

point cloud $|P|$ is usually much larger than the number $N_\rho \cdot N_\varphi \cdot N_\theta$ of cells in the accumulator array, major improvements concerning computational expenses are made by reducing the number of points rather than by adjusting the discretization of the Hough Space. To this end Kiryati et al. propose a probabilistic method for selecting a subset from the original point set [116]. The adaption of the SHT to the Probabilistic Hough Transform (PHT) is outlined in Algorithm 6.

Algorithm 6 Probabilistic Hough Transform (PHT)

```
1: determine m and t
2: randomly select m points to create P^m ⊂ P
3: for all points p_i^m in point set P^m do
4:    for all cells (ρ, φ, θ) in accumulator A do
5:       if point p_i lies on the plane defined by (ρ, φ, θ) then
6:          increment cell A(ρ, φ, θ)
7:       end if
8:    end for
9: end for
10: Search for the most prominent cells in the accumulator, defining the detected planes in P
```

m points ($m < |P|$) are randomly selected from the point cloud P. These points are transformed into Hough Space and vote for the plane the points may lie on. The dominant part of the runtime is proportional to $m \cdot N_\varphi \cdot N_\theta$. By reducing m, the runtime is reduced drastically.

To obtain similar good results as with the Standard Hough Transform it is important that a feature is still detected with high probability even when only a subset of m points is used. The optimal choice of m and the threshold t depend on the actual problem at hand. Sensor noise leads to planes that appear thicker than they are in reality. Discretization of the Hough Space in combination with the sliding-window approach help to take care of this problem. The more planes are present in a point cloud the less prominent are peaks in the accumulator. The same effect appears, when objects are present in the point cloud that do not consist of planes. Depending on these factors the optimal number m varies between data sets with different characteristics.

Adaptive Probabilistic Hough Transform The size for the optimal subset of points to achieve good results with the PHT is highly problem dependent. This subset is usually chosen much larger than needed to minimize the risk of errors. Some methods have been developed to determine a reasonable number of selected points. The Adaptive Probabilistic Hough Transform (APHT) [211] monitors the accumulator. The structure of the accumulator changes dynamically during the voting phase. As soon as stable structures emerge and turn into significant peaks voting is terminated.

Only those cells need to be monitored after each voting process that have been touched. The maximal cell of those is identified and considered for plane extraction. If several cells have the same high score, one of them is chosen. A list of potential maximum cells is updated. A comparison of consecutive peak lists allows for checking the consistency of the peak rankings in the lists. Algorithm 7 outlines this procedure.

To speed up the process the list update is only performed after a small batch of points has voted. The list of peaks is limited in size and incrementally ordered by the value of the cells.

Algorithm 7 Adaptive Probabilistic Hough Transform (APHT)

while stability order of S_k is smaller than threshold t_k and maximum stability order is smaller than threshold $t_{\max}$ **do**
  randomly select a small subset $P^m \subset P$ of size n
  for all points $\boldsymbol{p}_i^m$ in P^m **do**
    vote for the cells in the accumulator
    choose maximum cell from the incremented cells and add it to active list of maxima
  end for
  merge active list of peaks with previous list of peaks
  determine stability order
end while

When updating the list with a peak the coordinates of the peak are taken into account. If two peaks lie in the spatial neighborhood of each other, the lower one is disregarded or removed from the list. In the early stages of the algorithm changes in the order of peaks are frequent. As updating proceeds, the structure of the accumulator becomes clearer. The stopping rule for the algorithm is determined by the stability of the most dominant peaks.

A set S_k of k peaks in the list is called stable, if the set contains all the same largest peaks before and after one update phase. The order within the set is insignificant for the stability. The number m_k of consecutive lists in which S_k is stable is called the stability order of S_k. The maximum stability count is the cardinality k of the set S_k with the highest stability order. In case there are two sets with the same stability order the one with the higher cardinality is preferred. The stability order of the set S_k that has the maximum stability count is referred to as maximum stability order.

The stopping rule for detecting exactly one plane is if the stability order of S_1 exceeds a predetermined threshold. For detecting k objects the stability order of S_k has to exceed a predetermined number. Detecting an arbitrary number of planes using several iterations with a fixed k may lead to long runtimes. Thus one recommends to let the program run until the maximum stability order reaches a threshold. The maximum stability count is then the number of found objects.

Progressive Probabilistic Hough Transform The Progressive Probabilistic Hough Transform (PPHT) [134] calculates stopping times for a random selection of points dependent on the number of votes in one accumulator cell and the total number of votes. The background of this approach is to filter those accumulation results that are due to random noise. The algorithm stops whenever a cell count exceeds a threshold of s points that could be caused by noise in the input. The threshold is calculated each time a point has voted. It is predicated on the percentage of votes for one cell from all points that have voted. Once a geometrical object is detected, the votes from all points supporting it are retracted.

The stopping rule is flexible as stopping at any time leads to useful results. Features are detected as soon as the contents of the accumulator allow a decision. If the algorithm is not stopped it runs until no points are left in the input set. This happens when all points have either voted or have been found to lie on an already detected plane. Even if the algorithm is not quit

early this does not mean that all points must have voted. Depending on the structure of the input many points may have been deleted before voting because they belong to a plane that was detected. The PPHT is outlined in Algorithm 8.

Algorithm 8 Progressive Probabilistic Hough Transform (PPHT)

while still enough points in P **do**
  select a point $\boldsymbol{p}_i$ from point set P
  for all cells (ρ, φ, θ) in accumulator A **do**
    if point $\boldsymbol{p}_i$ lies on the plane defined by (ρ, φ, θ) **then**
      accumulate cell $A(\rho, \varphi, \theta)$
    end if
  end for
  remove point $\boldsymbol{p}_i$ from P and add it to P_{voted}
  if highest accumulated cell is higher than threshold t **then**
    select all points from P and P_{voted} that are close to the plane defined by the highest peak and add them to P_{plane}
    search for the largest connected region P_{region} in P_{plane}
    remove from P all points that are in P_{region}
    for all points $\boldsymbol{p}_j$ that are in P_{voted} and P_{region} **do**
      Recall $\boldsymbol{p}_j$ from the accumulator
      Remove $\boldsymbol{p}_j$ from P_{voted}
    end for
    if the area covered by P_{region} is larger than a threshold **then**
      add P_{region} to the output list
    end if
  end if
end while

Randomized Hough Transform The Randomized Hough Transform (RHT) decreases the number of cells touched by exploiting the fact that a curve with n parameters is defined by n points [210]. For detecting planes, three points from the input space are mapped onto one point in the Hough Space. This point is the one corresponding to the plane spanned by the three points. In each step the procedure randomly picks three points $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ from the point cloud. The polar coordinates of the plane spanned by the three points are calculated as described in section 4.3.1 and the corresponding cell $A(\rho, \varphi, \theta)$ is accumulated. If the point cloud consists of a plane with ρ, φ, θ, after a certain number of iterations there will be a high score at $A(\rho, \varphi, \theta)$.

When a plane is represented by a large number of points, it is more likely that three points from this plane are randomly selected. Eventually the cells corresponding to actual planes receive more votes and are distinguishable from the other cells. If points are very far apart, they most likely do not belong to one plane. To take care of this and to diminish errors from sensor noise a distance criterion is introduced: $\max\_\text{dist}(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3) \leq \text{dist}_{\max}$, i.e., the maximum point-to-point distance between $\boldsymbol{p}_1, \boldsymbol{p}_2$ and $\boldsymbol{p}_3$ is below a fixed threshold; for minimum distance, analogous. The basic algorithm is structured as described in Algorithm 9.

Algorithm 9 Randomized Hough Transform (RHT)

```
1: while still enough points in point set P do
2:   Randomly pick three points p_1, p_2, p_3 from the set of points P
3:   if p_1, p_2 and p_3 fulfill the distance criterion then
4:     Calculate plane (ρ, φ, θ) spanned by p_1 ... p_3
5:     Increment A(ρ, φ, θ) in the accumulator space.
6:     if the cell |A(ρ, φ, θ)| equals threshold t then
7:       (ρ, φ, θ) parameterize the detected plane
8:       Delete all points close to (ρ, φ, θ) from P
9:       Reset the accumulator
10:    end if
11:  else
12:    continue
13:  end if
14: end while
```

The RHT has several main advantages. Not all points have to be processed, and for those points considered no complete Hough Transform is necessary. Instead, the intersection of three Hough Transform curves is marked in the accumulator to detect the curves one by one. Once there are three points whose plane leads to an accumulation value above a certain threshold t, all points lying on that plane are removed from the input and hereby the detection efficiency is increased.

Since the algorithm does not calculate the complete Hough Transform for all points, it is likely that not the entire Hough Space needs to be touched. There will be many planes on which no input point lies. This calls for space saving storage procedures that only store the cells actually touched by the Hough Transform.

Summary of Hough methods

All methods of the Hough Transform described in this section have in common that they transform a point cloud into the Hough Space and detect planes by counting the number of points that lie on one plane represented as one cell in an accumulator. Table 4.1 summarizes the features in which the methods differ.

In the SHT the complete HT is performed for all points. This makes it the only deterministic Hough variant. The PPHT, APHT and RHT have a stopping rule. While for the RHT the stopping rule is a simple threshold, e.g., number of points left in the point cloud or number of planes detected, the PPHT has a stopping rule that is based on the number of points already processed. Thus the algorithm is less sensitive to noise in the data. The most sophisticated stopping rule is applied in the APHT. Here the stability of several maxima is monitored over time during the voting phase making the algorithm more robust towards the negative effects of randomized point selection. In the PPHT and the RHT points are removed from the point cloud once a plane they lie on is detected. This does not only speed up the algorithm due to the decreasing number of points but also lowers the risk of detecting a false plane that goes through

Table 4.1: Distinctive characteristics of Hough Transform methods.

	SHT	PHT	PPHT	APHT	RHT
complete/deterministic	+	-	-	-	-
stopping rule	-	-	+	+	+
adaptive stopping rule	-	-	+	++	-
delete selected planes from point set	-	-	+	-	+
touch only one cell per iteration	-	-	-	-	+
easy to implement	+	+	+	-	+

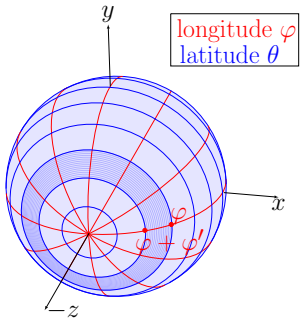
these points, especially in noisy data. The main advantage of the RHT is that in each iteration only one cell is touched. By avoiding to perform the complete Hough Transform the large amount of cells that correspond to planes only represented by very few points is most likely not to be touched by the RHT. The main disadvantage of the APHT is its implementation. While the other methods are straightforward to implement, the necessity to maintain a list of maxima that also takes into account neighborhood relations leads to a higher implementation complexity. A comparison of the performance of the Hough Transform methods follows after a discussion about the accumulator design.

The accumulator design

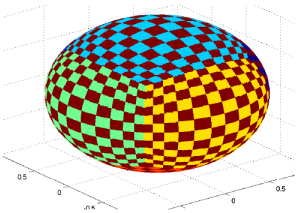
Without prior knowledge of the point cloud it is almost impossible to define proper accumulator arrays. An inappropriate accumulator, however, leads to detection failures of some specific planes and difficulties in finding local maxima, displays low accuracy, large storage space, and low speed. A trade-off has to be found between a coarse discretization that accurately detects planes and a small number of cells in the accumulator to decrease the time needed for the Hough Transform. Choosing a cell size that is too small might also lead to a harder detection of planes in noisy laser data.

Accumulator Array For the standard implementation of the three-dimensional Hough Transform [58] the Hough Space is divided into $N_\rho \times N_\varphi$ rectangular cells. The size of the cells is variable and is chosen problem dependent. Using the same subdivision for the three-dimensional Hough Space by dividing it into cuboid cells results in the patches seen in Fig. 4.19(a). The cells closer to the poles are smaller and comprise less normal vectors. This means voting in such an accumulator array favors the larger equatorial cells.

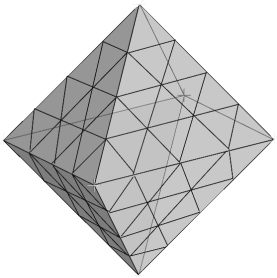
Censi and Carpin propose a design for an accumulator that is a trade-off between efficiency and ease of implementation [46]. Their intention is to define correspondences between cells in the accumulator and small patches on the unit sphere with the requirement that the difference of size between the patches on the unit sphere is negligible. Their solution is to project the unit



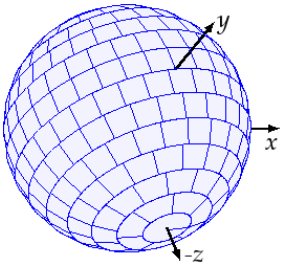
(a) Array



(b) Cube [46]



(c) Octahedron



(d) Ball

Fig. 4.19: (a) - (d): Mapping of the accumulator designs onto the unit sphere. (a) also illustrates the calculation of the length of a longitude circle at φ_i that determines the discretization for the ball design. The segment in question is the darker colored one.

sphere $\mathbb{S}^2$ onto the smallest cube that contains the sphere using the diffeomorphism

$$\varphi\colon \mathbb{S}^2 \to \text{cube}, \quad s \mapsto s/\left\|s\right\|_\infty .$$

Each face of the cube is divided into a regular grid. Fig. 4.19(b) shows the resulting patches on the sphere. A similar design focuses on the same decomposition in the direction of all coordinate axes [215]. This is achieved when partitioning the Hough Space by projecting the vertices of any regular polyhedron onto the unit space. The level of granularity can be varied by recursively subdividing each of the polyhedral faces. An example is given in Fig. 4.19(c). Each octahedron has the same subdivision. While both of these designs are invariant against rotation of 90° around any of the coordinate axes, they have one major drawback in common: when mapping the partitions on the unit sphere the patches appear to be unequally sized, favoring those planes represented by the larger cells.

Accumulator Ball The commonly used designs share one drawback, i.e., the irregularity between the patches on the unit sphere. Next, we present our novel design for the accumulator with the intention of having the same patch size for each cell. To this end, the resolution in terms of polar coordinates has to be varied dependent on the position on the sphere. Thus, the sphere is divided into slices (cf. Fig. 4.19(a)). The resolution of the longitude φ is kept as for the accumulator array. φ' determines the distance between the latitude circles on the sphere, e.g., the thickness of the slices. Depending on the longitude of each of the latitude circles the discretization has to be adapted. One way of discretization is to calculate the step width θ' based on the size of the latitude circle at φ_i. The largest possible circle is the equator located at $\varphi = 0$. For the unit sphere it has the length $\max_l = 2\pi$. The length of the latitude circle in the middle of the segment located above φ_i is given by $\text{length}_i = 2\pi(\varphi_i + \varphi')$. The step width in θ direction for each slice is now computed as

$$\theta'_{\varphi_i} = \frac{360\,^\circ \cdot \max_l}{\text{length}_i \cdot N_\theta}.$$

The resulting design is illustrated in Fig. 4.19(d). The image shows clearly that all accumulator cells are of similar size. Compared to the previously explained accumulator designs, the accumulator cube and the polyhedral accumulator, a possible drawback becomes obvious when looking at the projections on the unit sphere. The proposed design lacks invariance against rotations of multiples of $90\,^\circ$. This leads to a distribution of the votes to several cells. In practice however this problem is negligible since in most cases the planes to be detected do not align perfectly with the coordinate system, as shown in the experimental evaluation in [32]. There the accumulator designs, the array, the cube and the ball are compared on simulated as well as real 3D laser scans. The cuboid is chosen over the polyhedral design, since both designs seem to have similar characteristics and the cuboid design appears to be easier to manage. The results show that the assumptions about the flaws of the different designs were also visible in practice. The accumulator array has significant problems to detect planes close to the poles reliably. The accumulator cube shows good results with a slight disfavor for planes close to the edges and the corners. The accumulator ball performs the best for arbitrary planes. If the cap of the ball is divided into several patches, more cells intersect at the poles than at the other parts of the accumulator. This leads to a distribution of votes over all these patches, decreasing the count for each of these cells. This problem is solved by creating a circular cell around the pole that has the desired size of the patches and proceed with the rest of the sphere in the same manner as before. Using this design the ball and the cube show similar performance. The ball performs better in some test cases, the cube in others. On average the ball design slightly outperforms the cube. The experiments also show that once a plane is detected correctly the parameters are calculated equally well with each accumulator design.

Experimental evaluation

Evaluation of Hough methods For 2D data the Standard Hough Transform is one of the standard methods for detecting parameterized objects. But even there, enormous computational requirements have lead to the emerge of more and more methods to accelerate the Hough Transform without loss of precision. For 3D data the requirements increase drastically. Therefore,

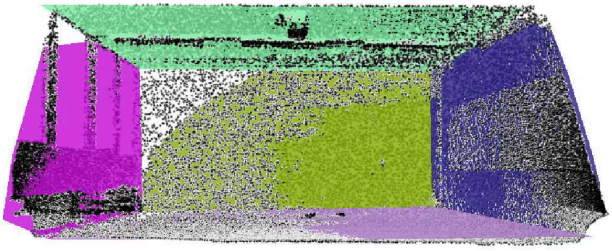


Fig. 4.20: Scan of an empty office from the AVZ data set with planes detected (RHT).

we are dealing with the question of applicability of the Hough Transform in the 3D case. In this section the performance of the different Hough methods is evaluated. The experiments are performed on an Intel Q9450 2.66 GHz processor with 4 GB RAM using one thread.

To show the applicability of the Hough Transform to real laser data, we apply the RHT to a laser scan of an empty office. The result is shown in Fig. 4.20. All planes were correctly detected within less than 600 ms on an Intel Core 2 Duo 2.0 GHz processor with 4 GB RAM. For further evaluation on the applicability of the Hough Transform see [26].

To evaluate the different Hough methods we use the same setup for each method. For easier evaluation we reduce the experimental setup to a simple test case. A cube with a side length of 400 cm is placed around the origin. Each side consists of 10,000 points which are randomly distributed over the entire face of the cube with a maximal noise of 10 cm. Different rotations are applied to the cube to simulate different orientations of planes. The advantage of using this simple model is the existence of ground truth data for the actual planes. The cube possesses a perpendicular structure which is characteristic for most man-made indoor environments. The first experiment investigates how well the Hough methods detect one plane. We express this in terms of mean and variance of the result. Using the cuboid accumulator design and only one face of the laser scan cube model we consider only one face of the accumulator. The accumulator face is divided into 22×22 faces. The slice considered is the one with $198 < \rho < 204$. The mean is calculated as the sum of all accumulator cells weighed by the score of that cell and divided by the number of cells. The error plotted is the distance between the calculated mean and the supporting point of the optimal plane representation of the face. The variance illustrates the distribution of the values in the accumulator face in relation to the calculated mean.

For the evaluation it is necessary to divide the accumulation phase and the maximum search phase. Therefore we cannot evaluate all Hough methods according to this scheme. For the PHT we vary the number of points used. The RHT is slightly adapted. Instead of running until a threshold is hit, the number of triples picked is determined as a percentage of the total number of points on the cube face. The average of running 20 times with each setting is plotted in Fig. 4.21. With more than 10% the RHT outperforms the SHT. Starting from 30% the error of the PHT is close to that of the SHT. The variance of the SHT and the PHT are close together. The RHT has a significantly smaller variance. Interpreting the results suggests that applying the PHT with at least 30% of the points leads to results that sufficiently represent the input

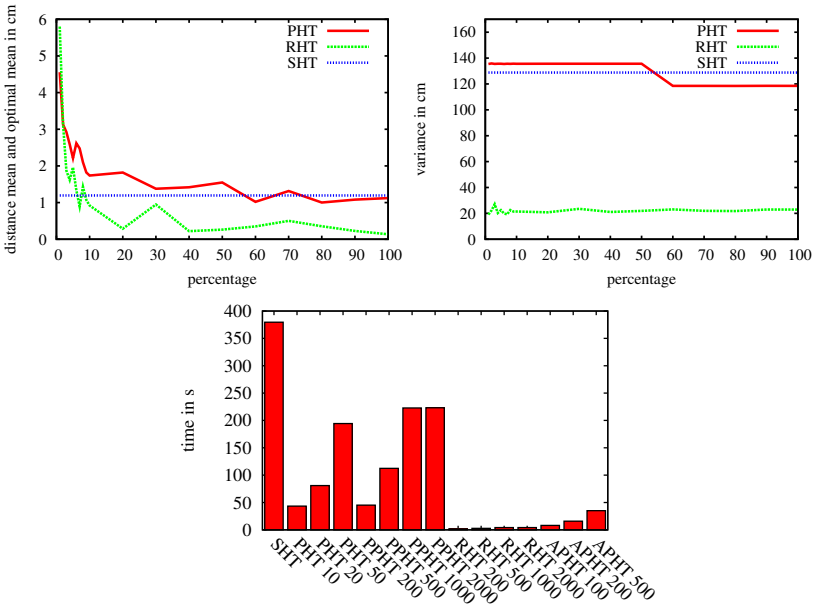


Fig. 4.21: Above: Comparison of the different point selection strategies. For the SHT all points are used and transformed into Hough Space. For the PHT only $x\,\%$ of the points are chosen and the HT applied to them. For the RHT $x\,\%$ times three points are chosen and the cell that corresponds to the plane spanned by these three points is accumulated. Plotted are the mean and the variance of one face of the resulting accumulator space. Below: Comparison of the runtime for the different Hough methods.

data. The results for the RHT have to be considered very carefully as the RHT benefits from the simplicity of the input data. If using only one plane as input the RHT has hardly any variance. Selecting three points from different planes leads to more erroneous results. Due to that a more differentiated test of the RHT is needed.

Using 9 different rotations of the modeled cube the different Hough methods are further compared. The accumulator used for this evaluation is the ball structure with $N_\varphi = 45$, $N_\theta = 90$ and $N_\rho = 100$. The average of 20 test runs for all Hough methods is plotted in Fig. 4.22. Shown is the number of mis-detected planes until each face of the cube is found exactly once. The quality of the detected planes is calculated as the angle between the perfect normal vector and the first normal vector of each detected face. The differences of the angle error are within a negligible range. The number of incorrectly detected planes differs depending on the rotation of the point cloud. In all cases the SHT performs noticeably better than the adapted version of the RHT (above). However, in its original version the RHT (cf. Section 4.3.4) has a phase where

the accumulator is reset and the points that belong to an already detected plane are deleted.

The results for the original RHT are shown in Fig. 4.22 (center) along with results on the same experiment for the PPHT. The thresholds for the maximal accumulator counts for RHT and PPHT are set to 1000. The results show that when deleting the detected planes in between the accuracy of the HT with respect to finding each plane exactly once is significantly improved. In most cases the first six detected planes correspond to six different faces of the cube. Noticeably, for the PPHT also the error in the orientation of the normal vector is significantly smaller. In practice the difference does not play an important role if plane fitting is used in order to diminish the negative effects of the discretization.

Like the RHT and PPHT the APHT includes a stopping rule for when to stop transforming points into Hough Space. Instead of detecting a single peak the stability of many peaks is analyzed. For detecting an unknown number of planes a maximum stability count at which the algorithm stops is set. Fig. 4.22 (below) shows a brief analysis of the APHT. The graph on the right shows an exemplary distribution of accumulator counts when using a list of 20 maxima and different maximum stability counts as stopping rule. It becomes clear that the counts between the detected planes and the not detected planes become very distinctive. However, the algorithm detects 8 planes for the cube data set. The left image shows a more thorough evaluation. The lines indicate the number of planes detected by the algorithm averaged over 20 runs. The boxes show the number of planes of this list that are a best representation for a cube faces. It is clear that the algorithm gives a close prediction of the actual planes but fails to correctly identify all planes in almost any case.

The remaining question is how far the different methods improve the run time of the algorithm. For this we run each algorithm 20 times, measure the time needed and plot the average time. For the variations we use different settings. The PHT is run with 10, 20 and 50 percent of the points. For the RHT and the PPHT maximal counts of 200, 500, 1000 and 2000 are used. For the APHT we used a maximum stability count of 100, 200 and 500. Since the RHT and PPHT have intermediate phases for deleting the points for one detected plane we include this phase into the measurements for all algorithms. This means the runtime includes the voting phase in the accumulator, the maximum search, plane fitting and creation of the convex hull of the planes.

Fig. 4.21 shows the runtimes for the different Hough algorithms. All variations drastically improve the runtime of the HT compared to the SHT. As expected, the runtime for the PHT decreases proportionally to the decrease of points. The PPHT yields the best quality results. The runtime is similar to the PHT. However, the experiments that lead to the results of the quality evaluation were done with a maximal accumulator count of 100 which leads to even shorter runtimes, as depicted in the graph. Considering that the PHT needs to be run with at least 30% of the points, the PPHT is considerably faster. The APHT needs astonishingly short runtimes. Nevertheless, in the configurations used in the experiments the APHT was not able to reliably detect all six planes. The RHT outperforms all the other HT methods by far concerning runtime. Considering that the quality of the detected planes was only slightly behind the PPHT the RHT seems to be the method of choice for detecting an unknown number of planes in a laser scan map in reasonable time.

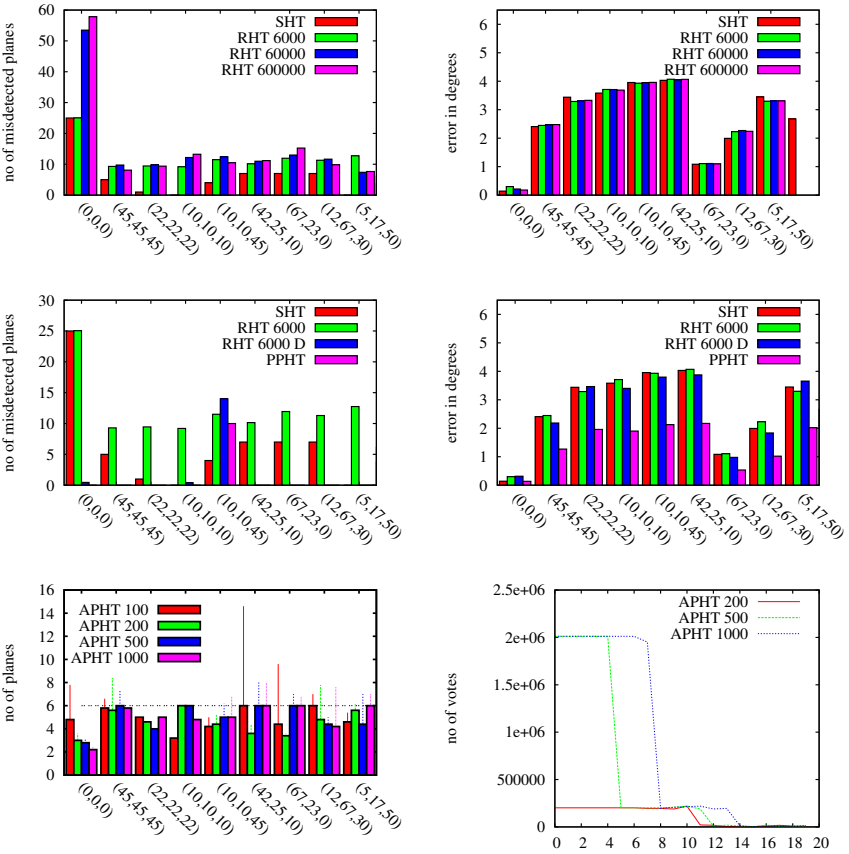


Fig. 4.22: The experiments were carried out on the modeled cube scan rotated by $(\alpha_x, \alpha_y, \alpha_z)$ without plane fitting after the HT. Above: Comparison of the RHT with the SHT when applying it on the test cube. For RHTx, x triples of points are selected. Center: Comparison of different Hough methods with respect to their ability to correctly detect all planes in a data set and their accuracy. Below: Results for running the APHT with different maximum stability counts on the cube data set.

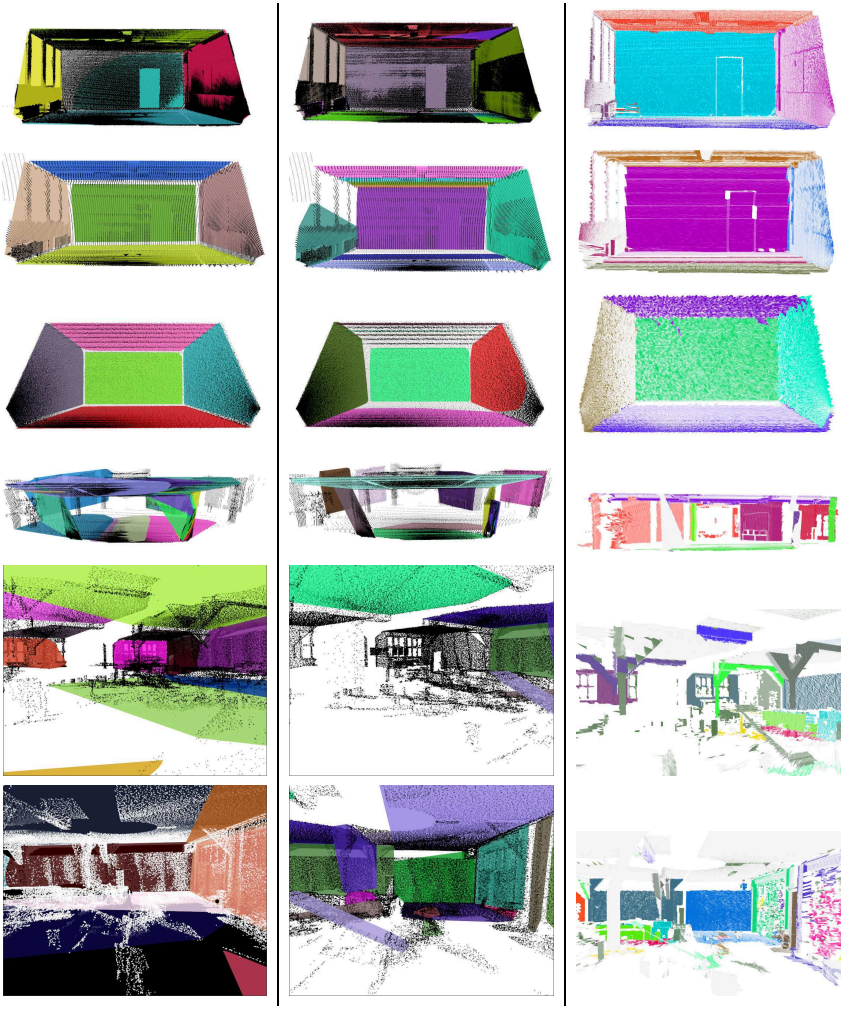


Fig. 4.23: Left: Results with the RHT. Middle: Region growing. Right: Hierarchical primitive fitting. From top to bottom: 1. AVZ data set empty room (high resolution) 2. AVZ data set empty room (low resolution) 3. simulated scan, 4. hall, 5.+6. Arena data set

Comparison with Region Growing We have analyzed the Hough Transform and other effective methods for plane detection. Since a detailed comparison with respect to RANSAC is given in [194] we focus here on region growing methods described in section 4.3.2, the approach by Poppinga et al. that works on range images (RG) [161] and the Hierachical Fitting of Primitives (HFP) presented by Attene et al. [9]. To achieve comparability we only used the plane fitting part of the prototype of the latter and neglected the sphere and cylinder fitting functionalities. As the algorithm works on triangle meshes the data was converted beforehand.

We test both region growing procedures on different laser range images. First, we use two laser scans of different resolution taken in an empty office room from the AVZ data set. The left wall consists mainly of windows and heating installations. On the right wall there is a whiteboard. Challenging are the wave-like systematic errors on the floor and the ceiling, that are especially present in the image with lower resolution. Second, we simulate a laser scan with the same wave-like structures. Third, a laser scan from a hall where eight hallways meet from the AVZ data set. Last, a 360 ° laser scan from the ARENA data set.

We apply two variants of the RHT. After a voting phase all points are selected that lie on the highest ranking plane. In the first variant a plane is fitted through these points and the convex hull of all these points is built. In the second variant an occupancy grid of the fitted plane is calculated and a two-pass clustering algorithm (see [55]) is applied to the selected points. Only the largest cluster is removed from the point set. If the variance of the fitted plane is above a threshold, the plane is neglected. The accumulator is discretized with $\rho' = 20\,\text{cm}$, $\varphi' = 176$ and $\theta' = 88$. The threshold t for the maximum accumulator value is set to 50. The maximum value for ρ, the threshold for the stopping rule, the maximum point to plane distance and the distance for clustering are chosen dependent on the data. For scans with a lower resolution a larger distance is allowed for neighboring points. For scans with a lower accuracy, a higher point to plane distance is allowed. The same values are used for RG. The chosen polygonization is also the convex hull. The distance for clustering is used in the triangulation to ignore triangles with too large edges.

Table 4.2 compares the runtimes of the four approaches. Three times measurements are listed for each data set and method. First, the runtime for the voting phase of the RHT and region growing respectively, second, the time for clustering and calculating the convex hull, and third, the sum of both parts. Additionally, the number of planes found by the algorithms is listed. For the mesh segmentation only the entire runtime is given. Comparison of the runtimes shows the competitiveness of the RHT. The mesh segmentation performs the worst but the runtimes need to be considered with care as a complete hierarchy of planes is generated. For data sets that consist mainly of planes (rows 1 to 3) the RHT outperforms the RG. The runtime of the RHT grows mainly with the number of planes and the number of non-planar points in the data, while the runtime of RG grows with the number of points. This is shown through the runtimes in the fourth and fifth row. However, the runtimes remain in a considerable range.

The scans with the found planes are shown in Fig. 4.23. All algorithms succeed in finding planar structures. From the experiments it appears that the RHT performs better in finding large planes while RG and HFP find planes accurately but tend to detect several smaller planes rather than one large plane where errors are present in the data. This becomes obvious when looking at the ceiling of the empty room or the simulated scan. RHT nicely fits one plane through the

Table 4.2: Runtimes in s for Randomized Hough Transform (RHT), Region Growing (RG) [161] and Hierarchical Primitive Fitting (HFP) [9] on different laser scans. Listed are the times for detecting planes, polygonization (for RHT and RG) and the total time (for all). N is the number of detected planes.

	# points	RHT	N	RHT C	N	RG	N	HFP
empty room	325,171	0.072	5	0.088	5	5.31	$\gg 5$	
		+0.599		+0.730		+2.33		
		= 0.671		= 0.818		= 7.64		78.4
empty room	81,631	0.096	5	0.092	5	1.22	9	
		+0.195		+0.200		+0.5		
		= 0.291		= 0.292		= 1.72		14.2
simulated	81,360	0.049	5	0.055	5	1.33	8	
		+0.182		+0.199		+0.49		
		= 0.231		= 0.254		= 1.82		18.7
hall	81,360	2.813	16	1.818	17	1.35	13	
		+0.234		+0.277		+0.4		
		= 3.047		= 2.095		= 1.75		36.0
arena	144,922	13.960	18	6.930	18	2.13	11	
		+0.477		+0.662		+0.57		
		= 13.960		= 7.592		= 2.70		16.0

ceiling while RG finds several smaller slices. The same is true for the arena. While the outlines of the building are nicely found by the RHT, RG finds smaller planes within the building that are completely ignored by the RHT. For the HFP the N best planes are depicted, where N is the larger number of planes detected by the other two algorithms. The HFP performs similar to the RG approach as it represents smaller regions. In the cluttered environments, the hall and the arena, the detected planar patches are very accurate. However, main structures such as the floor and the ceiling are missing.

Discussion

This section evaluates the Hough Transform with respect to the task of detecting planar structures in laser range images and presents the accumulator ball as an accumulator design. The advantage of this design is that it does not unjustifiably favor planes with specific parameters, due to the equal size of the patches. The evaluation of the different Hough methods shows clearly that the Randomized Hough Transform is the method of choice when dealing with three-dimensional data due to its exceptional performance as far as runtime is concerned. The randomized selec-

tion of points does not diminish but rather improve the quality of the result. Because points are removed from the data set once they have been found to lie on a plane, accuracy for the next plane increases. Comparison with the region growing [161] and the hierarchical fitting of primitives [9] shows that the RHT also competes with other plane extraction methods. Furthermore the results show that it is a good choice when trying to detect the underlying structures of the environment, as its main advantage lies in the detection of large structures.

4.4 Summary

To handle the large amount of data acquired with 3D sensing equipment in short periods of time, methods need to be found to reduce the computational burden when processing the data. Converting a point cloud into a panorama image takes advantage of the sweeping manner in which the data is acquired. This enables the adaption of methods from image processing. The ability to calculate the pixel coordinates directly leads to a fast construction. A panorama image has little overhead. The main drawback is the restriction to a single scan position. Even though it is technically possible to construct a panorama image from a composition of point clouds, the main advantage is lost even when storing all points that fall into each pixel. Either occluded points remain occluded or the neighborhood relationship between neighboring pixels does not hold true anymore for the points contained within.

Alternative representations exploit the 3D geometry of the data and generate a tree structure. In an octree the space is divided into equal sized cubes while a kD-tree splits the space along one split plane in each level. Both data structures allow to descend quickly into the tree thus not considering a large portion of the data. This increases the speed for nearest neighbor detection and identification of points belonging to a certain geometric structure, such as planes or cylinders.

The identification of planes is especially useful given that most man-made structures are composed of planar parts. Different approaches are presented in this chapter. RANSAC benefits from an octree data structure as each iteration requires the evaluation of all points. Region growing approaches require the knowledge about connectivity, making the panorama image an ideal data structure. This is of interest for the detection of the calibration pattern in Chapter 7. The Hough Transform and its variants are evaluated in detail. The RHT is shown to perform well when trying to detect the most prominent planes in a point cloud making it an ideal candidate for detecting the structural elements, such as walls, ceilings and floors for the reconstruction of interiors in Section 8.1. The reconstruction of interiors is challenging as laser scanning data suffers from occlusions. To acquire complete models of an environment it is therefore necessary to collect scans at various locations and to join the data into one common coordinate system. This is the focus of the next chapter.

Chapter 5

Registration of 3D point clouds

In the previous chapter data structures to handle data efficiently were discussed. These data structures focused on the representation of data from one acquisition position. This section describes how to join data from several positions, a process often referred to as scan registration or scan matching. It focuses on how the information from the 3D laser scanner is used to create a map of the environment that the robot traverses. 3D environment mapping using 3D scanners on mobile robots is subject to research [148, 189]. Given a map, the sensor data allows for localizing the robot in it. When a robot explores an unknown environment it has no access to such a map. However, given the perfect localization of the robot in a reference coordinate system the perceived sensor data can immediately be transformed into this reference frame. Unfortunately, robot localization is prone to errors. Accordingly, a robot in an unknown environment faces an optimization problem, the problem of simultaneously having to create a map and to localize itself in this map given erroneous input information. This problem is known as the SLAM (Simultaneous Localization and Mapping) problem. It is closely related to bundle adjustment known from Photogrammetry.

In a typical SLAM scenario the robot starts from a point in the environment that defines its coordinate system. Commonly for mobile robots the map is reduced to the 2D ground plane. To capture the entire environment, the robot has to acquire data at different positions and join this data to create one map. Maps created from laser scanner data typically belong to one of three categories: raw point data, lines or grid maps. All three categories have their advantages and disadvantages. When storing raw point data, none of the initially captured information is lost. Comparison of data from different time steps allows for identification of dynamic changes in the environment. The highest possible precision based on the specification of the scanning device is possible. However, handling these huge amounts of data is challenging. Methods need to be developed to access the data in a timely manner. For a simple obstacle check along a path all points need to be considered. To avoid this, data structures, as presented in the previous section, have to be used to enable fast access to the data. To facilitate navigation within the map vectorization is applied. Most man-made structures consist of straight line segments. Vectorizing all points on a line into a line segment allows for fast algorithms. Point operations on all line points are reduced to just one line operation. As a drawback finer structures are likely to be neglected in this representation and noisy data is prone to declare narrow passages impassable

for the robot. To account for the sensor noise in the data probabilistic occupancy grids are the method of choice. The environment is divided into small cells, each of which is assigned an occupancy value. Probability values for the cells are updated after each measurement. If the cell is traversed by the laser beam, it is considered free. If the laser beam is reflected in the cell, it is considered occupied. Based on probability each cell is classified into free, unseen or occupied. In the following, algorithms from the point and grid map category are explained in more detail while line maps are utilized in Chapter 6.

5.1 The ICP algorithm

The ICP algorithm (iterative closest point) is one of the most common algorithms for registering two point clouds. First proposed in 1992 [17], it has hence developed into a state of the art for optimizing the alignment of two point clouds. Its major drawback is the necessity to have pose estimates available. In robotics, this requirement is fulfilled if the robot is equipped with sensors to measure odometry. For other data sets, GPS measurements or feature-based approaches (see Section 5.3) provide these.

Starting from such initial pose estimates the algorithm calculates effectively the correct transformation between two point clouds by means of minimizing distances between corresponding points. Corresponding points are chosen based on Euclidean distances. The algorithm is described in Algorithm 10.

Algorithm 10 The ICP algorithm

Require: point clouds M and D

1: find point correspondences

2: minimize for rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$

$$E_{\text{ICP}}(\boldsymbol{R},\boldsymbol{t}) = \frac{1}{N}\sum_{i-1}^{N} ||\boldsymbol{m}_i - (\boldsymbol{R}\boldsymbol{d}_i + \boldsymbol{t})||^2 \tag{5.1}$$

3: **return** pose $(\boldsymbol{R},\boldsymbol{t})$

4: iterate 1 and 2

Given an already registered point cloud M and a point cloud D with an initial pose estimate the ICP first tries to find for each point $\boldsymbol{d}_i$ from D the point $\boldsymbol{m}_i$ in M that is closest to $\boldsymbol{d}_i$. N is the number of point correspondences. Then one needs to solve for a transformation $(\boldsymbol{R},\boldsymbol{t})$ (translation $\boldsymbol{R}$ and orientation $\boldsymbol{t}$) that minimizes the error function E_{ICP}, i.e., the distance between all point pairs. These two steps are iterated to find the best transformation between the two point clouds. For best results a threshold t_{dist} is introduced and all point pairs with a distance larger than t_{dist} are discarded from the calculations. Without the threshold t_{dist} the ICP is proven to converge monotonously towards a local minimum E_{min} as in each iteration the error is reduced [17]. In the next iteration the point correspondences are either the same as in the previous iteration or they have a smaller distance. When introducing the threshold the number of point correspondences N is not constant anymore. Point correspondences may be removed

when the distance between the points increases or more point correspondences may be found when points are moved within the range of the threshold thus increasing the error value [146]. In practice this procedure improves the matching results especially in areas with large occlusions as wrong correspondences from occluded areas have less influence on the registration.

For multiple scans the pairwise scan matching is adopted as follows: using the first scanning position as reference the nth scan is always registered against the $(n-1)$th scan. This way all scans are sequentially transformed into the same coordinate system.

To minimize the error function different methods have been presented in literature. These method can briefly be divided into two categories, namely direct and indirect methods. Indirect methods as Gradient Descent, the Gauss-Newton method or the Levenberg-Marquardt algorithm evaluate the error function (5.1) in various positions thus needing more time. Nüchter et al. describe several closed form solutions to the ICP problem [151]. In this thesis the method using singular value decomposition (SVD) will be used. The difficulty of minimizing the error function lies in the preservation of the orthonormality constraint of the rotation matrix $\boldsymbol{R}$. To enforce orthonormality the minimization is split into two parts, calculating the rotation first and deriving the translation afterwards from the results. To compute the rotation, the point sets are shifted such that the centroid lies on the origin of the coordinate system:

$$M' = \left\{\boldsymbol{m}_i' = \boldsymbol{m}_i - \boldsymbol{c}_m\right\}_{1,\ldots,N}, \quad D' = \left\{\boldsymbol{d}_i' = \boldsymbol{d}_i - \boldsymbol{c}_d\right\}_{1,\ldots,N} \tag{5.2}$$

by subtracting the mean

$$\boldsymbol{c}_m = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{m}_i, \quad \boldsymbol{c}_d = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{d}_i. \tag{5.3}$$

The error function is adapted introducing the points $\boldsymbol{m}_i'$ and $\boldsymbol{d}_i'$ from the translated point clouds M' and D':

$$E_{\mathrm{ICP}}(\boldsymbol{R},\boldsymbol{t}) = \frac{1}{N}\sum_{i=1}^{N}||\boldsymbol{m}_i' - \boldsymbol{R}\boldsymbol{d}_i' - \underbrace{(\boldsymbol{t} - \boldsymbol{c}_m + \boldsymbol{R}\boldsymbol{c}_d)}_{=\tilde{\boldsymbol{t}}}||^2 \tag{5.4}$$

$$= \frac{1}{N}\sum_{i=1}^{N}||\boldsymbol{m}_i' - \boldsymbol{R}\boldsymbol{d}_i'||^2 - \frac{2}{N}\tilde{\boldsymbol{t}}\sum_{i=1}^{N}\left(\boldsymbol{m}_i' - \boldsymbol{R}\boldsymbol{d}_i'\right) + \frac{1}{N}\sum_{i=1}^{N}||\tilde{\boldsymbol{t}}||^2. \tag{5.5}$$

Minimizing Equation (5.5) requires minimization of each of the three individual terms. The second sum computes to zero, as it is essentially the difference of the centroids of the new point clouds M' and D' which are designed to be zero. The third part has a minimum at $\tilde{\boldsymbol{t}} = 0$ which dissolves to

$$\boldsymbol{t} = \boldsymbol{c}_m - \boldsymbol{R}\boldsymbol{c}_d. \tag{5.6}$$

Accordingly the minimization reduces to calculating the rotation to minimize the first term from equation (5.5):

$$E_{\mathrm{ICP}}(\boldsymbol{R},\boldsymbol{t}) \propto \sum_{i=1}^{N}||\boldsymbol{m}_i' - \boldsymbol{R}\boldsymbol{d}_i'||^2 \tag{5.7}$$

and computing the translation according to (5.6).

Calculating the optimal rotation using SVD was introduced by Arun, Huang and Blostein [6]. Using matrix representation, the rotation $\boldsymbol{R}$ has to be an orthonormal 3×3 matrix. To find the optimal $\boldsymbol{R}$ the cross-correlation Matrix $\boldsymbol{H}$ is constructed from the point pairs as

$$\boldsymbol{H} = \sum_{i=1}^{N} \boldsymbol{m}_i' \boldsymbol{d}_i'^T = \begin{pmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{pmatrix} \tag{5.8}$$

where

$$S_{xx} = \sum_{i=1}^{N} m'_{x,i} d'_{x,i},\ S_{xy} = \sum_{i=1}^{N} m'_{x,i} d'_{y,i}, \ldots . \tag{5.9}$$

Deriving the matrices $\boldsymbol{V}$ and $\boldsymbol{U}$ by singular value decomposition $\boldsymbol{H} = \boldsymbol{U}\Lambda\boldsymbol{V}^T$ the rotation is calculated as $\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^T$.

The main drawback of the ICP algorithm is its necessity to have initial pose estimates available. For small pose changes the estimates are negligible, however, given the long range and the scan acquisition times of 3D laser scanners the assumption of small changes usually holds not true. Thus, additional methods are used to estimate the pose of the next scan. Global navigation satellite systems (GNSS) are restricted to outdoor scenarios and are limited in accuracy. To overcome these limitations the environment can be prepared with a local localization system. An active system, such as the iSpace system described in Chapter 2.6, defines its own coordinate system or passive markers that are placed in the environment are used as reference points. With both variants either a local coordinate system is defined or these systems are located within the global coordinate system. The two main disadvantages are that the environment needs to be prepared before data acquisition and that the localization system, e.g., the transmitters or the markers, alter the scene and are visible in the data.

The alternative is to use natural features, this is described in Section 5.3. In the robotics context the method of choice is to let the robot track the path while it moves from one scanning position to the next. Odometry is often applied but not very accurate. Laser scanners are among the most accurate measurement devices on robots. Thus an often applied method is to use fast 2D laser scanners to track the robot position over time and create a map simultaneously. In the following a 2D mapping methods using 2D laser scanners is described.

5.2 2D mapping

Laser-based 2D mapping methods can be categorized based on various aspects. Very few methods store the raw data. Instead, most commonly the environment is discretized in a 2D grid. A further distinction is made based on the dependency on odometry information. The method used within this thesis is GMapping [93] which uses odometry information. The alternative Hector SLAM [117] uses odometry or data from an inertial measurement unit optionally while OHM-TSD-SLAM [135] works completely without odometry. All these methods use occupancy grid maps for map representation. Thus, the next section gives a brief introduction on this representation before describing the GMapping algorithm in more detail.

5.2.1 Occupancy grid maps

Occupancy grid maps are a commonly used means of representing a 2D spatial map. In the digital age it comes naturally to discretize space using a regular grid. In robotics this method was originally proposed in the mid-eighties by Moravec and Elfes for noisy wide angle sonar data [139]. Nowadays it is the de-facto state of the art in robotic mapping using 2D laser scanners. The idea is rather simple, the space covered by the map is divided into a regular grid of arbitrary size and resolution. Using thresholding, averaging and clustering of noisy measurements the combination of several readings is expected to result in a final map that gives a good representation of the environment. For laser scanners the sensor readings are not as noisy as for other range sensors, however, since the position of the robot is also bound to errors, the mapping process contains a certain amount of uncertainty.

Each range measurement gives two types of information as depicted in Fig. 5.1. On the one hand, the final reading of the measurement gives the position of an obstacle, i.e., the corresponding cell is most likely at least partially occupied. On the other hand, the space between the sensor and the obstacle is probably empty. Considering the divergence of the beam, the occupancy information is not only limited to a line traversing the space but represented by a cone. This leads to a map representation with three different labels, `occupied` (black), `empty` (white) and `unknown` (gray). In addition to this label each cell is assigned two probability values, $\mathrm{Occ}(x, y)$ for the occupancy and $\mathrm{Emp}(x, y)$ for the emptiness of the cell. Both values are initialized with zero.

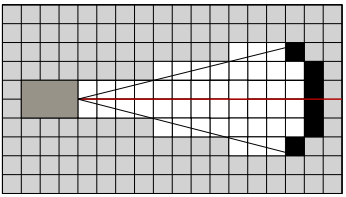


Fig. 5.1: Sketch of a range measurement. Black cells are `occupied`, white cells are `empty` and gray cells are `unknown`.

To start the mapping process, the entire map is set to `unknown`, meaning that insufficient information about its occupancy is available. After each sensor reading k the map is updated. One probability density function determines the probability $\mathrm{Emp}_k(x, y)$ based on the sensor reading and the distance of the cell to the main axis of the measurement. Likewise, another probability density function is applied to determine $\mathrm{Occ}_k(x, y)$ for all cells in the area of the actual range measurement. The values of each cell grazed by the range measurement k are updated as described in Algorithm 11.

Algorithm 11 Map update

Superposition of `empty` areas

1: $\mathrm{Emp}(x, y) := \mathrm{Emp}(x, y) + \mathrm{Emp}_k(x, y) - \mathrm{Emp}(x, y) \cdot \mathrm{Emp}_k(x, y)$ # Enhance

Superposition of `occupied` areas

2: $\mathrm{Occ}_k(x, y) := \mathrm{Occ}_k(x, y) \cdot (1 - \mathrm{Emp}(x, y))$ # Cancel

3: $\mathrm{Occ}_k(x, y) := \mathrm{Occ}_k(x, y) / \sum \mathrm{Occ}_k(x, y))$ # Normalize

4: $\mathrm{Occ}(x, y) := \mathrm{Occ}(x, y) + \mathrm{Occ}_k(x, y) - \mathrm{Occ}(x, y) \cdot \mathrm{Occ}_k(x, y)$ # Enhance

Thresholding

5: $$\mathrm{Map}(x, y) := \begin{cases} \mathrm{Occ}(x, y) & \text{if} \quad \mathrm{Occ}(x, y) \geq \mathrm{Emp}(x, y) \\ \mathrm{Emp}(x, y) & \text{else} \end{cases}$$

Identical labelings from several sensor readings enhance each other. If several range readings assert that a cell is `empty`, the confidence of it being empty is increased. The same holds true for the label `occupied`. On the contrary, evidence that the cell is `empty` weakens the certainty of it being `occupied`. Values for both $\mathrm{Emp}(x, y)$ and $\mathrm{Occ}(x, y)$ range between 0 and 1. Due to the physical properties of a sonar beam, the empty and occupied probabilities are not treated identically in the map update. The empty area consists of an entire volume that is probably empty, while the occupied probability is influenced by the lack of knowledge of the exact reflecting point. Due to the beam divergence it is only known that there is a point somewhere in the range of the distribution that reflects the signal. This effect is less pronounced for laser range measurements. For 2D laser scan occupancy grid maps it has become common practice to reduce the information to one probability value $p(x, y)$ where $p = 1$ refers to `occupied` and $p = 0$ to `empty`, respectively [195]. Due to the fact that measurements are integrated over time, sensor noise and dynamic obstacles are inherently erased from the map after some time and thus not represented in the final map.

Fig. 5.2 shows an example of the map update in an occupancy grid map. The robot moves from one static position to another and the map is expanded on the way. The laser scanner on the robot is slightly tilted downwards causing wrong range measurements when it sees the floor. As the robot moves, the false line of obstacles moves with the robot. While the robot keeps moving and sees those wrongly marked `occupied` cells repeatedly empty, the label is updated and the lines fade out.

5.2.2 GMapping

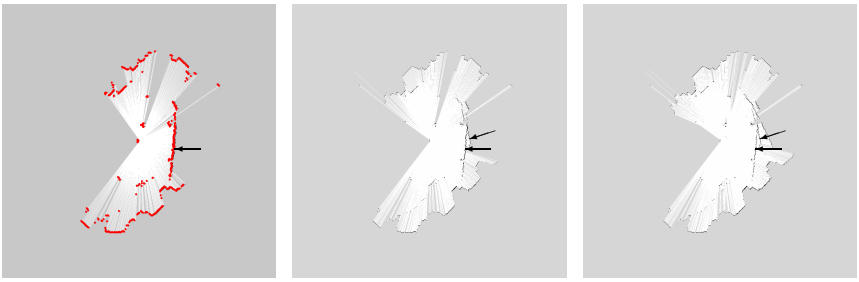
Grid-maps are ideally suited for probabilistic methods. Sensor measurements acquired with a mobile robot are prone to inaccuracies. Thus, all conclusions drawn from these measurements always underlie uncertainties. The occupancy grid method itself tries to overcome these uncertainties by using multiple measurements. So far the robot pose and thus the sensor pose were assumed to be known. However, this is usually only true to a certain degree. Odometry suffers from inaccuracies that are even more pronounced on changing ground conditions. Thus, it is common to use probabilistic methods also to represent the robot pose [195]. GMapping [92] is a probabilistic SLAM approach that owes its prominence to its early implementation as a ROS [190] module. GMapping uses Rao-Blackwellized particle filters where each particle holds its own map of the environment. The movement of the robot and the last observation are used to predict the next possible state thus maintaining a map of the already explored environment [93].

Using probabilistic methods, the SLAM problem is seen as the problem of estimating the joint conditional probability about the map m and the trajectory $x_{1:t} = x_1, \ldots, x_t$ of the mobile robot given the sensor observations $z_{1:t} = z_1, \ldots, z_t$ and the odometry estimates $u_{1:t} = u_1, \ldots, u_{t-1}$ acquired by the mobile robot:

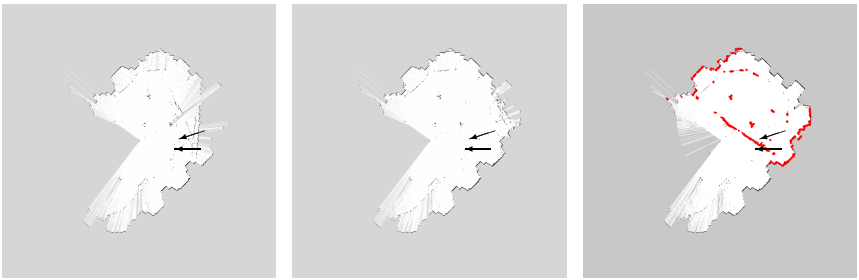
$$p(x_{1:t}, m | z_{1:t}, u_{1:t-1}). \tag{5.10}$$

Using so-called Rao-Blackwellization as proposed by Murphy [142], the estimation of the robot trajectory is separated from and performed before the computation of the map given this trajectory by factorization:

$$p(x_{1:t}, m | z_{1:t}, u_{1:t-1}) = p(m | x_{1:t}, z_{1:t}) \cdot p(x_{1:t} | z_{1:t}, u_{1:t-1}). \tag{5.11}$$



(a) 0 s (b) 5 s (c) 10 s



(d) 15 s (e) 20 s (f) 30 s

Fig. 5.2: Development of an occupancy grid map from the IMPERIAL HALL data set. The robot control loop is started and the robot stays at its starting position until $t = 0$ s. Then it moves until it reaches its next position at $t = 25$ s and stops moving again. Gray shows cells marked as `unknown`, white cells are `empty` and black cells `occupied`. In (a) and (f) the laser scan is depicted in red. The scanner is mounted slightly tilted downwards on the robot. Thus it sees the floor in front of it, as marked with the arrows. During the update, the wrong lines disappear.

As $x_{1:t}$ and $z_{1:t}$ are known, the posterior over the map $p(m|x_{1:t}, z_{1:t})$ is usually computed analytically [140]. Particle filters are a way of evaluating the posterior over the trajectory positions $p(x_{1:t}|z_{1:t}, u_{1:t-1})$. A particle filter holds several particles, i.e., samples of the state of a dynamic system. After each observation it updates the particle set by computing the posterior distributions. GMapping uses the Rao-Blackwellized sample importance resampling (SIR) filter for mapping using laser scans. It proceeds in four steps to update the set of samples, namely sampling, importance weighting, resampling and map estimation. Each particle consists of a potential trajectory and a map built from this trajectory and the observations.

In the sampling step the old particles $x_{t-1}^{(i)}$ and the proposal distribution π are used to generate the new set of particles $x_t^{(i)}$. Next, according to the importance sampling principle the

importance weighting step assigns a weight

$$w_t^{(i)} = \frac{p\left(x_{1:x}^{(i)}|z_{1:t}, u_{1:t-1}\right)}{\pi\left(x_{1:t}^{(i)}|z_{1:t}, u_{1:t-1}\right)}$$

to each particle, to account for the fact that the proposal distribution π and the target distribution of the successor state are in general not equal. To achieve efficiency, following [56] π is chosen such that it fulfills the assumption:

$$\pi(x_{1:t}|z_{1:t}, u_{1:t-1}) = \pi(x_t|x_{1:t-1}, z_{1:t}, u_{1:t-1}) \cdot \pi(x_{1:t-1}|z_{1:t-1}, u_{1:t-2}),$$

allowing for a recursive computation of the importance weights:

$$w_t^{(i)} \propto \frac{p\left(z_t|m_{t-1}^{(i)}, x_t^{(i)}\right) p\left(x_t^{(i)}|x_{t-1}^{(i)}, u_{t-1}\right)}{\pi\left(x_t|x_{1:t-1}^{(i)}, z_{1:t}, u_{1:t-1}\right)} \cdot w_{t-1}^{(i)}.$$

The better the proposal distribution approximates the target distribution, the better the prediction will be. Using the odometry motion model $p(x_t|x_{t-1}, u_{t-1})$ is common practice, but compared to the information given by a laser scanner, the odometry information is substantially less precise. If the state space covered by the observation likelihood and the odometry model differs substantially, especially, if the meaningful range covered by the observation likelihood is substantially smaller, only a small fraction of the drawn samples covers the regions with high observation likelihood, thus the number of samples has to be increased to cover the meaningful area sufficiently. As the runtime grows with the number of particles, an increase of samples is to be avoided. Therefore, in GMapping the last sensor observation z_t is integrated into the proposal distribution increasing the focus on the meaningful regions of the observation likelihood and changing the weights to:

$$w_t^{(i)} = w_{t-1}^{(i)} \cdot \int p\left(z_t|x'\right) p\left(x'|x_{t-1}^{(i)}, u_{t-1}\right) \mathrm{d}x'.$$

In GMapping a proposal distribution π is calculated specifically for dense grid maps generated from 2D laser scanner data. To this end scan registration is performed to calculate the observation likelihood functions in a first step. The overall algorithm is applied to generate one new particle $(m_t^{(i)}, x_{1:t}^{(i)})$ from each of the particles $(m_{t-1}^{(i)}, x_{1:t-1}^{(i)})$. As new measurements u_{t-1} and z_t become available, the algorithm works as follows. First, for each particle $(m_t^{(i)}, x_{1:t}^{(i)})$ the current pose $x_t'^{(i)}$ is estimated using the odometry information u_{t-1}. Second, using the pose estimate $x_t'^{(i)}$, the map m_{t-1} and the observations z_t a scan matching algorithm determines the improved pose $\hat{x}_t'^{(i)}$. The scan matcher operates in a limited region around $x_t'^{(i)}$. If it fails, the odometry motion model is used to generate the proposal distribution. Otherwise, in the third step, K positions $\{x_j\}$ are sampled from an interval $L^{(i)}$ around $\hat{x}_t'^{(i)}$.

In the final step the new pose $x_t^{(i)}$ for particle i is selected from the Gaussian approximation $\mathcal{N}\left(\mu_t^{(i)}, \sum_t^{(i)}\right)$ with mean

$$\mu_t^{(i)} = \frac{1}{\eta^{(i)}} \cdot \sum_{j=1}^{K} x_j \cdot p\left(z_t | m_{t-1}^{(i)}, x_j\right) \cdot p\left(x_j | x_{t-1}^{(i)}, u_{t-1}\right),$$

variance

$$\sum_t^{(i)} = \frac{1}{\eta^{(i)}} \cdot \sum_{j=1}^{K} p\left(z_t | m_{t-1}^{(i)}, x_j\right) \cdot p\left(x_j | x_{t-1}^{(i)}, u_{t-1}\right) \cdot \left(x_j - \mu_t^{(i)}\right)\left(x_j - \mu_t^{(i)}\right)^T$$

and the normalization factor

$$\eta^{(i)} = \sum_{j=1}^{K} p\left(z_t | m_{t-1}^{(i)}, x_j\right) \cdot p\left(x_j | x_{t-1}^{(i)}, u_{t-1}\right).$$

The weight update is performed as:

$$w_t^{(i)} = w_{t-1}^{(i)} \cdot \eta^{(i)}.$$

After the new pose for a particle i is determined, the importance weights and the maps $m^{(i)}$ are updated according to the new poses $x_t^{(i)}$ and the observation z_t.

The algorithm keeps a finite number of particles. The complexity of the mapping algorithm is linear to the number of particles. Only the resampling process has a complexity of $\mathcal{O}(NM)$ where N is the number of particles and M is the size of the map. To limit the runtime and to avoid particle depletion, in most update steps the pose and the map of the particle are merely updated. Resampling is only performed when the resampling criterion N_{eff} falls below $N/2$, where N is the number of particles. N_{eff} is calculated following [56] using the normalized weight $\tilde{w}^{(i)}$ of particle i as:

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} \left(\tilde{w}^{(i)}\right)^2}$$

and is used to measure how well the particle set approximates the target posterior by measuring the dispersion of the importance weights.

5.3 Feature-based registration (FBR)

If no pose estimates were acquired during data collection the remaining option is to determine them directly from the data. Apart from range information modern laser scanners often capture the amount of light that is returned to the sensor. This information, known as reflectance, reflectivity or intensity value, can be used to detect features in the data. In terrestrial laser

scanning it is common practice to attach high reflective markers to the environment and detect them in the data to register the laser scans. This procedure takes a lot of time to place the markers in good positions, find them in the data and register the data. Often times the exact positions of the markers are also measured using a tachymeter extending the time needed in the field even further. As an additional disadvantage the markers will be visible in the data thus adulterating the scene and the mapping. Alternative methods use natural features rather than manually attached markers. Conditioned by the buildup of the laser scanning platform the points are captured as range, reflectance and spherical coordinates. This facilitates the generation of a panorama image using the spherical coordinates and the reflectance information as described in Chapter 4.1.3. The 3D data is thus projected onto an image enabling the use of image-based feature detection methods as presented in Section 4.2. Feature detection methods analyze the image and create a description of areas with high changes in intensity. The most commonly used features are Sift (Scale invariant feature transform) features. They also show superior performance for feature-based point cloud registration [107]. As the Sift feature detector works in gray scale images the panorama images from reflectance values of laser scans are ideally suited for feature matching. To register point clouds automatically corresponding features are detected in the panorama images of scan pairs.

Distinctive image features are those patches with a high change in reflectivity, i.e., a high gradient. Unfortunately high gradients also occur at edges. If these edges are jump edges, the features are naturally not stable over several scans, as the different scanning location and the different perspective will change the characteristics of the jump edges. Due to the availability of the depth information these features are easily removed from the detected features. For a quick filtering, a window of 3×3 image pixels around the feature is chosen and the corresponding points are evaluated based on their range values. If the range difference of the points exceeds a predefined threshold, the feature is considered unstable and is discarded. A more advanced method calculates the standard deviation of the chosen points instead and removes the features with a high standard deviation [104]. Removing the features increases the speed and the stability of the matching procedure likewise.

The matching procedure performs as follows. Feature correspondences found in two reflectance panoramas are used to calculate pairwise transformation matrices of the point clouds. For this purpose the algorithm identifies the feature in one image that is the closest to the sampled feature from the other image based on a comparison of their descriptors (see Fig. 5.3). Several algorithms such as the $k-$nearest neighbor (KNN) search and the radius KNN search are possible solutions to this problem. The ratio nearest neighbor search as presented by Lowe [129] has shown the most promising results. Three feature pairs are required to determine the transformation between two scans. To avoid testing all

$$\binom{N_f}{3} = \frac{N_f!}{3!\,(N_f-3)!} \tag{5.12}$$

combinations of subsets containing three points from a feature matching with N_f feature pairs, a RANSAC-like (RANdom SAmple Consensus) [72] method is used to determine the correct transformation. In an iterative approach, three matches are randomly chosen from the feature set. $M_f = \{\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3\}$ is the point set consisting of the three features from the first scan and $D_f = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3\}$ the point set consisting of the corresponding features from the second scan.

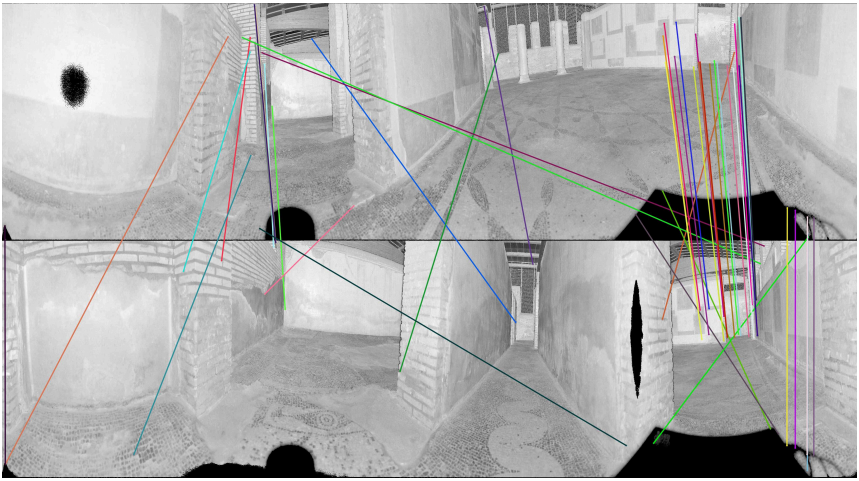


Fig. 5.3: Illustration of a scan pair with Mercator projection and matched features from the garden house in Ostia.

First, a quick check is performed by evaluating the triangle spanned by the three features from each point cloud in 3D. Second, if the sides of the triangle have similar length, the transformation between the two triangles is computed. As for the ICP algorithm, the triangles are moved such that their centroid defines the center of the reference frame to separate the rotation from the translation building the translated point sets M'_f and D'_f. Then, the rotation that minimizes the euclidean distance between the three point pairs is calculated using the closed form solution proposed by Horn [100]. Using the same notation as in Equation (5.8) with $N = 3$ the matrix $\boldsymbol{H}$ is constructed from the point pairs:

$$\boldsymbol{H} = \begin{pmatrix} S_{xx}+S_{yy}+S_{zz} & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx} \\ S_{yz}-S_{zy} & S_{xx}-S_{yy}-S_{zz} & S_{xy}+S_{yx} & S_{zx}+S_{xz} \\ S_{zx}-S_{xz} & S_{xy}+S_{yx} & -S_{xx}+S_{yy}-S_{zz} & S_{yz}+S_{zy} \\ S_{xy}-S_{yx} & S_{zx}+S_{xz} & S_{yz}+S_{zy} & -S_{xx}-S_{yy}+S_{zz} \end{pmatrix}. \quad (5.13)$$

The Eigenvector $\boldsymbol{v} = (v_1, v_2, v_3, v_4)^T$ corresponding to the maximum Eigenvalue of $\boldsymbol{H}$ is the quaternion describing the rotation that solves the minimization. The rotation matrix is calculated from the quaternion as:

$$\boldsymbol{R} = \begin{pmatrix} v_1^2+v_2^2-v_3^2-v_4^2 & 2v_2v_3-2v_1v_4 & 2v_2v_4+2v_1v_3 \\ 2v_2v_3+2v_1v_4 & v_1^2-v_2^2+v_3^2-v_4^2 & 2v_3v_4-2v_1v_2 \\ 2v_2v_4-2v_1v_3 & 2v_3v_4+2v_1v_2 & v_1^2-v_2^2-v_3^2+v_4^2 \end{pmatrix}. \quad (5.14)$$

The translation $\boldsymbol{t}$ is then derived using the centroids $\boldsymbol{c}_m$ and $\boldsymbol{c}_d$ according to Equation (5.6). Third, the matching result is evaluated. All matched feature pairs with a distance less than

a threshold λ_d are considered as inliers. Throughout this thesis $\lambda_d = 0.5\,\text{m}$ is chosen for all experiments. For a transformation that produces more than ten inliers the error metric Q is computed:

$$Q = E - IN_i^2, \tag{5.15}$$

with the summed distance E between matching features, the number of inliers N_i and a scaling factor I to scale the dimensionless number of inliers.

The three steps are repeated and the transformation that yields the highest quality value Q is chosen to register the scan. Technically, any of the methods presented in [151] to solve the 2-scan ICP algorithm can be used to compute the transformation, especially the SVD-based computation presented in section 5.1. Throughout the experiments in this thesis, the method implemented in 3DTK [149] is used that calculates the rotation using quaternions.

Depending on the projection method used for the panorama generation the optimal image size varies. The larger the image, the more detail is depicted. At the same time, however, the processing time for the registration increases. Therefore image size optimization can be carried out by calculating the optimized size of the image based on the aspect ratio of the projection to reduce the extra distortion introduced by fitting a panorama image to a fixed size image [106].

5.4 Registration with scale

The ICP algorithm and the feature-based registration calculate the rotation and translation between two point clouds based on point correspondences. For the ICP algorithm these correspondences are determined based on proximity and need to be repeated iteratively, while for FBR they are determined on features. The calculation requires the point clouds to be identically scaled. This cannot always be guaranteed for data coming from different sources. Point clouds generated using SfM approaches have generally no scale. To register them using the methods suggested here, the algorithm has to be slightly adjusted. Here the variant of the sICP presented in [94] following the derivation by Horn [100] is used. Given two point clouds M' and D' that are centered onto their centroids the scale factor s is calculated as

$$s = \sqrt{\frac{\sum_{i=1}^{N} ||\boldsymbol{m}_i||^2}{\sum_{i=1}^{N} ||\boldsymbol{d}_i||^2}} \tag{5.16}$$

where N is the number of point correspondences. The scale factor is independent of the rotation. In case of perfect correspondences, equation (5.16) can directly be applied. If correspondences are unknown as for the ICP, it is necessary to extend the correspondence search to a bidirectional search, i.e., not only for all points in D' the correspondences in M' are determined, but also for all points in M' the closest points in D'. Let $(\boldsymbol{R}, \boldsymbol{t})$ be the transformation calculated in the ICP algorithm, then the new transformation $(\boldsymbol{R}_s, \boldsymbol{t}_s)$ that incorporates the scale factor is given by

$$\begin{aligned} \boldsymbol{R}_s &= s\boldsymbol{R}, \\ \boldsymbol{t}_s &= \boldsymbol{c}_m - s\boldsymbol{R}\boldsymbol{c}_d. \end{aligned}$$

5.5 Uncertainty-based registration

The success of a registration algorithm depends on the quality of the correspondences. To account for the uncertainty in the feature matching process it is often desired to give a notion of the uncertainty of the registration process, i.e. the uncertainty of the poses calculated by the registration process. A probabilistic approach to the scan registration was presented in [130] and extended to 6 DoF in [29,30]. Using Euler angles the pose $\boldsymbol{X}$, the pose estimate $\bar{\boldsymbol{X}}$ and the pose error $\Delta\boldsymbol{X}$ are represented as:

$$\boldsymbol{X} = \begin{pmatrix} t_x \\ t_y \\ t_z \\ \theta_x \\ \theta_y \\ \theta_z \end{pmatrix}, \bar{\boldsymbol{X}} = \begin{pmatrix} \bar{t}_x \\ \bar{t}_y \\ \bar{t}_z \\ \bar{\theta}_x \\ \bar{\theta}_y \\ \bar{\theta}_z \end{pmatrix}, \Delta\boldsymbol{X} = \begin{pmatrix} \Delta t_x \\ \Delta t_y \\ \Delta t_z \\ \Delta\theta_x \\ \Delta\theta_y \\ \Delta\theta_z \end{pmatrix}.$$

Following [151] the error function (5.7) is changed to

$$E_{\text{pose}}(\boldsymbol{X}) = \sum_{i=1}^{N} ||\boldsymbol{X}_d \oplus \boldsymbol{d}_i - \boldsymbol{m}_i||^2 = \sum_{i=1}^{N} ||\boldsymbol{Z}_i(\boldsymbol{X})||^2 \tag{5.17}$$

denoting the positional error of a scan at pose $\boldsymbol{X}_d$ using the notation from [130]. The compounding operation $\oplus$ transforms a point $\boldsymbol{d}_i$ from the scanner coordinate system into the global coordinate system:

$$\boldsymbol{d}'_i = X_d \oplus \boldsymbol{d}_i = \begin{pmatrix} t_x \\ t_y \\ t_z \\ \theta_x \\ \theta_y \\ \theta_z \end{pmatrix} \oplus \begin{pmatrix} d_{x,i} \\ d_{y,i} \\ d_{z,i} \end{pmatrix} = \begin{pmatrix} d'_{x,i} \\ d'_{y,i} \\ d'_{z,i} \end{pmatrix}$$

with

$$\begin{aligned}
x'_{d_i} &= t_x - d_{z,i}\sin\theta_{t_y} + \cos\theta_{t_y}(d_{x,i}\cos\theta_{t_z} - d_{y,i}\sin\theta_{t_z}) \\
y'_{d_i} &= t_y + d_{z,i}\cos\theta_{t_y}\sin\theta_{t_x} + \cos\theta_{t_x}(d_{y,i}\cos\theta_{t_z} + d_{x,i}\sin\theta_{t_z}) + \sin\theta_{t_x}\sin\theta_{t_y}(d_{x,i}\cos\theta_{t_z} - d_{y,i}\sin\theta_{t_z}) \\
z'_{d_i} &= t_z - \sin\theta_{t_x}(d_{y,i}\cos\theta_{t_z} + d_{x,i}\sin\theta_{t_z}) + \cos\theta_{t_x}(d_{z,i}\cos\theta_{t_y} + \sin\theta_{t_y}(d_{x,i}\cos\theta_{t_z} - d_{y,i}\sin\theta_{t_z})).
\end{aligned}$$

Assuming small pose errors $\Delta\boldsymbol{X}$, the Taylor expansion linearizes the error E_{pose} (5.17) by approximating it using the pose estimates:

$$\boldsymbol{Z}_i(\boldsymbol{X}) \approx \bar{\boldsymbol{X}} \oplus \boldsymbol{d}_i - \boldsymbol{m}_i - \nabla\boldsymbol{Z}_i(\bar{\boldsymbol{X}})\Delta\boldsymbol{X} = \boldsymbol{Z}_i(\bar{\boldsymbol{X}}) - \nabla\boldsymbol{Z}_i(\bar{\boldsymbol{X}})\Delta\boldsymbol{X}$$

$\nabla\boldsymbol{Z}_i(\bar{\boldsymbol{X}})$ is the Jacobian of the compounded pose. To separate the pose information from the point information a matrix decomposition $\nabla\boldsymbol{Z}_i(\bar{\boldsymbol{X}}) = \boldsymbol{M}_i\boldsymbol{H}$ is applied. $\boldsymbol{M}_i$ is constructed in a

way that it contains solely the point information:

$$\boldsymbol{M}_i = \begin{pmatrix} 1 & 0 & 0 & 0 & -d_{y,i} & -d_{z,i} \\ 0 & 1 & 0 & d_{z,i} & d_{x,i} & 0 \\ 0 & 0 & 1 & -d_{y,i} & 0 & d_{x,i} \end{pmatrix},$$

resulting in the matrix $\boldsymbol{H}$ that represents the pose information:

$$\boldsymbol{H} = \begin{pmatrix} 1 & 0 & 0 & 0 & \bar{t}_z \cos(\bar{\theta}_x) + \bar{t}_y \sin(\bar{\theta}_x) & \bar{t}_y \cos(\bar{\theta}_x) \cos(\bar{\theta}_y) - \bar{t}_z \cos(\bar{\theta}_y) \sin(\bar{\theta}_x) \\ 0 & 1 & 0 & -\bar{t}_z & -\bar{t}_x \sin(\bar{\theta}_x) & -\bar{t}_x \cos(\bar{\theta}_x) \cos(\bar{\theta}_y) - \bar{t}_z \sin(\bar{\theta}_y) \\ 0 & 0 & 1 & \bar{t}_y & -\bar{t}_x \cos(\bar{\theta}_x) & \bar{t}_x \cos(\bar{\theta}_y) \sin(\bar{\theta}_x) + \bar{t}_y \sin(\bar{\theta}_y) \\ 0 & 0 & 0 & 1 & 0 & \sin(\bar{\theta}_y) \\ 0 & 0 & 0 & 0 & \sin(\bar{\theta}_x) & \cos(\bar{\theta}_x) \cos(\bar{\theta}_y) \\ 0 & 0 & 0 & 0 & \cos(\bar{\theta}_x) & -\cos(\bar{\theta}_y) \sin(\bar{\theta}_x) \end{pmatrix}. \quad (5.18)$$

The specific decomposition is chosen due to its similarity to the solution given in [130]. A more detailed description of the decomposition is given in [25]. Concatenating all the $\bar{\boldsymbol{Z}}_i =$ to a $3N$ size vector $\boldsymbol{Z}$ and all the $\boldsymbol{M}_i$ to the $3N \times 6$ matrix $\boldsymbol{M}$ yields the approximation of the pose error:

$$E_{\text{pose}} \approx (\boldsymbol{Z} - \boldsymbol{M}\boldsymbol{H}\Delta\boldsymbol{X})^T (\boldsymbol{Z} - \boldsymbol{M}\boldsymbol{H}\Delta\boldsymbol{X}).$$

Minimizing E_{pose} constitutes a least squares linear regression. Modeling the solution as a Gaussian distribution the ideal pose that solves this minimization problem is given by:

$$\bar{\boldsymbol{E}} = (\boldsymbol{M}^T \boldsymbol{M})^{-1} \boldsymbol{M}^T \boldsymbol{Z}$$

with the covariance estimation:

$$\boldsymbol{C} = s^2 (\boldsymbol{M}^T \boldsymbol{M}),$$

where s^2 is the unbiased estimate of the covariance of the identical, independently distributed errors of $\boldsymbol{Z}_i$:

$$s^2 = (\boldsymbol{Z} - \boldsymbol{M}\bar{\boldsymbol{E}})^T (\boldsymbol{Z} - \boldsymbol{M}\bar{\boldsymbol{E}}) / (2N - 3). \quad (5.19)$$

$\bar{\boldsymbol{E}}$ minimizes the linearized pose $\boldsymbol{H}\Delta\boldsymbol{X}$. To yield an optimal estimation of the robot pose it is necessary to update the poses and the covariances using the pose estimates and the computed solution:

$$\boldsymbol{X} = \bar{\boldsymbol{X}} - \boldsymbol{H}^{-1} \bar{\boldsymbol{E}},$$
$$\boldsymbol{C} = (\boldsymbol{H}^{-1}) \boldsymbol{C} (\boldsymbol{H}^{-1})^T.$$

5.6 The global ICP algorithm

The previously described methods have in common that inconsistencies occuring during the registration process remain in the data. When data is joined sequentially small local errors for

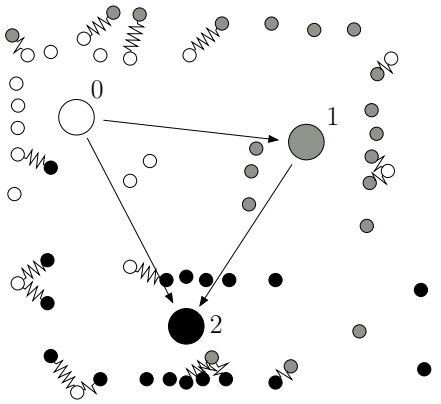


Fig. 5.4: Schematic view of the global optimization of three scans. Each scan is shown in a different gray value, the large circle represents the scan position. The correspondences resemble springs and the algorithm tries to move the scans in a way that the force on the springs is minimized. A local algorithm moves each scan individually. In the example, scan 2 would be registered to scan 1. With only few correspondences in the same area, this registration is doomed to fail. With the additional correspondences to scan 0, the chance to register the data is increased.

the registration of each pair add up and lead to larger errors for long sequences. To overcome this issue a global optimization was proposed by Lu and Milios [130] for 2D data and extended to 3D in [29] that is essentially a global variant of the ICP algorithm.

Given a robot path with $n+1$ scan poses $\boldsymbol{X}_0, \ldots, \boldsymbol{X}_n$ a graph of corresponding scans is constructed and for each neighboring pair point pairs are determined. Fig. 5.4 visualizes the concept. The nodes V_i of the graph represent the scan poses X_i, the directed edges $D_{i,j}$ visualize the nearest neighbor search. As for the ICP, each scan has a connection to its predecessor. Instead of just registering the data sequentially in the example another link is added from scan 0 to scan 2, i. e., the nearest neighbor search tries to find a neighbor in scan 0 for each point from scan 2. The point pairs are visualized as small springs. With only very few correspondences between scan 2 and scan 1, the registration is doomed to fail. With the additional correspondence to scan 0 the chance to register the data is increased. This idea is incorporated into the error function. Instead of just having correspondences between one pair of scans the new error function E_{opt} solves for transformations for all scans simultaneously:

$$E_{\text{opt}} = \sum_{j \to k} \sum_{i=1}^{N_{j \to k}} ||\boldsymbol{R}_j \boldsymbol{m}_i + \boldsymbol{t}_j - (\boldsymbol{R}_k \boldsymbol{d}_i + \boldsymbol{t}_k)||^2. \tag{5.20}$$

The outer sum represents all edges of the graph. $N_{j \to k}$ is the number of point pairs found between scan j and k. Note that the nearest neighbor search is directional, i.e., for all points from scan k the nearest neighbor in scan j is searched. Methods to minimize E_{opt} are presented in [151].

Throughout this thesis the uncertainty-based minimization using Euler angles from [29] is used.

Assuming that the linear error metric $\boldsymbol{E}'_{j,k}$ relates all pose pairs $\boldsymbol{X}'_j$ and $\boldsymbol{X}'_k$ that are connected in the graph by edge $D_{i,j}$ the Mahalanobis distance is used to describe the global error of all poses

$$W = \sum_{j \to k} (\bar{\boldsymbol{E}}_{j,k} - \boldsymbol{E}'_{j,k})^T \boldsymbol{C}^{-1}_{j,k} (\bar{\boldsymbol{E}}'_{j,k} - \boldsymbol{E}'_{j,k}) = \sum_{j \to k} (\bar{\boldsymbol{E}}_{j,k} - (\boldsymbol{X}'_j - \boldsymbol{X}'_k)) \boldsymbol{C}^{-1}_{j,k} (\bar{\boldsymbol{E}}'_{j,k} - (\boldsymbol{X}'_j - \boldsymbol{X}'_k)).$$

The error between two poses is modeled by the Gaussian distribution $(\bar{\boldsymbol{E}}_{j,k}, \boldsymbol{C}_{j,k})$. Minimizing W leads to the optimal poses for all scans.

Let $\boldsymbol{H}$ be the signed incident matrix of the pose graph, $\boldsymbol{X}$ the concatenation of all the poses $\boldsymbol{X}'_1$ to $\boldsymbol{X}'_n$, $\bar{\boldsymbol{E}}$ the concatenated vector consisting of all the $\bar{\boldsymbol{E}}'_{j,k}$ and $\boldsymbol{C}$ the block-diagonal matrix comprised of $\boldsymbol{C}^{-1}_{j,k}$ as submatrices, then W is rewritten in matrix notation as:

$$W = (\bar{\boldsymbol{E}} - \boldsymbol{H}\boldsymbol{X})^T \boldsymbol{C}^{-1} (\bar{\boldsymbol{E}} - \boldsymbol{H}\boldsymbol{X}).$$

Minimizing this function yields new optimal pose estimates. The minimization of $\boldsymbol{W}$ is accomplished via the following linear equation system: The solution $\boldsymbol{X}_{\min}$ that minimizes Equation (5.21) and its covariance $\boldsymbol{C}_X$ is given by:

$$\boldsymbol{X}_{\min} = (\boldsymbol{H}^T \boldsymbol{C}^{-1} \boldsymbol{H})^{-1} \boldsymbol{H}^T \boldsymbol{C}^{-1} \bar{\boldsymbol{E}} \tag{5.21}$$

$$\boldsymbol{C}_X = (\boldsymbol{H}^T \boldsymbol{C}^{-1} \boldsymbol{H})^{-1}. \tag{5.22}$$

Substituting $\boldsymbol{H}^T \boldsymbol{C}^{-1} \boldsymbol{H} = \boldsymbol{G}$ and $\boldsymbol{H}^T \boldsymbol{C}^{-1} \bar{\boldsymbol{E}} = \boldsymbol{B}$ simplifies the equation:

$$\boldsymbol{X}_{\min} = \boldsymbol{G}^{-1} \boldsymbol{B}$$
$$\boldsymbol{C}_X = \boldsymbol{G}^{-1}.$$

The matrix $\boldsymbol{G}$ consists of the submatrices

$$\boldsymbol{G}_{j,k} = \begin{cases} \sum_{k=0}^{n} \boldsymbol{C}^{-1}_{j,k} & (j = k) \\ \boldsymbol{C}^{-1}_{j,k} & (j \neq k). \end{cases} \tag{5.23}$$

The entries of $\boldsymbol{B}$ are given by:

$$\boldsymbol{B}_j = \sum_{\substack{k=0 \\ k \neq j}}^{n} \boldsymbol{C}^{-1}_{j,k} \bar{\boldsymbol{E}}_{j,k}. \tag{5.24}$$

Solving the linear optimal estimation problem (5.22) is equivalent to solving the following linear equation system:

$$\boldsymbol{G}\boldsymbol{X} = \boldsymbol{B}. \tag{5.25}$$

However, the actual positional error of two poses $\boldsymbol{X}_j$ and $\boldsymbol{X}_k$ is not linear. Using the same notation as for the 2-scan case in section 5.5 the inner sum of the error function (5.20) is rewritten as:

$$\boldsymbol{E}_{j,k} = \sum_{i=1}^{m} \|\boldsymbol{X}_j \oplus \boldsymbol{d}_i - \boldsymbol{X}_k \oplus \boldsymbol{m}_i\|^2 = \sum_{i=1}^{m} \|\boldsymbol{Z}_i(\boldsymbol{X}_j, \boldsymbol{X}_k)\|^2 .$$

Analogously, the Taylor expansion of $\boldsymbol{Z}_i(\boldsymbol{X}_j, \boldsymbol{X}_k)$ is used to obtain the linearized pose difference $\boldsymbol{E}'_{j,k}$:

$$\boldsymbol{Z}_i(\boldsymbol{X}_j, \boldsymbol{X}_k) \approx \boldsymbol{Z}_i(\bar{\boldsymbol{X}}_j, \bar{\boldsymbol{X}}_k) - \left(\nabla_{\boldsymbol{X}_j} \boldsymbol{Z}_i(\bar{\boldsymbol{X}}_j, \bar{\boldsymbol{X}}_k)\Delta\boldsymbol{X}_j - \nabla_{\boldsymbol{X}_k} \boldsymbol{Z}_i(\bar{\boldsymbol{X}}_j, \bar{\boldsymbol{X}}_k)\Delta\boldsymbol{X}_k\right).$$

Here, $\nabla_{\boldsymbol{X}_j}$ refers to the derivative with respect to $\boldsymbol{X}_j$. Utilizing the same matrix decomposition $\boldsymbol{M}_i\boldsymbol{H}$ of $\nabla\boldsymbol{Z}_i(\bar{\boldsymbol{X}})$ as in the 2-scan case $\boldsymbol{Z}_i(\boldsymbol{X}_j, \boldsymbol{X}_k)$ is approximated as:

$$\boldsymbol{Z}_i(\boldsymbol{X}_j, \boldsymbol{X}_k) \approx \boldsymbol{Z}_i(\bar{\boldsymbol{X}}_j, \bar{\boldsymbol{X}}_k) - \boldsymbol{M}_i\boldsymbol{E}'_{j,k},$$

where $\boldsymbol{E}'_{j,k}$ is the linear error metric given by:

$$\begin{aligned}\boldsymbol{E}'_{j,k} &= (\boldsymbol{H}_j\Delta\boldsymbol{X}_j - \boldsymbol{H}_k\Delta\boldsymbol{X}_k)\\ &= (\boldsymbol{X}'_j - \boldsymbol{X}'_k).\end{aligned}$$

$\boldsymbol{E}'_{j,k}$ is linear in the quantities $\boldsymbol{X}'_j$ that will be estimated by the algorithm. Again, the minimum of $\boldsymbol{E}'_{j,k}$ and the corresponding covariance are given by

$$\bar{\boldsymbol{E}}_{j,k} = (\boldsymbol{M}^T\boldsymbol{M})^{-1}\boldsymbol{M}^T\boldsymbol{Z} \tag{5.26}$$

$$\boldsymbol{C}_{j,k} = s^2(\boldsymbol{M}^T\boldsymbol{M}). \tag{5.27}$$

Here $\boldsymbol{Z}$ is the concatenated vector consisting of all $\boldsymbol{Z}_i = \bar{\boldsymbol{X}}_j \oplus \boldsymbol{d}_i - \bar{\boldsymbol{X}}_k \oplus \boldsymbol{m}_i$.

For the method to work, the reference pose has to be fixed. Assuming the reference pose to be $\boldsymbol{X}_0 = 0$ to obtain the optimal pose estimates the results have to be transformed just like in the 2-scan case.

$$\begin{aligned}\boldsymbol{X}_j &= \bar{\boldsymbol{X}}_j - \boldsymbol{H}_j^{-1}\boldsymbol{X}'_j,\\ \boldsymbol{C}_j &= (\boldsymbol{H}_j^{-1})\boldsymbol{C}_j^X(\boldsymbol{H}_j^{-1})^T.\end{aligned}$$

If $\boldsymbol{X}_0$ is nonzero, the solutions have to be transformed by:

$$\begin{aligned}\boldsymbol{X}'_i &= \boldsymbol{X}_0 \oplus \boldsymbol{X}_i\\ \boldsymbol{C}'_i &= \boldsymbol{K}_0\boldsymbol{C}_i\boldsymbol{K}_0^T\end{aligned}$$

where

$$\boldsymbol{K}_0 = \begin{pmatrix} \boldsymbol{R}_{\theta_{x_0},\theta_{y_0},\theta_{z_0}} & 0 \\ 0 & \boldsymbol{I}_3 \end{pmatrix}$$

Algorithm 12 Optimal estimation algorithm

1. Compute the point correspondences $\boldsymbol{d}_i^k, \boldsymbol{m}_i^j$.
2. For any link $D_(j,k)$ in the given graph compute the measurement vector $\bar{\boldsymbol{E}}_{j,k}$ by Eq. (5.26) and its covariance $\boldsymbol{C}_{j,k}$ by Eq. (5.27).
3. From all $\bar{\boldsymbol{E}}_{j,k}$ and $\boldsymbol{C}_{j,k}$ form the linear system $\mathbf{GX} = \mathbf{B}$, with $\mathbf{G}$ and $\mathbf{B}$ as given in Eq. (5.23) and (5.24) respectively.
4. Solve for $\mathbf{X}$.
5. Update the poses and their covariances.

with a rotation matrix $\boldsymbol{R}_{\theta_{x0},\theta_{y0},\theta_{z0}}$.

The algorithm for computing the global pose estimates is given as Algorithm 12. As for the ICP algorithm, the global optimization is executed iteratively for improving the results successively. Step 3 is sped up by component-wise computation of $\mathbf{G}$ and $\mathbf{B}$. The components $\boldsymbol{C}_{j,k}^{-1} = (\mathbf{M}^T\mathbf{M})/s^2$ and $\boldsymbol{C}_{j,k}^{-1}\bar{\boldsymbol{E}}_{i,j} = (\mathbf{M}^T\mathbf{Z})/s^2$ are expanded into summations, as shown in detail in [29]. The most expensive operation is solving the linear equation system $\mathbf{GX} = \mathbf{B}$. Since $\mathbf{G}$ is a positive definite, symmetric $6N \times 6N$ matrix, where N is the total number of point pairs, this is done by Cholesky decomposition in $\mathcal{O}(n^3)$.

5.6.1 Practical approach for fully automatic global optimization

A schematic view of a fully automatic method using both ICP and the global optimization is depicted in Fig. 5.5. The algorithm performs by sequentially adding scans using the ICP algorithm while new data comes in. After each addition a loop detection is performed using one of two options. Either a loop is detected if the current scan has a distance below a certain threshold or more than a predefined number of point correspondences to any of the already registered scans. For both methods a minimum loop size is given but the second method is computationally more expensive because it requires the calculation of point correspondences. If a loop is detected the graph is calculated and the global optimization is executed. Similar to the pair-wise ICP, the global optimization iteratively solves for a transformation. Note that also the graph is recalculated in each iteration to account for significant pose changes. Once this method converges, new data is again added via ICP until a new loop is detected.

This procedure is evaluated using the HANNOVER1 data set in [29] and a distance of 7.5 m for loop detection. Fig. 5.6 shows the effect of the global optimization on the registration. The data set consists of a large robot run with several loops. In the fourth loop one can clearly see, how a single wrong registration in the pairwise approach at a bend can lead to large inconsistencies in the map. Most SLAM approaches do not recover from such a problem. When detecting the loop closure in the HANNOVER1 data set, the global ICP generates a graph connecting the scans from the first traversal with the scans from the second traversal of the same path. While the scan positions are moved closer together, more links are added in the graph. After 20 iterations

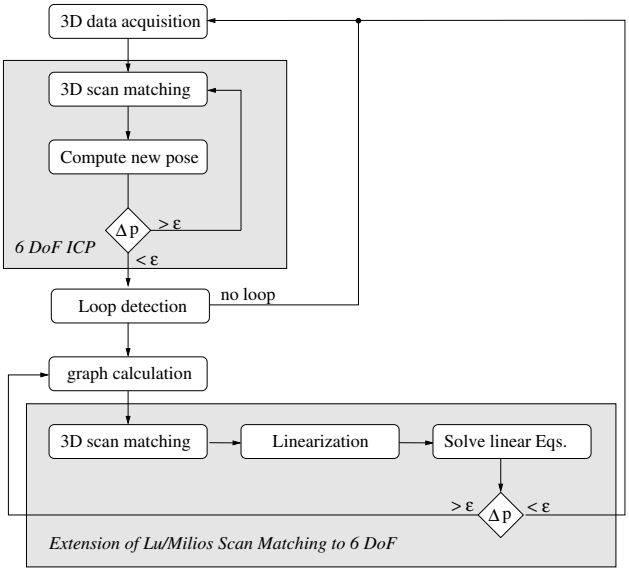


Fig. 5.5: The interaction of ICP and and the global optimization in the overall algorithm. The threshold ε used to determine if a scan changed its pose is the same in both algorithm parts. (Source: [29])

the graph has reached a stable configuration and the remaining iterations are used to remove small registration errors.

5.7 Evaluating pose estimates for 3D mapping

In the HANNOVER1 data set example shown above data was acquired with a rotating SICK laser scanner. Due to the short range frequent 3D scans were taken along the path. With the short distance between consecutive scans odometry yielded sufficient pose estimates for the ICP algorithm. When using a terrestrial laser scanner the increased range allows to take measurements far less frequently. As the odometry errors increase with growing path length this section evaluates different methods to generate the pose estimates for the ICP algorithm using the OSTIA and the WÜRZBURG RESIDENCE PALACE data sets.

5.7.1 Mapping evaluation using the OSTIA data set

A floor plan was created from a 2D orthogonal projection of the final model of the OSTIA data set without the protection roof and is depicted in Fig. 5.7. The scan positions are marked in the floor plan. The robot coordinate system is defined by the position of the robot at the start of

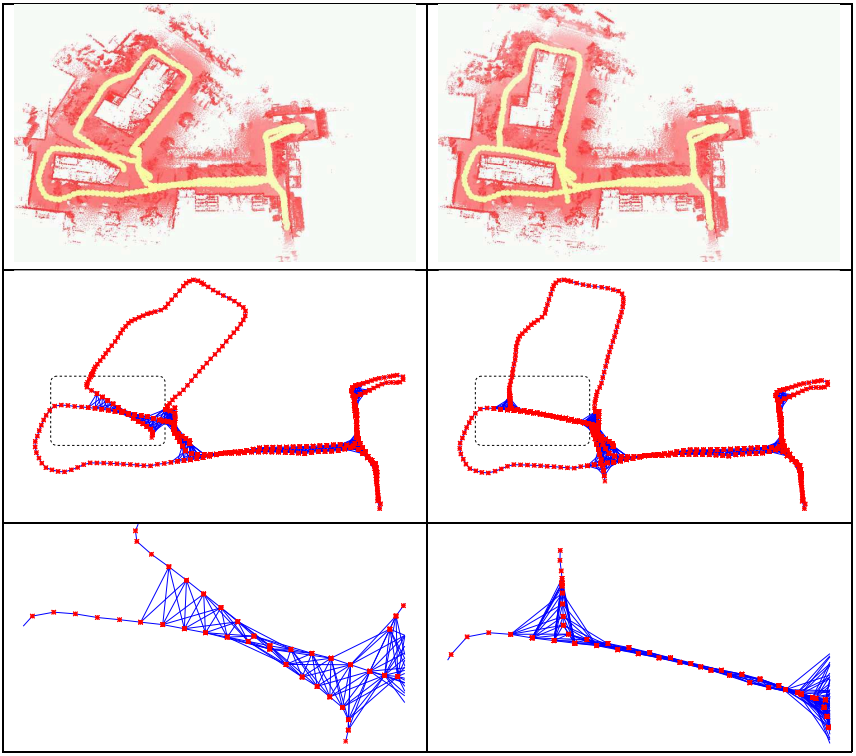


Fig. 5.6: LUM iterations before and after the fourth closed loop of the HANNOVER1 dataset. Above: Forth loop closing after 1 iteration and after 20 iterations. Middle: The corresponding graphs. Bellow: Zoom into the boxed areas. When turning into the upper loop, the pairwise registration fails. By loop closing and graph recomputation this failure is corrected and the robot pathes are merged. The zoomed-in graph representation shows how more links are added to the graph after merging the paths partially in the first iterations. (Source: [29])

the robot control program. Thus, for every run the coordinate system is different. We tried to overcome this issue by starting the robot at the same position every day. However, the inaccurate starting position in combination with the long and curvy robot path to the first scan of the run often lead to large errors in the pose estimates causing the odometry-based mapping algorithms to fail. The discontinuity of the path also caused problems for the sequential ICP algorithm as subsequent scans often have no overlap. Therefore, the poses were manually improved and scan pairs for matching with sufficient overlap manually chosen to create a coarse registration.

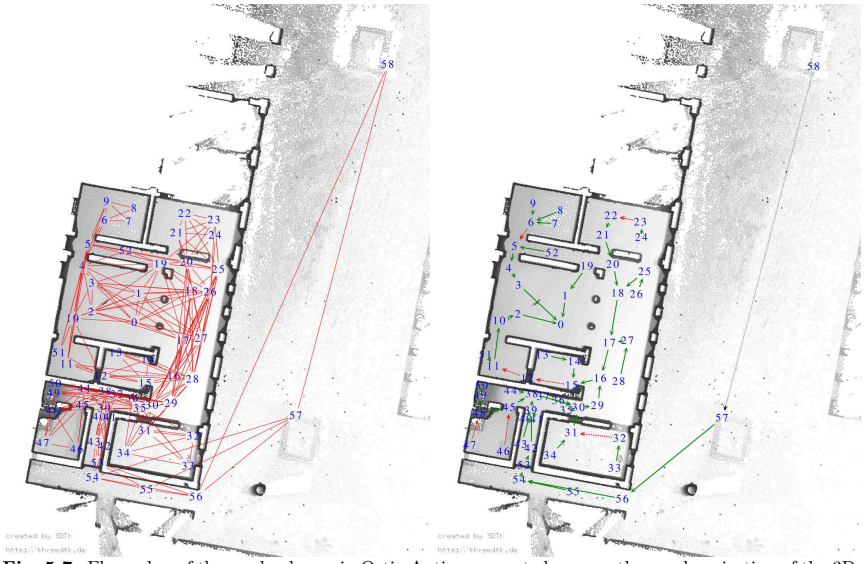
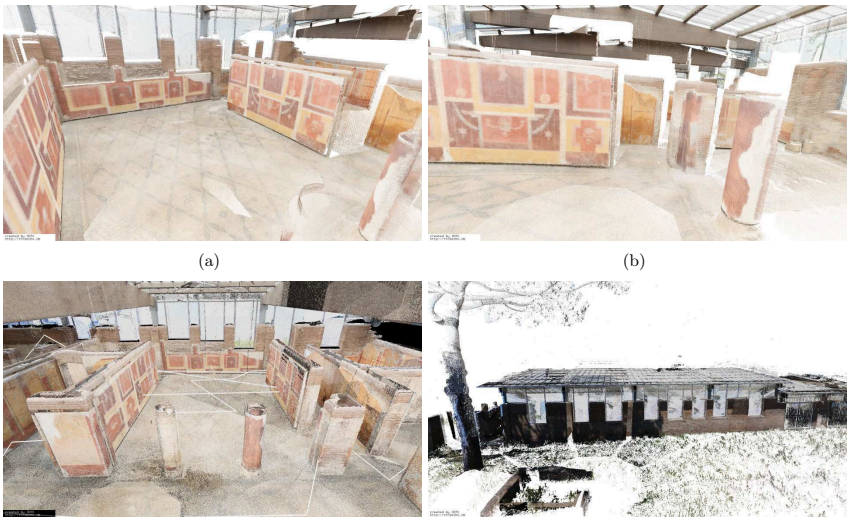


Fig. 5.7: Floor plan of the garden house in Ostia Antica generated as an orthogonal projection of the 3D model without the protection roof. The scan positions are marked. Left: The red lines show the graph used during the global optimization of the model reconstruction. Right: The arrows indicate how the scans were registered with the feature-based registration. Green arrows with solid lines indicate the scans that were registered with the standard settings. Red dashed lines mark those pairs where a modification of the parameters was necessary. The outdoor scan pair connected with a black dotted line was not registrable at all.

Afterwards, the global optimization was applied to create the final model using the graph shown in Fig. 5.7.

Thus the feature-based algorithm is used to automate the registration process further. The floor plan was used to determine which scans should have correspondences. In Fig. 5.7 the arrows point towards the scan each scan was registered to. Green arrows with solid lines mark the scan pairs that were successfully registered with the standard parameters. Red dashed lines mark those pairs where a modification of the parameters was necessary. The registration failed for only one scan pair, the two outdoor scans which had a distance of approximately 50 m between them and therefore very little resemblance in the panorama images. Images showing the final reconstruction are shown in Fig. 5.8. Also in the 3D representation no registration errors are apparent.



(a) (b)

(c) (d)

Fig. 5.8: The model of the garden house in Ostia Antica. (a) and (b) Two views of the data acquired on the first day. (c) The complete model with data from all days. The white lines mark the scanning sequence. (d) Outside view of the complete model.

5.7.2 Mapping evaluation using the Würzburg Residence Palace data

To evaluate the different pose estimation methods experiments were performed in the Würzburg Residence using the iSpace system to evaluate the quality of the resulting model comparing different mapping algorithms [33]. This section focuses on evaluating the mapping accuracy. Evaluation of the coloring process follows in Chapter 7.

To achieve precise pose estimates for the 3D laser scanner the localization sensors of the robot have to be calibrated to the laser scanner. The method used for odometry, IMU and the 2D laser scanner is explained in [66]. To localize the robot in the iSpace coordinate system using the attached iSpace sensor frame the sensor frame was calibrated to the VZ-400 laser scanner. After setting up the transmitters of the iSpace system several reflective markers were attached to objects in the environment. The centers of the markers were measured with the iSpace hand-vector bar, thus determining their position in the iSpace coordinate system. These markers show up clearly in the reflectance data of the laser scanner. To measure their precise position first a full scan of the environment is carried out. The RiScan Pro Software is used to detect the markers in the environment. An automatic procedure exists, but due to the fact that it creates a high number of false detections the markers are selected manually from the data as they are clearly

visible due to their high reflectivity. In a second step, fine scans of the markers are performed. The software controls the scanner automatically to scan the area around the selected markers with a very high resolution. If the existence of the marker is verified, its precise position in the local coordinate system of the scan is calculated. Third, the coordinates of the markers in the coordinate system defined by iSpace are imported as control points and the scans registered to these control points based on the marker position. This yields the position and orientation of the laser scanner in the iSpace coordinate system at the time the scan was taken. Additionally, the pose of the sensor frame is also recorded. In the following poses will be treated as transformation matrices $\boldsymbol{T}$, consisting of the rotation $\boldsymbol{R}$ and the translation $\boldsymbol{t}$. Repeating this procedure for n scans gives n pairs of poses for the Riegl laser scanner $\boldsymbol{T}_{r,i}$ and the sensor frame $\boldsymbol{T}_{m,i}$. From these poses the transformation $\boldsymbol{T}_{m\to r}$ between the coordinate systems is calculated as:

$$\boldsymbol{T}_{m\to r,i} = \boldsymbol{T}_{r,i}\boldsymbol{T}_{m,i}^{-1}. \tag{5.28}$$

To reduce noise the average over all transformation matrices $\boldsymbol{T}_{m\to r,i}$ is calculated as:

$$\bar{\boldsymbol{T}}_{m\to r} = \frac{1}{n}\sum_{i=0}^{n-1}\boldsymbol{T}_{m\to r,i}. \tag{5.29}$$

This procedure works for the translation but is not guaranteed to yield valid solutions for the rotation, i.e., an orthogonal matrix with determinant one. Instead projecting $\bar{\boldsymbol{R}}_{m\to r}$ onto $SO(3)$ [101] yields the nearest orthonormal matrix. Let $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, $\boldsymbol{e}_3$ be the Eigenvectors and λ_1, λ_2, λ_3 the Eigenvalues of the square matrix

$$\boldsymbol{HHs} = \bar{\boldsymbol{R}}_{m\to r}^{T}\cdot\bar{\boldsymbol{R}}_{m\to r} \tag{5.30}$$

then the final rotation matrix is calculated as:

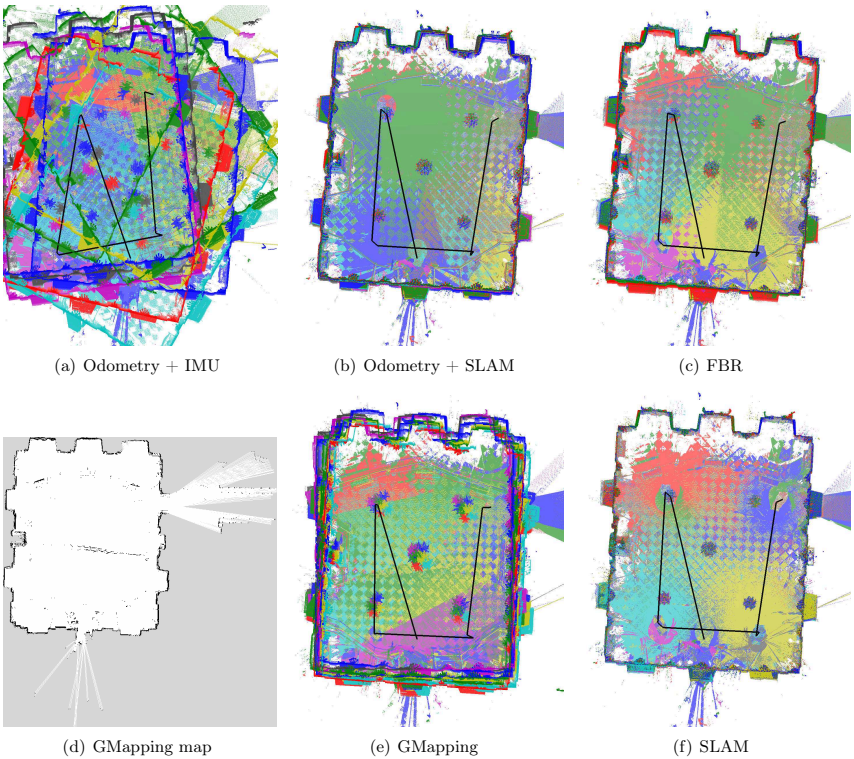
$$\hat{\boldsymbol{R}}_{m\to r} = \bar{\boldsymbol{T}}_{m\to r}\cdot\left(\frac{\boldsymbol{e}_1\cdot\boldsymbol{e}_1^T}{\sqrt{\lambda_1}} + \frac{\boldsymbol{e}_2\cdot\boldsymbol{e}_2^T}{\sqrt{\lambda_2}} + \frac{\boldsymbol{e}_3\cdot\boldsymbol{e}_3^T}{\sqrt{\lambda_3}}\right) \tag{5.31}$$

Afterwards, for each new scan position the position of the laser scanner in the iSpace coordinate system is calculated:

$$\hat{\boldsymbol{T}}_{r,i} = \hat{\boldsymbol{T}}_{m\to r}\boldsymbol{T}_{m,i}. \tag{5.32}$$

White Hall

During first experiments in the White Hall of the Residence the robot was manually driven to 9 positions to collect data. The starting position was close to the door leading to the Imperial Hall. The other 8 positions were chosen in the corners of the hall facing once towards the wall and once the other way to account for the tilted mount of the laser scanner. Fig. 5.9 illustrates some of the localization methods, showing scan positions, marked by a black line, and the resulting 3D model. It is obvious that the pose estimates from odometry contain large errors. In an environment such as this, they still suffice as initial pose estimates for ICP, in more complex



(a) Odometry + IMU (b) Odometry + SLAM (c) FBR

(d) GMapping map (e) GMapping (f) SLAM

Fig. 5.9: Floor plan of the White Hall. Projection of the 3D point cloud without the roof using poses from odometry and IMU (a), slam6D with pose estimates from odometry and IMU (b), feature-based registration (c), GMapping (e), slam6D with pose estimates from GMapping (f). Each scan is drawn in a different color. The black line connects the scan positions. (d) shows the map created from GMapping using the 2D laser scanner.

environments however they easily render the registration impossible. The map generated from GMapping seems to represent the environment well. The structure of the hall is clearly visible. Some faulty black lines are seen in the center of the hall. Using the pose estimates to position the 3D scans directly reveals some inaccuracies as the walls from different scans do not align perfectly. This is illustrated by using different colors for the individual scans. Applying the global optimization on top of the pose estimates from GMapping improves the results visibly. Not only the walls overlap but also the color distribution in the maps changes. Close to the

Fig. 5.10: The final 3D model of the White Hall in the Würzburg Residence.

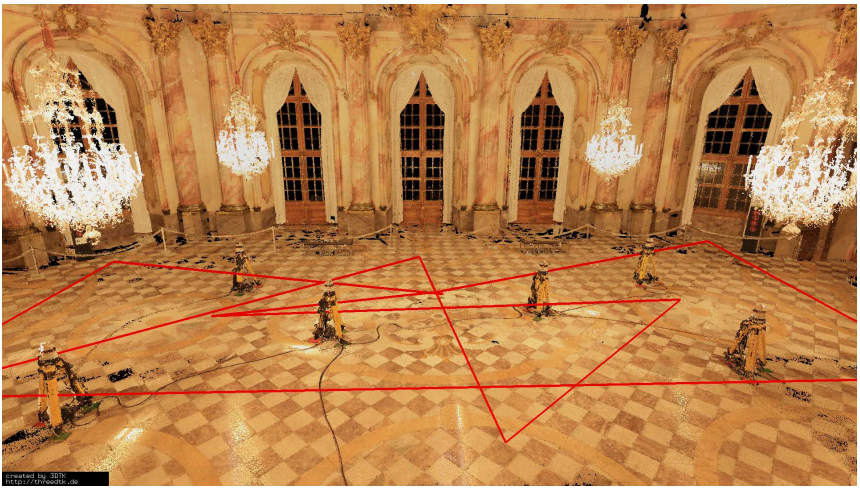


Fig. 5.11: Final 3D model of the Imperial Hall with transmitters and robot poses.

individual scanning positions the density of the color of that scan becomes higher, while in the other maps the distribution is more random. This effect is the same when comparing the results with start estimates from different methods. The final 3D model is displayed in Fig. 5.10 or can be seen in the video at [174].

Imperial Hall

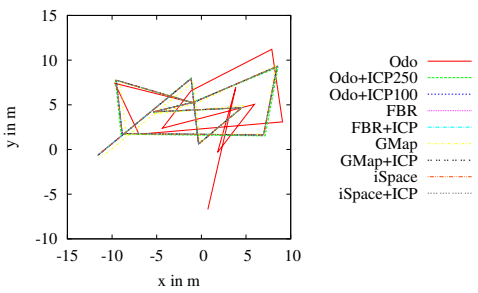
To achieve a quantitative evaluation of the methods the iSpace system is set up before data collection in the Imperial Hall. Six transmitters are set up and calibrated to define a coordinate system as depicted in Fig. 5.11. iSpace is then able to measure the robot position with high

precision using the sensor frame attached to the top of the laser scanner. Furthermore, points in the environment are measured using the hand-vector bar to evaluate the quality of the final model (cf. Fig. 3.6).

The robot was manually driven to 11 scanning positions. For the registration methods the pose of the first scan in anchored in the iSpace coordinate system. At each scanning position the position of the robot was recorded using iSpace. Fig. 5.12(a) shows the scan positions determined using the different localization systems. Again, odometry has large errors. All other systems yield similar pose estimates. After applying the global optimization to all of these estimates the final results are identical. The convergence limit was set to 0.1 cm movement per pose. Neighbor relations are established between all scans within a distance of less than 7.5 m.

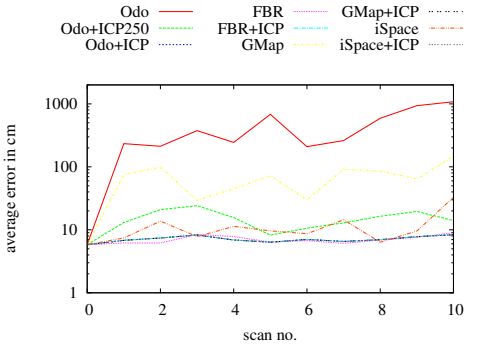
For a quantitative evaluation of the pose estimates 7 distinct points in the environment were measured using the iSpace hand-vector bar. These points were identified in each individual point cloud and transformed into the iSpace coordinate system using the pose estimates calculated with the different localization methods. For each scan the average of the distances between the transformed points and the globally measured point coordinates are calculated. Points that cannot clearly be identified in the point cloud are ignored in the averaging. The results are plotted in Fig. 5.12(b). As expected, the error for odometry pose estimates is tremendous. A registration is still possible due to the fact that the environment contains of a single large hall. However, this asks for two phases of the ICP method, first with a maximum distance of 250 cm between point pairs and then with a maximum distance of 100 cm to generate a correct model. Even though at first sight GMapping produces a feasible map, the missing accuracy becomes obvious here. This is caused by several factors, the limitations of the 2D laser scanner, the computational requirements to perform the 2D registration continuously and the use of a voxel map with inherent loss of precision. Decreasing the voxel size automatically increases the computational requirements.

Despite the promised accuracy of the iSpace system, the error in the model using this system directly for localization is surprisingly high. Possible reasons for this are interferences from the large windows and the glass chandeliers. Additionally, the rotational accuracy promised by the system is not as high as the positional accuracy. However, small rotation errors have already a large impact on the final model given the large distances to the walls. Furthermore, to maximize the observed area in the environment, the robot was moved to the corners of the hall, where the transmitter coverage was not optimal. For each positional measurement the system calculates an uncertainty, which is shown in Fig. 5.12(c). It can be seen that the large errors for the reference points correspond to the scans with high uncertainty. Feature-based registration yields the best initial pose estimates. They differ only very slightly from the final results. The remaining error is caused by three factors. First, not all points could be measured with high precision. Second, some of the reference points were in areas that were hard to capture with the laser scanner due to the reflective properties of the material. In combination with the low resolution in areas far away from the laser scanner the manually selected points lack precision. Third, the registration methods use the first scan as reference. The errors in the localization of this scan are therefore propagated to the remaining scans. Nevertheless, the evaluation shows that the error is minimal for the ICP approach independent of the method used for pose estimates. The final model of the Imperial Hall is depicted in Fig. 5.11. A video is available at [172, 173].



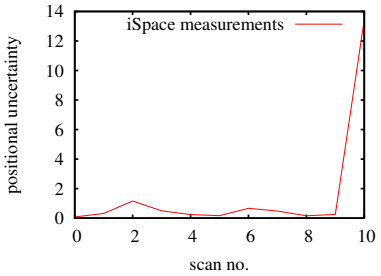
(a)

The poses using the different localization methods: Odo = Odometrie and IMU, ICP = Global optimization with a maximum distance of 100 cm between points, ICP250 = Global optimization with a maximum distance of 250 cm between points, GMap = GMapping, FBR = Feature-based registration using the Mercator projection, Sift features and the Sift descriptor for panorama matching.



(b)

Average error in cm of the measured control points for the map generated using the different mapping methods.



(c)

Positional uncertainty calculated from the iSpace system for each scan position.

Fig. 5.12: Evaluation of the iSpace measurements.

5.8 Summary

An occlusion free 3D representation of the environment requires joining point clouds from different positions into one common coordinate system. Contrary to feature-based methods point-based algorithms work on the raw points and are not dependent on the existence of features in the data. In this thesis the ICP algorithm is chosen as it requires no preprocessing. It does however, rely on initial pose estimates. Various methods for determining these pose estimates are evaluated in this chapter. It is shown that for robots equipped with a short range laser scanner that move short distances between consequtive scans the wheel odometry yields sufficient estimates.

The ICP gives locally good results. Nevertheless, small inaccuracies remain and are propagated to the following scans on a long robot run. When closing the loop the resulting large offsets become evident and lead to inconsistent maps. To overcome this issue a GraphSLAM solution is propsed that automatically builds a graph over all neighboring poses and minimizes the point-to-point distances between all connected scans simultaneously. It is shown that this method even has the ability to recover from matching failures of individual scans and yields globally consistent maps.

When taking scans less frequently, which is typically the case for long range laser scanners, the ICP often cannot recover from the summed up odometry errors. Rotational errors are especially problematic. In these cases other estimates are required. Here feature-based methods are a good choice. They utilize the precise 3D measurements, thus giving precise pose estimates already. While they are computationally complex, they are also applicable for data from terrestrial laser scanning.

Less precise but in real time available estimates are obtained from 2D mapping. By maintaining a 2D map small odometry errors are immediately compensated. The comparibly low density and precision of the 2D laser scanner in combination with the discretized map reduced the accuracy of the pose estimates. As input for the ICP algorithm they are shown to be sufficient. The ability to calculate 2D maps in real time is used in the next chapter to plan the scanning positions for exploration on the fly.

Chapter 6

Path planning

Sensor placement planning is needed for the goal directed acquisition of 3D data. The task of a sensor placement planning algorithm is to find a set of sensor configurations needed for obtaining a detailed environment model. Since a typical 3D laser scan takes 3 to 5 minutes for one position, depending on the resolution, it is desirable to minimize the number of scanning positions. This leads to an optimization problem similar to the Art Gallery Problem (AGP) (where to place guards such that the entire gallery is guarded). The AGP problem is NP hard and is usually solved by heuristics that perform well in practice [90]. These methods are categorized as model-based sensor placement planning (*a priori* model of the environment is known) and non-model-based methods. The latter are applied for exploration tasks in which the robotic system has to navigate autonomously in an unknown environment and build its own model. The planner must determine the next best view (NBV) based on the information collected from previous scans. Most exploration strategies push the robot onto the border between explored and unexplored regions [62, 197]. The majority of exploration algorithms is not reliable when applied under real world conditions due to sensitivity to uncertainty of measurements, localization, and map building. A small divergence in localization at the pre-computed NBV point can lead to many unnecessary movements. Moorehead et al. include the uncertainty of the robot pose into the exploration strategies [138]. Recently, numerous sensor placement planning algorithms have been developed for the reconstruction of 3D environment models. Most methods take 3D scans based on a 2D exploration strategy [189]. For creating a full 3D thermal model of a building it is essential to consider the 3D geometry of the environment to ensure that all parts of the building are mapped. Blaer and Allen propose a 3D NBV method which plans additional viewing locations based on a voxel-based occupancy procedure for detecting holes in the model [20]. Low and Lastra present a full non-model-based 3D NBV method based on an exhaustive hierarchical 3D view metric evaluation [128]. However, computational complexity is still the major challenge in designing practical 3D NBV solutions. Just recently a novel voxel-based 3D NBV method was proposed [165] that focuses on maximizing the data collected of structural elements, i.e., walls, floors and ceiling. Similar to the approach presented in Section 8.1 they first detect the floor and the ceiling using a z-histogram and then determine the enclosing polygon of the remaining data projected onto a plane. For the irregular prism enclosing the the 3D data they train a support vector machine (SVM) to classify the voxels as `structure`, `occluded-structure`,

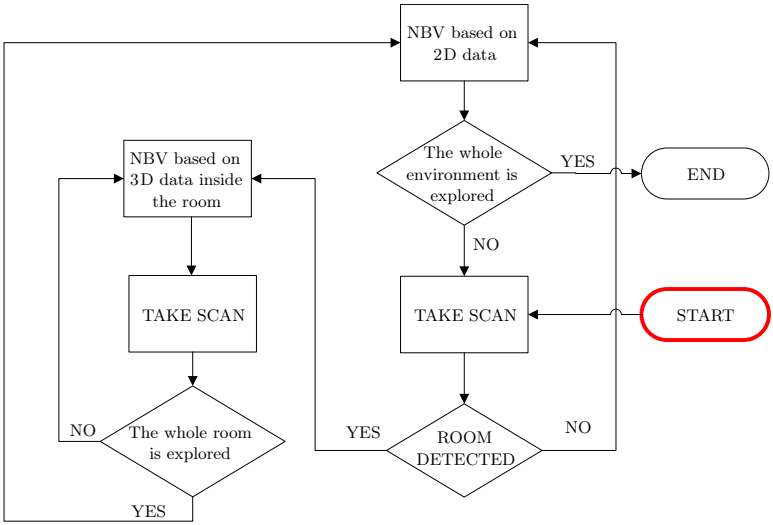


Fig. 6.1: Block scheme of the sensor placement planning algorithm

`clutter`, `occluded-clutter` and `empty`. The probabilistic next best view planner attempts to maximize the number of `occluded-structure` voxels seen from the next position. In their evaluation the approach presented here performs similar to their approach, even though the evaluation focuses on the number of structural element voxels and not on the total number of unseen voxels.

6.1 Sensor placement planning

The sensor placement planning module enables fully autonomous exploration in order to acquire a dense model of the environment. An algorithm based on the 2D horizontal plane in the point cloud at a specific height can be used efficiently for many tasks. However, this is not sufficient for creating a complete 3D model. Consider an empty room without obstacles. In a 2D scenario a robot with a horizontal field of view of 360° will finish after one scanning position because it perceives all the walls of the room from any position within the room. Constraints in the vertical field of view of the scanner that cause missing data at the floor and the ceiling or parts of the walls outside of the considered height level will simply be ignored. This is even more eminent in complex shaped rooms or when obstacles like chairs, tables, wardrobes, etc. are present in the room and occlude other objects at varying height. Consequently, if the aim is to have a complete 3D model, a method that searches for unexplored areas in 3D is needed.

Implementing and applying a complete 3D sensor placement planning algorithm in a large indoor environment needs significant memory space for storing the occupancy information of the whole environment. One would also need to store the exploration status of each part of the environment. All together this stretches the memory demands a lot. Additionally, the computational effort needed to process all the stored information will be high and will increase exploration time. The approach proposed in [39] combines 2D and 3D planning to enable the tracking of three-dimensional information of the environment with lower computation and memory demands. The block scheme of the overall sensor placement planning algorithm is depicted in Fig. 6.1. The main idea relies on a typical indoor environment structure, composed of enclosed spaces like rooms, halls, corridors and so on. Starting with exploration based on only 2D measurements and following the next best view (NBV) positions obtained from 2D planning, the robot detects an enclosed space, i.e. a room, and models it. At that moment the procedure of searching for the NBV position switches from 2D to 3D NBV planning, which takes into account the whole 3D environment information captured by the 3D laser scanner. The NBV planning algorithm based on 3D information explores only the detected room as a small unit of the large environment, thus only needing to store all the 3D information of this small part. Therefore, a combination of 2D and 3D information based exploration keeps the computational and memory requirements low, because only a small part of the 3D model is used while exploring. When the detected room is explored, the 3D NBV planning algorithm terminates and exploration continues again with the 2D NBV planning until a new unexplored enclosed space is detected and the algorithm switches back to 3D NBV planning. The algorithm terminates when the 2D NBV algorithm detects that the whole environment is explored, i.e. when there is not any so-called jump edge left in the memory. A jump edge is in this context an edge that separates explored and unexplored regions of the environment. The next sections describes the three main modules of the algorithm: 2D NBV planning, room detection and 3D NBV planning.

6.1.1 2D NBV planning algorithm

The 2D NBV planning algorithm is based on the approach presented in [4] and does not require any information about the environment beforehand. Initiated with a blank map it starts to explore the environment based on the first scan.

The inputs are range values uniformly distributed on the 360° field of view. The ranges are extracted from the 3D point cloud so that all range data lies in the plane parallel to the floor plane. To ensure that the robot does not hit any obstacles, a slice covering the entire height of the robot is used to create the map, i.e., all values that are between approximately 30 and 70 cm above the ground are used to create a 2D floor plan. The relatively large distance to the floor was chosen to account for small inaccuracies in the leveling of the robot.

Assuming that the environment model is initially unknown it is incrementally built after each scan. The model is hierarchical with three abstraction levels. At the lowest level the grid map is used to store the static and dynamic obstacle information needed for path planning and obstacle avoidance. The next abstraction level contains the polygonal representation of the environment which stores environment edges such as walls and other obstacles, which have been extracted from the range data, and jump edges. The most abstract level contains scanning position candidates which are considered for finding the NBV scanning position, i.e., the next goal position for

the path planning module. We assume a setup where the robot localization problem is solved. The GMapping module (cf. Section 5.2) under ROS [190] is used in our experiments. While exploring, the robot has to navigate between scanning positions in an unknown environment. We use a motion planning algorithm based on the D* algorithm and the Dynamic Window obstacle avoidance algorithm described in [179].

The 2D NBV planning algorithm is composed of three consecutive steps, which are executed at each scanning position: (1) vectorization – extracting lines from range data, (2) creation of the exploration polygon EP – building the most recent model of the environment, and (3) selection of the NBV sensor position – choosing the next goal for the path planning module. These three steps are explained in the following.

Vectorization

The main goal of the vectorization step is to obtain line segments from the input range data extracted from the 3D scan using the least squares method. First, the Progressive Probabilistic Hough Transform (PPHT) from the OpenCV library [134, 156] is applied to the range data to calculate an initial estimation of the line segments, which are then used to group all range data around the calculated line segments according to their distance $\Delta\rho_k$ from the lines. Second, more precise line parameters are calculated by a least squares line fitting algorithm.

Creation of the exploration polygon

A polygonal representation of the environment is used for selecting the NBV position, for creating the grid-map, and for path planning. The calculated line segments from the vectorization step form the measurement polygon as follows. The ending points of adjacent detected lines are connected with *jump edges* and define the polygon P_i at the ith scanning position p_i. The result is a polygon P_i that is composed of real line segments and artificial edges, i.e., jump edges, between them. However, some jump edges from the new scan might fall into the already explored area from previous scans. To discard those jump edges we use the union of the new polygon P_i (from the last scan) and the old exploration polygon EP_{i-1} (from previous scans) as a representation of the currently explored area and we discard jump edges within the union of the polygons (from GPC library [199]). In each step i, EP_i is updated as $EP_i = EP_{i-1} \bigcup P_i$. The union of two polygons keeps only those edges from both polygons that are most distant from the point of view of the robot, and thus ensures that new jump edges are not created within previously explored regions. Jump edges that are longer than the preset value Δr are considered for the selection of the next scanning position. The minimal length of the jump edge Δr is chosen in accordance with the robot dimensions and ensures that small jump edges are discarded, i.e., those the robot cannot pass through. If the non-empty extended polygon EP_i contains no reachable jump edge, it is considered as the reliable polygonal description and the exploration process stops.

Selection of the Next Best View scanning position

We use a heuristic criterion for selecting the NBV position similar to [62]. By taking a scan directly in front of the jump edge it is easy to imagine that we will gain a larger amount of

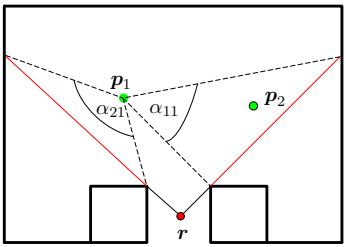


Fig. 6.2: Selection criterion based on angles of visibility.

new information than by scanning further away from the jump edge inside of the explored area. Therefore, one candidate scanning position is assigned to each jump edge. It is an obstacle free position near the mid point of the jump edge at distance d from the jump edge. d is chosen to be equal to the dimensions of the robot to ensure safety in case the jump edge is close to an obstacle that has not yet been detected. Additionally, d must be larger than the minimal sensor range. The next sensor position is chosen by maximization of a criterion that estimates the amount of unexplored regions seen from each potential position.

Fig. 6.2 shows two candidate positions $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ with jump edges denoted by red lines. The current scanning position is at $\boldsymbol{r}$. The measure of the size of the unexplored region that is possibly visible from the k-th candidate position is calculated from the angles in the triangles that are defined by the k-th candidate position and all jump edges. To maximize the information gain all jump edges have to be considered that are visible from the candidate position. In Fig. 6.2 both jump edges are visible from $\boldsymbol{p}_1$. Considering also the length d_j of the shortest path from the path planning module between the current robot position $\boldsymbol{r}$ and the j-th candidate position the selection criterion is as follows:

$$I_j = k_1 \frac{1}{d_j} + k_2 \sum_{i=1}^{N} \alpha_{ij}. \tag{6.1}$$

N is the number of candidate positions and α_{ij} is the angle in the triangle defined by the j-th candidate position and the i-th jump edge. Two parameters k_1 and k_2 are used as weighting parameters of angle and distance estimations, respectively. Numerous experiments in simulation and with the real robot showed good performance with k_1 set to the maximal range distance and k_2 set to $1/N$ which averages over all angles.

6.1.2 Room detection

The crucial step of the proposed sensor placement planning algorithm is the room detection, which is used to switch between the 2D and 3D NBV planning algorithms. After each scan taken at the position chosen by the 2D NBV planning, the room detection algorithm searches for the enclosed space (room) based on the current information captured from previous scans. If the room is detected, the 2D NBV planning is paused and the 3D NBV planning starts.

In the 2D exploration phase a 2D cut of the environment is used that represents the area that is traversable by the robot. For room detection such a slice close to the ground is not suitable because objects occlude the room boundaries and windows and doors interrupt them. The main idea for room detection is grounded on the detection of a closed space in the 2D polygonal line map of the environment obtained by vectorization of range data at the most suitable height level above the floor. The most suitable height level for room detection is an obstacle free 2D plane at the height where the room borders and walls are easily detected, i.e., a plane close to the ceiling. We choose it manually before the exploration starts. From the current 2D line map (taken at a suitable height level) of the workspace the room detection algorithm tries to detect a room. Let A be the lines corresponding to real environment edges obtained from the last scan. The algorithm searches for a room starting from the first line within the set A and tries to close the loop through other lines in the entire map. The process proceeds by finding the line closest to the current line in each step. When the loop is closed, i.e., when the nearest line is the starting line again, the room is detected. The nearest line search refers to the nearest ending point of a line in the vicinity of the current line. The vicinity area around the current line is defined with the radius parameter equal to the expected doorway width. With that, we ignore holes caused by doorways and windows inside the room. If there is no room detected starting from the first line in set A, the detection process starts from the second line in A. The order of the lines in A is not relevant. In case no closed polygon can be found with any line from set A as starting line, the room could not be detected in that step and exploration continues with the 2D NBV algorithm.

6.1.3 3D NBV planning algorithm

The room detection algorithm provides the room parameters including the boundary area and the coordinates of the room. When the room is detected, at least one scan inside the room is available since the robot has already entered the detected room and taken at least one scan inside which has been used for the room detection. From the available scans inside the room the initial 3D model of the room is built and the exploration continues by using the 3D model based NBV planning. The main idea of the 3D NBV planning algorithm is described hereafter.

To obtain NBV positions the room model needs to hold information of the explored and unexplored area inside the room. We use a voxel based representation where each voxel has one of the following labels: `occupied` if the volume within the voxel is occupied, `unseen` if the occupied status of the voxel is unknown or `empty` if the voxel is empty with no obstacles inside. The dimensions of the room define the number of voxels that should be used to cover the whole area of the room, i.e., memory allocation for the detected room. The shape of the enclosed space can be arbitrary. A cuboid 3D voxel model that encloses the entire detected room is chosen. The voxel model has to be large enough to enclose the largest room expected in the environment. Although there could be voxels in the cuboid model which are not part of the room, they are not considered during NBV position planning.

Once the room is detected, its voxel based 3D map is initialized with all voxels set to `unseen`, since we do not have any information on the environment. The stored scans that were taken inside the detected room are used to update its initial voxel based 3D model. This is performed by using the ray tracing algorithm which traces a ray from the position where the scan was taken

to each data point in the scan. Each voxel that is crossed by the ray is marked as `empty` and the voxel containing the data point is marked as `occupied`. The potential NBV position candidates are all voxels at the height of the laser scanner with the status `empty`. Position candidates that are not reachable for the robot are then removed from the list. The aim is to choose the candidate scanning position from where the most `unseen` voxels could be seen. For each candidate scanning position we count the number of `unseen` voxels. A ray is traced from the candidate scanning position to each `unseen` voxel and, if all crossed voxels are `empty`, the counter is incremented. Since considering every `unseen` voxel is unnecessarily time consuming only `unseen` voxels with at least one `empty` neighbor are taken into account [20]. In that way the number of voxels that need to be tested is decreased and voxels outside the room boundaries are not considered. The approach is similar to the jump edges in 2D in the way that `unseen` voxels that are taken into account actually corresponds to jump planes which divide explored and unexplored regions. Constraints in the field of view and range properties that are limits introduced by the sensors are considered by checking the range and the angle between the candidate scanning position and the `unseen` voxels. If the constraints are not satisfied, the `unseen` voxel is not counted. After finding a location that maximizes the number of unseen voxels, i.e., the NBV position, the robot drives to it, takes the 3D scan, and the whole procedure is repeated. The algorithm stops when the number of `unseen` voxels that can be seen from the best candidate position is below some predefined threshold V_{min} and the room is considered explored.

6.2 Planning results

The experiments carried out here are part of the Automation Lab data set. Fig. 6.3 shows some of the RGB images collected during the experiments. The aim was to build a complete 3D model of the environment based on 3D scans with thermal information attached. Each scan took 3 minutes and 15 seconds. The laser scanner is constrained by a vertical field of view of 100° and the thermal camera with a vertical opening angle of 60°, respectively. Since the laser scanner field of view exceeds that of the thermal camera, the constraints of the algorithm are set to consider the field of view of the camera. The voxel volume used in the experiment was set to $0.008\,\mathrm{m}^3$ ($0.2\,\mathrm{m} \times 0.2\,\mathrm{m} \times 0.2\,\mathrm{m}$). For creating the 2D map a slice of the 3D scan was taken at a height between approximately 30 cm and 70 cm above the ground to make sure the robot hits no obstacles at any height. The room detection height level was set to 2.5 m to avoid difficulties with windows and doors. Fig. 6.4 to 6.6 present the results of an experiment illustrating the behavior of the proposed sensor placement planning algorithm.

Fig. 6.4 shows the floor plan of the explored environment which consists of three rooms and a corridor. The scanning positions chosen by the proposed sensor placement planning algorithm are ordered from the start position (position 1) to the end position (position 13) and marked with the robot footprints. Green footprints refer to scanning positions generated by the 2D algorithms and red footprints by the 3D NBV algorithm, respectively. Some photos illustrating the environment are given in Fig. 6.3. The photos are excerpts from the data that the robot Irma3D acquired during the experiment. The robot started from position 1 in room 1 and took the initial scan. The first scan was sufficient for the room detection algorithm to detect room 1. When the room was detected, the jump edges inside room 1 were discarded, leaving only the

(a) (b) (c)

(d) (e) (f)

Fig. 6.3: Pictures of the environment taken by Irma3D during the experiments. The map with the scanning positions is given in Fig. 6.4. (a) and (b) room 1 seen from scanning position 2; (c) the corridor seen from position 6 facing towards position 4; (d) looking into room 3 from position 6; (e) the small corridor connecting room 2 and 3 seen from position 11; (f) room 2 seen from position 10.

jump edges detected outside of room 1 for continuing with the 2D NBV planning algorithm after exploring room 1. The initial 3D voxel model of the room was built and the 3D NBV planning algorithm was switched on. Then the 3D NBV planning algorithm chose position 2, took the scan there and updated the 3D model of room 1. Since the number of unseen voxels was larger than the threshold (V_{min} = 15 voxels) the 3D NBV algorithm continued with the exploration

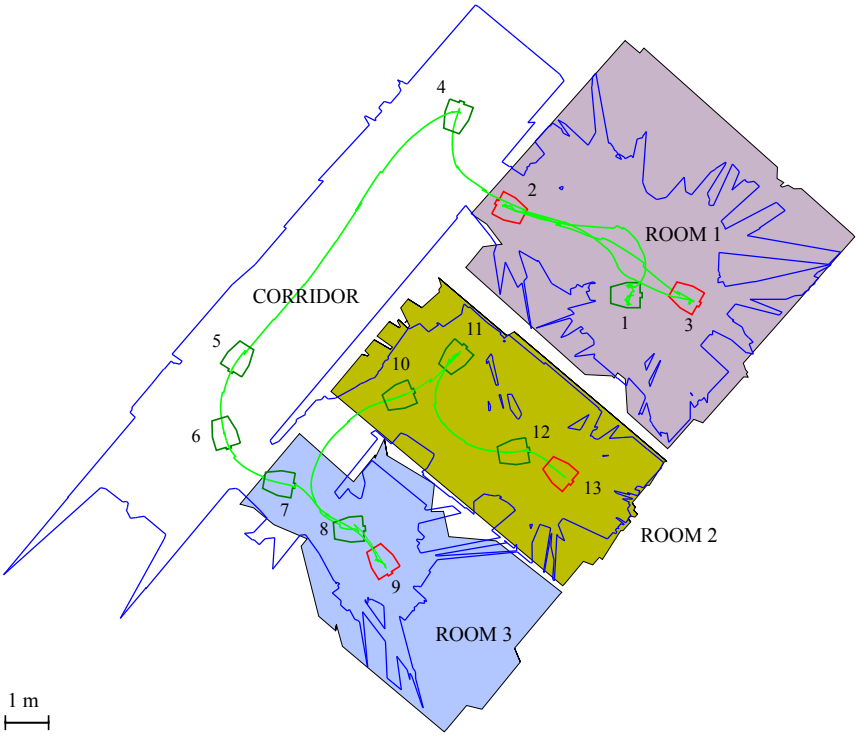
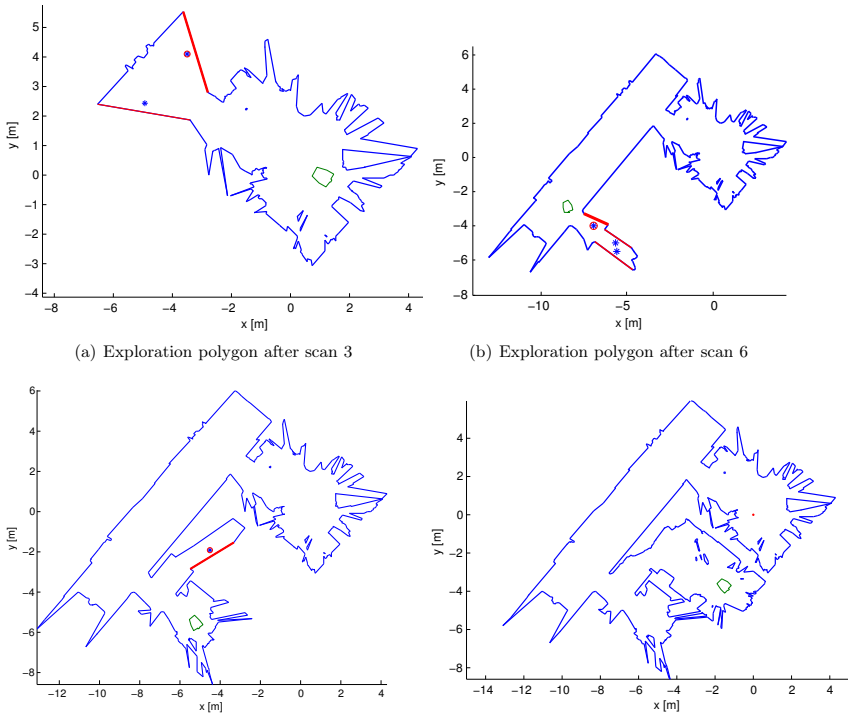


Fig. 6.4: Robot path and scanning positions during exploration.

of room 1 and chose scanning position 3. After the third scan, the 3D model of room 1 was of the desired accuracy and the exploration was continued with the 2D NBV algorithm, which moved the robot to position 4 in the corridor, i.e. in front of one of the two available jump edges (see Fig. 6.5(a)). The 2D NBV algorithm chose two additional scanning positions in the corridor (positions 5 and 6). The reason for the jump edge at position 5 is a range constraint of 8 m on the laser data in 2D causing part of the corridor not to be seen in the 2D laser data from position 4. Further away from the robot the points are very sparse making line extraction difficult. After the corridor, the robot moved to position 7 at the entrance of room 3, i.e. to the edge with the lowest value of the criterion (Fig. 6.5(b)) without exploring the corridor with the 3D NBV algorithm. The corridor was not considered a room since it was closed on both sides with glass doors as can be seen in Fig. 6.3(c). At the room detection height level of 2.5 m the



(a) Exploration polygon after scan 3

(b) Exploration polygon after scan 6

(c) Exploration polygon after scan 9

(d) Exploration polygon after scan 13

Fig. 6.5: Important exploration polygons during the experiment (jump edges are in red color and 2D NBV scanning positions marked with cross characters in red circles).

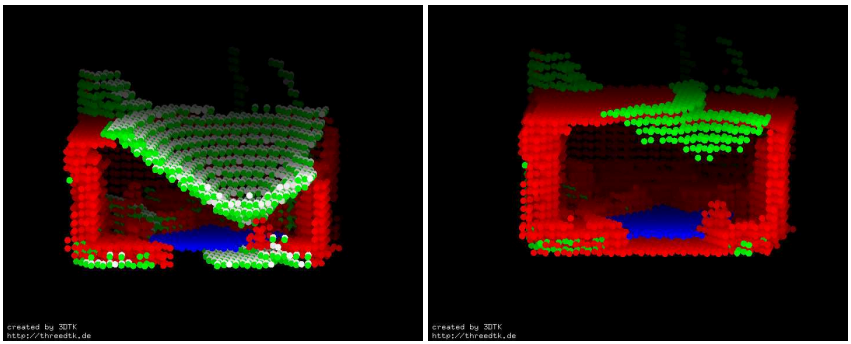
glass doors were not visible in the laser scan.

Due to the lack of clutter in the corridor three scanning positions were sufficient to satisfy the criteria of the 2D NBV algorithm. After scan 8 room 2 was detected and fully explored with only one additional position from the 3D NBV algorithm. Afterwards, 2D NBV planning chose position 10 based on the polygon shown in Fig. 6.5(c) and position 11 and 12 before room 3 was detected. Finally, position 13 was chosen by the 3D NBV algorithm to achieve the required accuracy of the room model. The exploration of the environment finished here as there were no jump edges in the memory (Fig. 6.5(d)), which means that the whole environment was explored and modeled.

Fig. 6.6 represents the 3D voxel models of room 1 after the first (6.6(a)) and the third (6.6(b))

Table 6.1: The number of `unseen (U)`, `occupied (O)` and the number of `unseen` voxels that can be seen from the best next position in 3D (`UB`) at each scan inside room 1.

Voxel type	Scan 1	Scan 2	Scan 3
`O` (*red*)	4091	6207	7034
`U` (*green*)	2104	1248	1207
`UB` (*white*)	965	203	0



(a) Initial model of room 1 after the first scan (b) Final model of room 1 after the third scan

Fig. 6.6: 3D voxel based models of room 1. `occupied` voxels are colored red, potential position voxels (`PP`) are colored blue, `unseen` voxels are colored green. White voxels are unseen voxels that can be seen from at least one `PP` position in the workspace.

scan. The `occupied` voxels are colored red, potential position voxels (`PP`) are colored blue, while `unseen` voxels are colored green. In the figure we also have white voxels, which are unseen voxels that can be seen from at least one `PP` position in the workspace. They are a subset of the `unseen` voxels and are treated as such in the exploration algorithm. They only serve to show that some of the `unseen` voxels could never be seen from any position. The specific cone in the figure is a consequence of the sensor field of view constraints. The other `unseen` (green) voxels are situated under the tables, chairs and other obstacles inside the room representing unexplored area. As can be seen in the model and in the Table 6.1, the number of `unseen` voxels that can be seen from the best position is zero after the last scan inside room 1.

To evaluate the benefits of the 3D exploration strategy, we conducted two new experiments in room 1, one using only 2D exploration and one with both 2D and 3D exploration. We placed the robot at the same starting position in both experiments. The results are seen in Fig. 6.7 and 6.8. Fig. 6.7 shows the final exploration polygon with the positions where the scans were taken. Rectangular marks show positions chosen only by 2D NBV and circles refer to 3D NBV

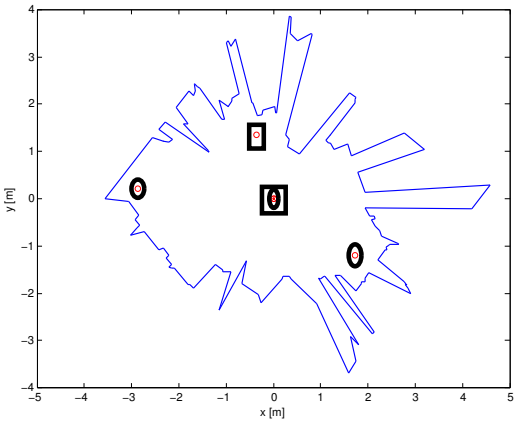


Fig. 6.7: Comparison of 2D and 3D NBV algorithm

Table 6.2: The number of `unseen (U)`, `occupied (O)` and the number of `unseen` voxels that can be seen from the best next position in 3D (`UB`) at each scan inside room 1 after 2D and 3D exploration.

	3D NBV			2D NBV	
Voxel type	Scan 1	Scan 2	Scan 3	Scan 1	Scan 2
`O` (*red*)	4068	5321	6106	4033	4958
`U` (*green*)	2275	1524	1192	2320	2036
`UB` (*white*)	1025	361	0	1017	703

based positions. In the 2D mode the robot finished after two scanning positions. As can be seen in the panorama image, large parts of the ceiling, the floor and the walls are not captured with the thermal camera. The unexplored area is drastically reduced when 3D exploration is employed as well. From Table 6.2 it becomes clear that 703 `unseen (UB)` voxels could still be seen from the NBV position after the 2D exploration.

6.3 Summary

This chapter presents a system for autonomous 3D exploration and thermal data acquisition of an indoor environment. To perceive the entire environment a model based 3D sensor placement approach is proposed that uses a voxel representation and room extraction to drastically decrease the computational requirements while still reaching high accuracy. Thus a system is designed

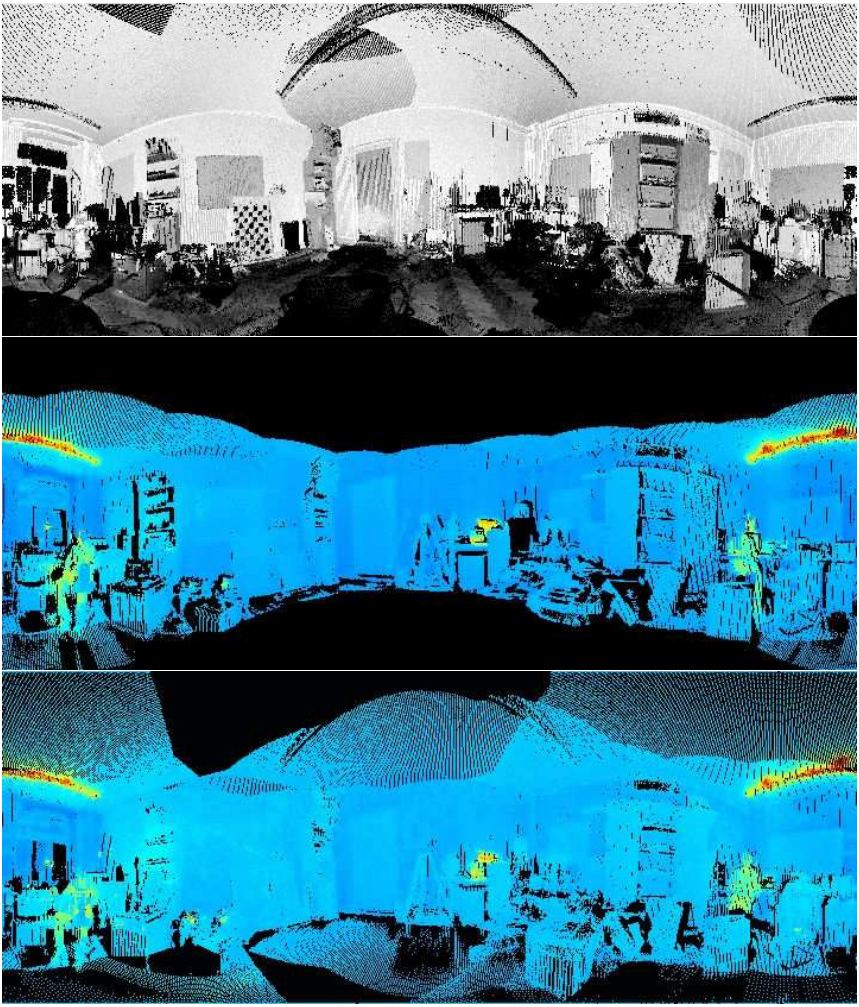


Fig. 6.8: Panorama images of room 1. Top: The entire room with reflectance values as captured by the laser scanner using 3D exploration. Middle: The part captured with thermal information using 2D exploration. Bottom: The part captured with thermal information using 3D exploration. Note that the thermal camera has a drastically reduced field of view as compared to the laser scanner.

that collects the necessary data autonomously to create a 3D thermal model. The system is more adequate for indoor office buildings or for very cluttered environments, in which the room extraction algorithm will find a closed loop in the part of the environment that is not a real room. However, setting a predefined maximal cuboid size supports the exploration of larger rooms. If the currently collected data reaches the predefined maximal cuboid size, the algorithm will switch to the 3D phase using a cuboid part of the environment that does not exceed the allocated memory. The laser scan poses calculated in the 2D map of the exploration module serve as initial poses for the 3D mapping explained in the previous chapter. The next chapter describes the processing that creates a 3D thermal model from the autonomously collected data.

Chapter 7

Thermal modeling

Building thermal 3D models of environments has received some attention recently. Ham and Golparvar-Fard model and evaluate thermal models and the energy performance of buildings [97]. For this purpose they co-calibrate a thermal camera with an RGB camera. The color images are used to create a 3D model using SfM. SfM approaches are prone to failure in regions with few features or repeating structures, the density of the resulting model is low and the scale is unknown. Vidas et al. [200, 201] focus on co-calibrating a thermal camera and a Microsoft Kinect. They developed a hand-held system that creates a 3D model of the environment based on registering the Kinect data. This approach yields a dense model but is limited to the accuracy of the Kinect camera and requires a human operator. Laser scanning has the advantage that the resulting dense model has a high geometric accuracy and is not as sensitive to repeating structures and feature-less areas as SfM approaches are. Gonzáles-Aguilera et al. [49] combine the technology of laser scanners and thermal cameras to create models of building exteriors. They extract features from both the thermal images and the projections of the point cloud from the laser scanner and match these to register the data. In indoor environments this approach is prone to errors as only small parts will be visible due to the small opening angles of thermal cameras. In combination with the low resolution of typical thermal cameras images tend to have too few features for a reliable registration. To the best of our knowledge, our system presented in [39] was the first fully autonomous system for modeling using 3D scanning and thermal imaging.

Each sensor sees the world in its own coordinate system. To relate the data perceived by each sensor these coordinate systems have to be related to each other. The previous chapter describes 3D mapping methods that join point clouds into one coordinate system. As a camera projects the perceived information onto the image plane the data is reduced to 2D pixel coordinates. Thus, obtaining the original environment information is not directly possible due to the loss of information. To overcome this limitation the 3D laser scanner serves as a means to recover the lost information. Since the camera is fixed on top of the scanner the relative pose between both sensors does not change. After determining the transformation describing the relative pose by calibration 3D laser data and image data are merged.

The procedure for joining 3D point clouds with thermal and color information consists of four steps that are explained in the following. First, intrinsic calibration determines the geometric properties of the camera as explained in Chapter 2.2. The underlying concept is identical for

thermal and color cameras but the methods differ slightly. Second, the extrinsic calibration calculates the transformation between the coordinate systems of the different sensors. Third, the points are projected onto the images to retrieve thermal and color information. Last, scan matching transforms the enhanced point clouds into a common coordinate system.

7.1 Camera calibration

Combining several hardware components within one system requires knowledge of their geometric properties. This is achieved by geometric calibration. The calibration consists of two parts, namely intrinsic and extrinsic calibration.

Intrinsic calibration for a camera describes the process of determining the internal parameters of the camera that define how a point in world coordinates is transformed into image coordinates, i.e. projected onto the image plane. The standard camera model consists of two parameters for the focal length, two coordinates for the camera center, three radial distortion coefficients and two tangential distortion coefficients. The standard method to determine them is to take images of a pattern with known features. The image coordinates of the features and their known relation in world coordinates are used to formulate a non-linear least squares problem that is solved using the Levenberg-Marquardt algorithm [218].

Extrinsic calibration refers to the process of determining the relative pose (orientation and translation) between two components. The prerequisite for extrinsic calibration is the intrinsic calibration of the devices.

7.1.1 Intrinsic calibration of thermal and color camera

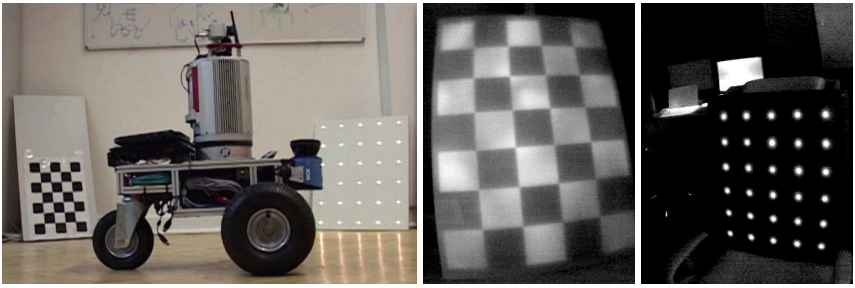
To reference the cameras to the coordinate system of the laser scanner we need the specific properties of both coordinate systems. Each camera has unique parameters that define how a point $\boldsymbol{m} = (x, y, z)^T$ in world coordinates is projected onto the image plane. These parameters are calculated through a process known as geometric camera calibration. Given the focal length (f_x, f_y) of the camera and the camera center (c_x, c_y) based on the pinhole camera model (cf. Section 2.2.3) the image coordinates $\tilde{\boldsymbol{m}} = (\bar{x}, \bar{y})$ are calculated as:

$$\begin{pmatrix} \bar{x} \\ \bar{y} \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}}_{\boldsymbol{A}} \begin{pmatrix} x/z \\ y/z \\ 1 \end{pmatrix}. \tag{7.1}$$

$\boldsymbol{A}$ is called the intrinsic or camera matrix. Given the radial distortion coefficients k_1, k_2, k_3 and the tangential distortion coefficients p_1, p_2 and $r = \sqrt{x^2 + y^2}$ the image points (x_d, y_d) in the distorted image are calculated as

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = \begin{pmatrix} \bar{x}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 \bar{x}\bar{y} + p_2(r^2 + 2\bar{x}^2) \\ \bar{y}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2\bar{y}^2) + 2p_2 \bar{x}\bar{y} \end{pmatrix} \tag{7.2}$$

To determine the parameters of optical cameras chessboard patterns are commonly used because the corners are reliably detectable in the images. A number of images N showing a



(a) (b) (c)

Fig. 7.1: (a) The robot Irma3D in front of two calibration patterns. (b) A chessboard pattern and (c) a light bulb pattern as seen in the thermal image.

chessboard pattern with known number M and size of squares are recorded. Assuming $\mathring{z} = 1$, $\mathring{x}$ and $\mathring{y}$ the coordinates of the pattern features given through their known distance, and the rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$ that relate the pattern to the camera coordinate system the projection is defined up to a scale factor s by

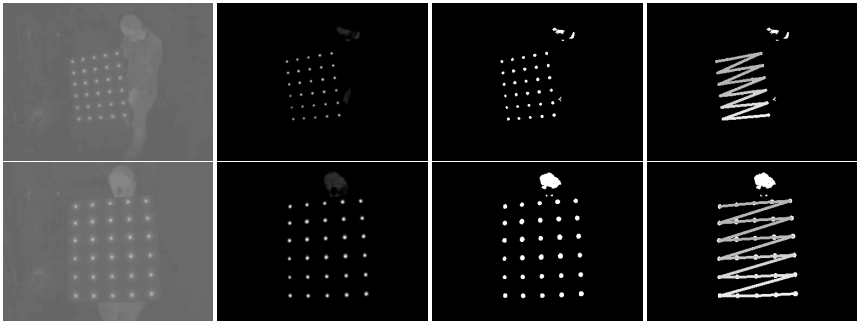
$$s \begin{pmatrix} \bar{x} \\ \bar{y} \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}}_{\boldsymbol{A}} \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix}}_{(\boldsymbol{R},\boldsymbol{t})} \begin{pmatrix} \mathring{x} \\ \mathring{y} \\ \mathring{z} \\ 1 \end{pmatrix}. \tag{7.3}$$

Detecting the internal corners in each image allows to formulate equations (7.3) and (7.2) as a non-linear least squares problem and to solve for the calibration parameters [41, 218]:

$$\sum_{i=1}^{N} \sum_{j=1}^{M} ||\bar{\boldsymbol{m}}_{ij} - \hat{\boldsymbol{m}}(\boldsymbol{A}, \boldsymbol{D}, \boldsymbol{R}_i, \boldsymbol{t}_i, \mathring{\boldsymbol{m}}_j)||^2. \tag{7.4}$$

$\bar{\boldsymbol{m}}_{ij}$ are the detected coordinates of corner j in image i. $\hat{\boldsymbol{m}}(\boldsymbol{A}, \boldsymbol{D}, \boldsymbol{R}_i, \boldsymbol{t}_i, \boldsymbol{m}_j)$ is the projection function of point $\mathring{\boldsymbol{m}}_j$ in image i, according to equation (7.3) and (7.2) using the camera matrix $\boldsymbol{A}$ and the distortion coefficients $\boldsymbol{D}$. The rotation matrix $\boldsymbol{R}_i$ and the translation vector $\boldsymbol{t}_i$ describe the position and orientation of image i in relation to the camera.

For low resolution thermal cameras detection of a chessboard pattern is error-prone even after heating it with an infrared lamp (Fig. 7.1(b)). For pixels that cover the edge of the squares the temperature is averaged over the black and white parts thus blurring the edges. Luhmann et al. [131] explored calibration procedures using different types of thermal cameras. Generally an object with a unique pattern having distinct targets is used which eases labeling and increases accuracy of the calibration process. The points are actively or passively heated. In case of passive heating different materials cause the pattern to show up. Luhmann et al. developed a pattern



(a) -20-100°C (b) $30 - 60$°C (c) > 32.54°C (d) Detected features

Fig. 7.2: Gray scale thermal images for the detection of the calibration pattern.

consisting of targets of self-adhesive foil on an aluminum plate. While the targets emit radiation related to their own temperature the reflective metal surface reflects the cold temperature of space thus leading to a strong contrast in temperature. Unfortunately this concept is not applicable for the co-calibration of a thermal camera and a laser scanner as it is very difficult to position the board in a way that the sky is reflected without occlusions and the board is completely visible in the laser scan. Instead, we suggest a pattern with clearly defined heat sources such as small light bulbs that shows up clearly in thermal images (Fig. 7.1(c)).

Fig. 7.1 shows our pattern in the background. It is composed of 30 tiny 12 Volt lamps, each with a glass-bulb diameter of 4 mm. The overall size of the board is 500 mm (width) × 570 mm (height). Identifying the heat sources in the image enables us to perform intrinsic calibration in the same way as for optical cameras. The approach is similar to the approach used by Ham and Golparvar-Fard [97]. The main difference comes from the ability of the Optris PI160 and Optris PI400 thermal cameras to output the raw temperature information for each pixel rather than providing a color coded image only. While the color settings have to be carefully tuned to even allow for manual detection of the light bulbs, the raw temperature information can easily be used to detect the calibration pattern automatically in the data.

For image processing the temperature data is transformed into a gray scale image. Fig. 7.2 shows the process of the transformation on two example images. Fig. 7.2(a) depicts a gray scale image covering the entire measurement range of the camera. In a first step the measurement range is cropped to the approximate temperature range of the light bulbs (Fig. 7.2(b)). In the next step the image is transformed into a binary image with a threshold (Fig. 7.2(c)). The thresholds have been empirically chosen and are adjustable for different calibration patterns. Fig. 7.3 shows the temperature trend of the 30 light bulbs over a period of 11 min. After less than a minute each of them surpasses the threshold for the binary image. In the binary image connected white areas are detected. To refine the position of the potential light bulb the position is determined by averaging over the gray values from the previous image. Given these refined positions the blobs are sorted and those belonging to the calibration pattern identified using the

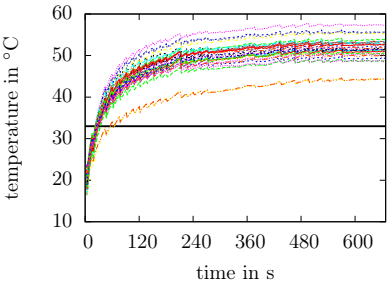


Fig. 7.3: Temperature trend of the 30 light bulbs over a period of 11 min.

approach illustrated in Fig. 7.4. First, the convex hull $P \subset F$ of the feature set F is determined using the Jarvis' March convex hull algorithm [164]. Then the algorithm seeks to find a corner of the calibration pattern from the convex hull. The potential corner point is marked as a star in Fig. 7.4. Starting from any point $\boldsymbol{p}_i \in P$ in the convex hull the three closest neighbors $\boldsymbol{n}_0, \boldsymbol{n}_1, \boldsymbol{n}_2$ from all features in F are sought. Those two points $\boldsymbol{n}_j$ and $\boldsymbol{n}_k$ that enclose the largest angle with $\boldsymbol{p}_i$ are the hypothetical next features in the first column of the pattern and in the first row, respectively. Let $\boldsymbol{v}_j = \boldsymbol{n}_j - \boldsymbol{p}_i$ be the vector from the starting point to its neighbor the next potential feature is expected within a search radius of $\boldsymbol{n}_{\text{pot}} = \boldsymbol{p}_i + 2 \cdot \boldsymbol{v}_j$. If a feature point is found, the search is continued, using the vector between the current point and its predecessor. This is depicted in Fig. 7.4(b) and Fig. 7.4(f)-(i). If no feature point is found in a search region and neither the number of rows nor the number of columns in the feature pattern is reached, the search is continued with a new starting point (Fig. 7.4(f)-(h)). Once all points for the first row and column are found the process continues to find the remaining features by attempting to find the missing point of a rhombus as sketched in Fig. 7.4(c). Fig. 7.4(d) and (j) show the correctly detected calibration patterns with the starting point and the search region used to find each of the points. In a last step the orientation of the pattern is determined and the features are sorted so that they start in the upper left corner as seen in Fig. 7.2 (d).

7.1.2 Extrinsic calibration – cameras and laser scanner

After calculating the internal parameters of the cameras we need to align the camera images with the scanner coordinate system, i.e., extrinsic calibration. The three rotation and three translation parameters are known as the extrinsic camera parameters and define the geometric relation between camera and laser scanner. The mathematical formulation for this problem is identical to determining the intrinsic calibration parameters. Equation (7.3) describes how a 3D point is transformed into image coordinates up to a scale factor s. Suppose there are N pairs of images and point clouds collected and M distinct feature points in each pair. Let $\tilde{\boldsymbol{m}}_j = (\hat{x}_j, \hat{y}_j, \hat{z}_j)$ be a 3D point in the scanner's own coordinate system or the coordinate system of the device used to capture the point cloud and $\tilde{\boldsymbol{m}}_j$ the image coordinates corresponding to this

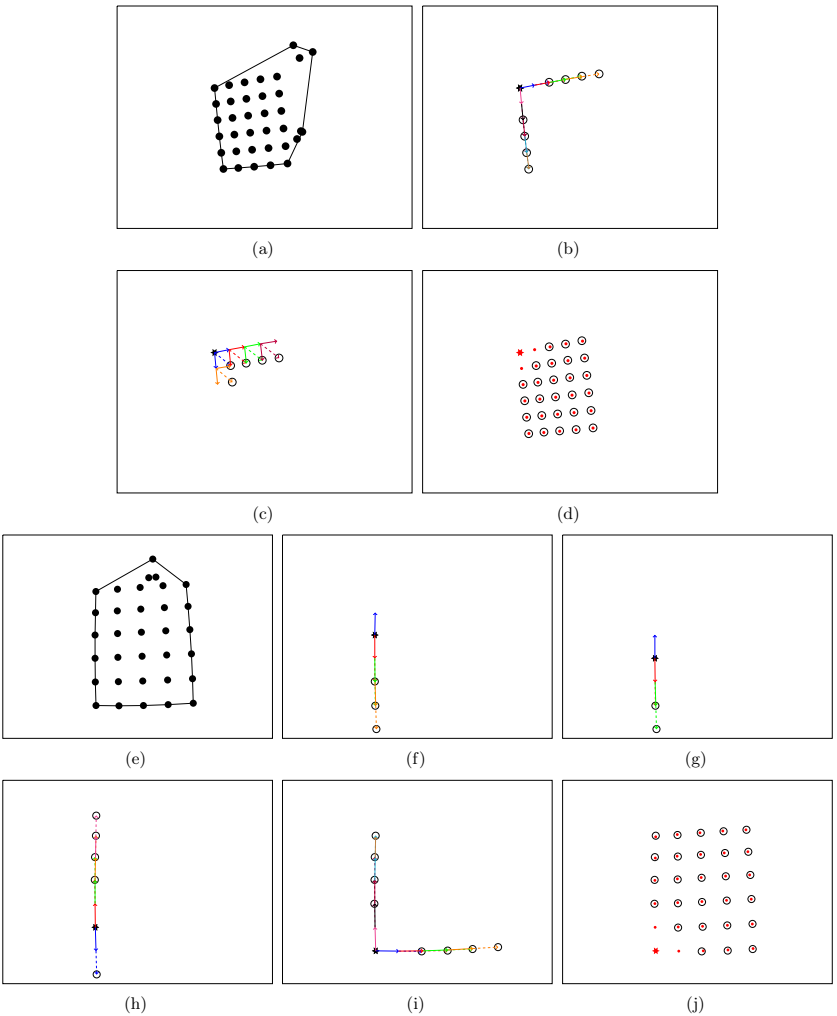


Fig. 7.4: Sorting of the blobs from the calibration pattern, detected in Fig. 7.2. (a) and (e) the convex hull. (b), (f), (g), (h), (i): Trying to find a corner point (starting points marked with ⋆ and extending the search to the first row and column. (c): expanding from the first row and column to the remaining points. Solid arrows show the vector between actual features while dashed arrows show the extension to the search region, marked with a circle. (d) and (j): The final result with the starting point ⋆ and the search areas marked as circles.

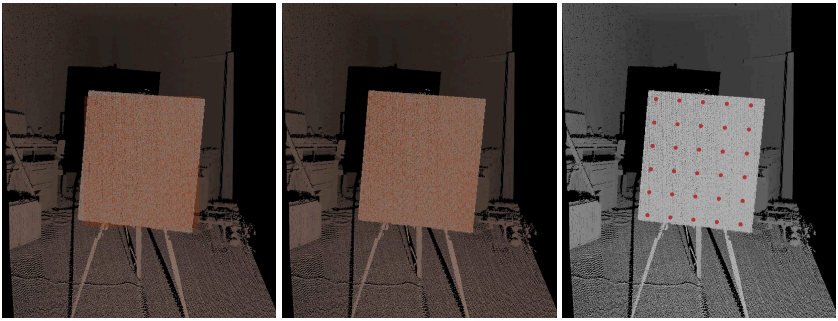


Fig. 7.5: Determining the position of the calibration pattern.

point. Considering the distortions in the image as independent and identically distributed noise, then the maximum likelihood estimate of the transformation between the scanner and camera coordinate system is obtained by minimizing the reprojection error

$$\sum_{j=1}^{M} ||\tilde{\boldsymbol{m}}_j - \hat{\boldsymbol{p}}(\boldsymbol{A}, \boldsymbol{D}, \boldsymbol{R}_i, \boldsymbol{t}_i, \mathring{\boldsymbol{m}}_j)||^2 \tag{7.5}$$

where $\boldsymbol{R}_i$ is the rotation matrix and $\boldsymbol{t}_i$ the translation vector of the ith image. $\boldsymbol{A}$ is the intrinsic matrix and $\boldsymbol{D}$ contains the distortion parameters as calculated in the intrinsic camera calibration. $\hat{\boldsymbol{p}}(\boldsymbol{A}, \boldsymbol{D}, \boldsymbol{R}_i, \boldsymbol{t}_i, \mathring{\boldsymbol{m}}_j)$ defines the projection of point $\mathring{\boldsymbol{m}}_j$ in image i, according to equation (7.3) and (7.2). The extrinsic calibration is achieved by first finding the best transformation for each pair of image and laser scan given the detected features in image and laser scan and the intrinsic calibration of the camera. For best calibration results the transformation has to be chosen that minimizes the reprojection error for all image pairs. This is evaluated in Section 7.2.3.

The extrinsic calibration approach assumes to have a number of points that are identifiable in both the laser scan and the image. For this purpose we attach the calibration pattern onto a board. For the optical camera this is a printed chessboard pattern and for the thermal camera light bulbs arranged in a regular grid pattern. Two examples of the calibration patterns are depicted in the background of Fig. 7.1.

The positions of the points in these patterns are known. Algorithm 13 gives the general idea of detecting the points in a laser scan. To improve speed and robustness it is advisable to first remove all points that are outside the area of interest. Options for this are for example a simple range filter or a box filter, i.e. an axis aligned box is defined and all points outside this box are discarded. For step 2 the different plane detection methods described in chapter 4.3 can be used with some modifications. In the following three different options are described and evaluated.

The RANSAC algorithm [72] is a simple but nonetheless effective algorithm for plane detection. Its easy implementation makes it an ideal candidate for this task, especially since the problem at hand is very specific and asks for the detection of only one plane, given that the

Algorithm 13 Calibration pattern detection in a laser scan.

Require: point cloud, specification of calibration pattern
1: discard points outside the area of the expected board
2: find the plane of the board
3: project a generated plane model into the center of the detected plane
4: use ICP (Iterative Closest Point) algorithm (cf. section 5.1) to fit the plane model to the data points
5: **if** each point from the plane model has a corresponding point in the point cloud **then**
6: **return** position of the light bulbs according to ICP result
7: **end if**

environment is sufficiently prepared beforehand. To facilitate the detection of the calibration board in the point cloud data and to enable the easy positioning at different locations, the board is mounted on a tripod. This way the board has little connection to the ground. It is moved to different positions and at each position one scan and one image are recorded. After removing the floor, the ceiling and most objects behind the board from the point cloud with a thresholding technique, the board becomes the most prominent plane in the data and is detected using the implementation of the RANSAC algorithm described in section 4.3.3. Due to the long acquisition time for the laser scan, the operator can easily move out of the data. Granted an open space for the collection of the calibration and the leveling of the laser scanner this method works reliably.

However, for a setup as used for the two data sets in the Würzburg Residence Palace (cf. Section 3.5), where the scanner was mounted on the robot with a significant incline to enable the view of the ceiling, this method is bound to fail. The floor is not easily removed with a simple thresholding technique. The same holds true for narrow corridors, where the orientation of the scanner, i.e., the scanner's own coordinate system, is not aligned to the walls. An alternative solution detects a predefined number of planes. As the octree implementation does not allow for easy removal of points, the RANSAC solution currently only allows finding a single plane. Thus the Hough Transform is used here. The APHT seems ideally suited for this task. However, it works best when finding an exact number of planes or a number of prominent planes. During the calibration pattern detection there are usually some prominent planes, like the floor, the ceiling and walls. If there are other planar structures in the environment that are more prominent than the calibration pattern, they surpass it in some scans but might be occluded in others. This makes it almost impossible to find suitable parameters for the stopping rule that work for all scans in a calibration data set. Due to the low runtime and the better performance with respect to the removal of duplicate planes the RHT is chosen. Depending on the environment the number of planes to be removed is chosen including the floor, ceiling and walls which cannot be removed with the thresholding technique. The remaining planes up to a maximum number are evaluated in the next step.

The thresholds have to be carefully chosen for each calibration environment, a fact that prevents a fully automatic calibration process. Additionally, for setups that use a 3D camera instead of a laser scanner it is desirable to speed up the process by not forcing the person moving the calibration pattern to move out of the data. For this purpose a region growing approach is introduced that only relies on the dimensions of the calibration pattern. As the

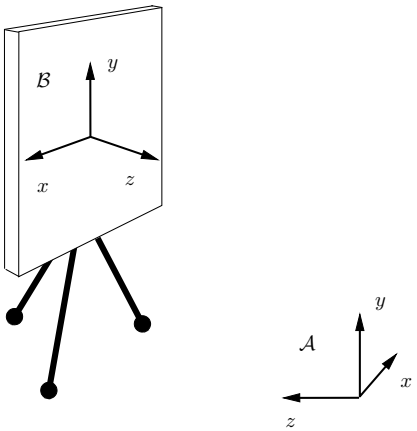


Fig. 7.6: Coordinate system transform.

calibration pattern cannot be occluded in the data and is placed in open space one benefits from the main advantage of region growing approaches in this context, the fact that they consider the connectivity of the points. Once the points in the equirectangular panorama image with a resolution of 2° are segmented into planar regions, some filters are applied to remove regions. For the detection of the calibration pattern the filters are chosen to consider the extent in the direction of each coordinate axis. This extent is calculated by determining the distance between the minimum and maximum value of each coordinate, x_{dist}, y_{dist} and z_{dist}. If the height y_{dist} of a patch exceeds 150 cm or any of the other extents x_{dist} or y_{dist} exceeds 120 cm, the region is discarded. Furthermore, if all extents x_{dist}, y_{dist} and z_{dist} are less than 50 cm or the squared sum of the extents $x_{\text{dist}}^2 + y_{\text{dist}}^2 + z_{\text{dist}}^2$ is below $2500\,\text{cm}^2$ the region is discarded.

Once the plane is detected with any of the plane detection methods, the precise position of the calibration pattern needs to be determined. Let $\boldsymbol{n} = (n_x, n_y, n_z)$ be the normal vector returned by the plane detection algorithm and $\boldsymbol{c} = (c_x, c_y, c_z)$ the mean of the points contributing to the result of the plane detection. A plane model is generated by subsampling points on a plane with the dimensions of the calibration board. This plane model is generated by uniformly distributing points in $M = \{(m_x, m_y, m_z) \in \mathbb{R}^3 | -w/2 \leq m_x \leq w/2, -h/2 \leq m_y \leq h/2, m_z = 0\}$ within the extent $w \times h$ of the calibration pattern. The board is generated on the xy-plane, thus the normal vector is given by $\boldsymbol{n}_m = (0, 0, 1)$. To calculate the transformation for the model so that it is located in the center of the detected plane facing in the same direction as the plane is achieved by change of basis. Let $\mathcal{A}$ be the coordinate system defined by the scanner, i.e., the Cartesian

coordinate system with base

$$\boldsymbol{A}_{\text{kart}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{7.6}$$

that is located at the scan position. The second coordinate system $\mathcal{B}$ is constructed on the detected plane as illustrated in Fig. 7.6. The origin $\boldsymbol{b}_o = (c_x, c_y, c_z)^T$ is the center of the plane. Since the plane model lies on the xy-plane, the third basis vector is the normal vector $\boldsymbol{b}_z = (n_x, n_y, n_z)^T$. Determining the exact roll angle of the board is impractical due to noise in the data. Thus, as an estimate, the coordinate system is assumed to be facing upward. The second basis vector $\boldsymbol{b}_y$ is calculated as the unit vector on the plane in positive y-direction, as the projection of :

$$\boldsymbol{b}_y = \frac{(0 \quad 1 \quad 0)^T - \boldsymbol{b}_z \cdot \rho_y}{\|(0 \quad 1 \quad 0)^T - \boldsymbol{b}_z \cdot \rho_y\|} = \begin{pmatrix} -n_x\rho_y \\ 1 - n_y\rho_y \\ -n_z\rho_y \end{pmatrix} \cdot \frac{1}{\sqrt{n_x^2\rho_y^2 + (1 - n_y\rho_y)^2 + n_z^2\rho_y^2}}$$

with

$$\rho_y = c_x n_x - (c_y + 1) \cdot n_y + c_z n_z + \rho.$$

The first basis vector $\boldsymbol{b}_x$ has to be perpendicular to $\boldsymbol{b}_y$ and $\boldsymbol{b}_z$ and is thus calculated as the cross product

$$\boldsymbol{b}_x = \frac{\boldsymbol{b}_y \times \boldsymbol{b}_z}{\|\boldsymbol{b}_y \times \boldsymbol{b}_z\|}.$$

The rotation for describing a point $\boldsymbol{m}_i$ as seen from coordinate system $\mathcal{B}$ in the coordinate system of $\mathcal{A}$ is given by the homogeneous transformation matrix:

$$\boldsymbol{R}_{\mathcal{B}\to\mathcal{A}} = \begin{pmatrix} \boldsymbol{b}_x & \boldsymbol{b}_y & \boldsymbol{b}_z & \boldsymbol{b}_o \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Therefore the global points are obtained by transforming all points $\boldsymbol{m}_i \in M$ by $\boldsymbol{R}_{\mathcal{B}\to\mathcal{A}} \cdot \boldsymbol{m}_i$. These global points are the start estimate for the position of the calibration pattern. To fit the model perfectly to the data the ICP algorithm (cf. Section 5.1) is used, thus giving the exact pose (position and orientation) of the calibration board. The board detection is rated successful if after convergence every point from M has a corresponding point in the scan, within a range derived from the point density of the point cloud. Since the positions of the light bulbs or chessboard corners on the board are known, their exact position in 3D space is directly calculated from the final pose of the board.

7.1.3 3D to 2D projection and color mapping

During the data acquisition phase laser scans and images are acquired simultaneously. After determining the relations between scanner and cameras in the calibration step this relation is used directly to assign temperature and color values to the point cloud. Due to the different fields of view of the sensors, they each perceive slightly different parts of the world. A region that is visible from one sensor might be occluded for the other sensor. When mapping the color and temperature information to the point cloud this causes wrong correspondences and therefore faulty assigned values. This effect is intensified by low resolution cameras. With only 160×120 pixels per image for the Optris PI160 Imager used during the first experiments, each pixel corresponds to many 3D points seen by the laser scanner leading to errors at edges. Consequently small calibration inaccuracies have a large impact on the results. To solve this problem initially a fast procedure was implemented. All points that were projected onto one pixel and its neighboring pixels were clustered depending on their distance to the scanner. Assuming that most points fall onto the correct pixel a heuristic based on distance to the 3D scanner and size of the cluster determines which points are considered and enhanced with thermal information [24]. The method works sufficiently for the low resolution of the thermal camera but fails for higher resolution cameras. Therefore a ray tracing procedure was implemented that checks whether a point in the point cloud can be seen by the camera. This procedure connects the point $\boldsymbol{p}$ and the camera position $\boldsymbol{c}$ with a straight line $\overline{\boldsymbol{pc}}$ and selects all points with a distance less than a threshold t to $\overline{\boldsymbol{pc}}$, i.e., all points $\boldsymbol{o}_i$ for which

$$|\boldsymbol{p} - \boldsymbol{o}_i|^2 - \frac{|(\boldsymbol{p} - \boldsymbol{o}_i) \cdot (\boldsymbol{p} - \boldsymbol{c})|^2}{|\boldsymbol{p} - \boldsymbol{c}|^2} < t^2 \tag{7.7}$$

holds true. If any point $\boldsymbol{o}_i$ lies between $\boldsymbol{p}$ and $\boldsymbol{c}$, $\boldsymbol{p}$ is not visible from the camera and is therefore discarded. The threshold t accounts for small inaccuracies in the calibration. To speed up the checking procedure the points are organized in a kD-tree data structure (cf. Algorithm 4). With a quick check those voxels are immediately discarded that are not traversed by the ray and therefore all the points within are ignored.

7.2 Evaluation of the calibration methods

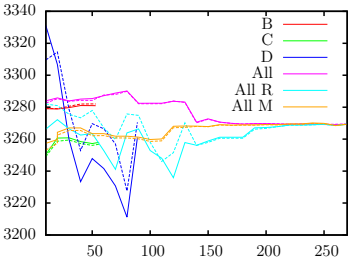
The calibration consists of several steps that are evaluated individually in this section. Several aspects need to be taken into consideration for the calibration procedure. For a portable system it is difficult to design a mount that keeps the camera in the exact same position, thus the calibration needs to be performed for each data acquisition again. Therefore a portable pattern is desired together with a calibration procedure that requires only little manual interaction.

7.2.1 Intrinsic calibration

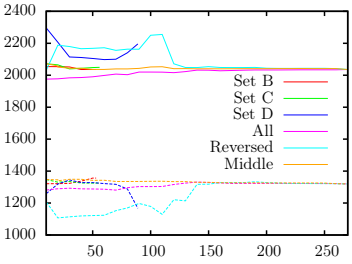
Intrinsic camera calibration is a well-researched problem in photogrammetry and Computer Vision and various methods as well as various calibration patterns exist to determine the intrinsic camera parameters. The established method in Computer Vision is to use planar patterns with detectable features in a predefined arrangement while in Photogrammetry 3D test fields are

often used consisting of an arrangement of identifiable 3D points or lines [76, 127, 183]. From studies comparing the different methods [167, 176, 188] and different targets [8, 216] no method has clearly shown to outperform the others but several conclusions have been made. With a large 3D calibration pattern that covers the entire field of view a few images are sufficient to achieve reliable calibration parameters while planar patterns require a larger number of images taken at varying angles [76, 182]. In recent years the AprilTag library has attracted a lot of attention [154]. It uses fiducial markers, similar to QR codes, that are uniquely identifiable in the image. This increases robustness and allows for calibration images where not the entire pattern is seen. As each marker consists of several squares it already contains several features. However, a sufficient resolution is required to detect them reliably. A calibration tool was developed for the AprilTags that guides the user with the goal to reduce the number of calibration images and increase accuracy [168]. Not all of the suggestions on how to get the best calibration results are applicable in the present case. When changing the zoom factor of the lens, the intrinsic calibration changes [127]. This problem could be overcome by using a fixed lens system, however, also the focal point and the aperture as well as changes in temperature and humidity influence the calibration as explained in Section 2.2. Even the internal structure is not guaranteed to be stable [183]. Thus, for a general camera it is recommended to repeat the intrinsic calibration for each data acquisition requiring a portable calibration pattern. It is recommended in literature to use a 3D calibration structure over a planar 2D structure [40, 76, 120, 182, 217]. It is obvious that 3D patterns allow for additional constraints such as planarity or orthogonality. But the consideration of these constraints introduces additional computational complexity. A thorough evaluation is necessary to confirm that the lack of these constraints is not easily compensated by taking more calibration images. A comparison between a large 3D pattern and an A3 sized chessboard pattern gives results favoring the 3D structure [40] but conclusive systematic studies with patterns of comparable size are lacking and calibration with planar patterns has been shown to give good results in literature.

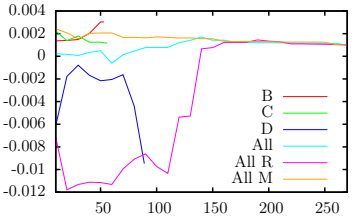
A stable yet portable 3D pattern is much harder to accomplish than the flat pattern used here. Additionally it is harder to mount on a tripod for the extrinsic calibration. In the calibration images the entire field of view should be covered by the calibration pattern. This is easily accomplished with a large pattern at close distance to the camera. This poses the problem of having the calibration pattern in focus at short distances and the object to be surveyed in focus during the data acquisition. This is especially relevant for outdoor scenarios. Therefore the evaluation here will focus on a calibration series from the HELGOLAND data set. With an aperture set to $f/4$ the hyperfocal distance is 4.26 m according to Table 2.3. This gives a near focal distance of more than 2 m according to Fig. 2.19. Due to the large distance between the scanning positions and the *Lange Anna* it might even be necessary to chose a focal point further away than the hyperfocal distance. It is obvious that several images are necessary to cover the entire field of view with the calibration pattern in focus. This also makes the use of more complicated calibration features challenging. However, in future work the use of AprilTags in this scenario will be considered. Four sets (A, B, C, D) of 99 images were acquired. Images where the corners were not reliably detected were removed leading to a total of 270 images. For the evaluation 10 images are added sequentially to the minimization and the trend of the intrinsic calibration parameters was plotted in Fig. 7.7. Besides the individual sets the whole data set was evaluated, once starting from the beginning (All), once starting from the end (All R) and



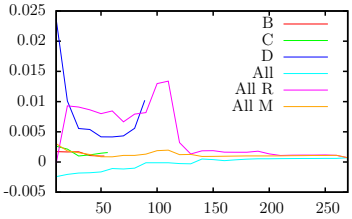
(a) Focal length f_x (solid), f_y (dashed)



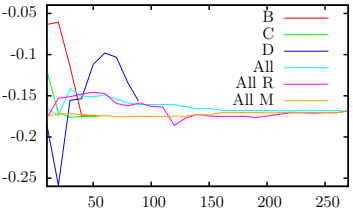
(b) Principal point c_x (solid), c_y (dashed)



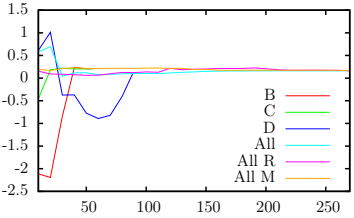
(c) Distortion coefficient p_1



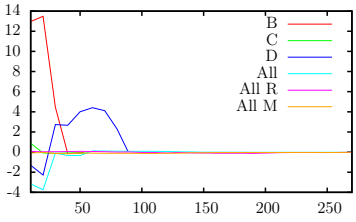
(d) Distortion coefficient p_2



(e) Distortion coefficient k_1

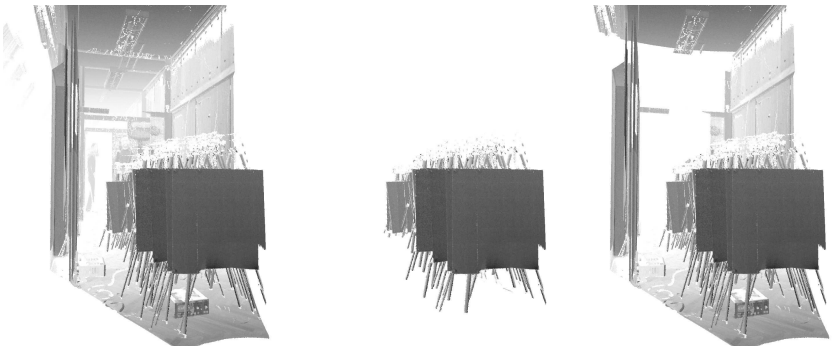


(f) Distortion coefficient k_2



(g) Distortion coefficient k_3

Fig. 7.7: Trend of intrinsic camera calibration using data acquired in Helgoland. Four sets (A, B, C, D) of 99 images each were acquired. Images where the corners were not reliably detected were removed leading to a total of 270 images. The evaluation was carried out by adding 10 images sequentially to the minimization. The individual sets, and the whole data set was evaluated, once starting from the beginning (All), once starting from the end (All R) and once starting from the middle (All M) and adding images on both sides. Note that set A is not plotted due to its equality to the beginning of All.



(a) Full data (b) Box filter (c) Range filter

Fig. 7.8: The 55 calibration patterns from the ZAGREB data set. (a) The complete data set; (b) the data cropped with a box filter of $450 \times 130 \times 147$ cm to remove everything except the calibration patterns; (c) the data cropped at a range of 5 m.

once starting from the middle (All M) and adding images on both sides. The change in the calibration parameters is significant for each of the four data sets. Only after 100 to 150 images the change is less dominant and the parameters appear to converge to a certain value. It should be noted here, that the impact of adding ten images decreases the more images are already in the data set. Nevertheless the trend gives a hint to the necessity of more than 100 images for a stable calibration in this scenario.

7.2.2 Calibration pattern detection in point clouds

To evaluate the different plane detection methods for the calibration pattern three data sets are chosen. The ZAGREB data set is evaluated first. The 70 scans depicted in Fig. 7.8 have a FOV of $50° \times 100°$ and a resolution of $0.08°$ resulting in a maximum possible number of $781,250$ points and the actual average number of $531,366$ points. The calibration data was recorded in a long and narrow hallway with a width of approximately 1.90 m. The scanner was manually positioned so that its coordinate axes aligned with the corridor. For the RANSAC method an axis aligned box of $450 \times 130 \times 147$ cm is manually defined that reduces the point cloud to the area containing the calibration patterns. 15 scans were discarded as they were taken during the alignment process or intersected clearly with the box. The 55 remaining original point clouds as well as the cropped point clouds are shown in Fig. 7.8. The success rate of the pattern detection was determined by visual inspection and the results are presented in Table 7.1. The RANSAC method finds 52 of the 55 calibration patterns. All three failures are true negatives. In two cases the board is closer than the minimum range of the scanner, the third pattern is cropped by the box. However, the manual step of finding the ideal box took a considerable amount of time.

To overcome this necessity several planes are detected using the Hough Transform. As the

Table 7.1: Calibration pattern detection for ZAGREB data set

	RANSAC	H 5%	H 2%	H 5 m 5%	H 5 m 2%	Region growing
Successes	52	41	45	21	48	53
Failures	3	14	10	34	7	2
True positives	52	39	43	21	48	53
True negatives	3	14	10	34	7	2
False positives	0	2	2	0	0	0
False negatives	0	0	0	0	0	0
Time in min	14.8	32.7	54.4	9.9	16.5	16.3

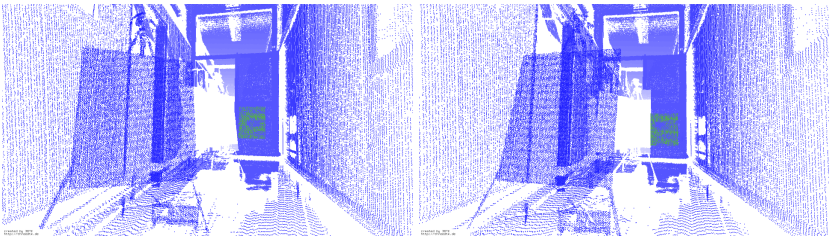


Fig. 7.9: False positives in detecting the calibration pattern. Points on the glass door are measured, leading to an additional plane.

success of the automatic procedure is purely based on point correspondences, it is essential that all planes larger than the calibration pattern are discarded. In the ZAGREB data set this includes the floor, the ceiling, the left and the right wall. The parameters for the RHT are chosen accordingly. The accumulator is discretized with $N_\varphi = 78$, $N_\theta = 180$ and $N_\rho = 100$ and a maximum for $\rho = 5\,\text{m}$. The plane detection terminates if a maximum of ten planes was found or if the point cloud is reduced to $r\%$ of its original size. The algorithm then discards the first 4 detected planes and tries to register the board model to the remaining ones until successful. The results are shown in the third and fourth column of Table 7.1. The number of successes is far less then when using the bounding box. But more important is the number of false positives, i.e., the cases where the algorithm claims a successful calibration but the detected plane is not the calibration pattern. These examples are shown in Fig. 7.9, where points on the glass door are measured and result in an additional plane. Consequently, the calibration pattern detection needs to be manually verified when using this method. The solution is a simple distance criterion. The pattern is always positioned in front of the door, thus a maximum distance of 5 m is introduced, leading to the point clouds depicted in Fig. 7.8(c) where the glass door is removed from the data. The results from applying RHT to this data are shown in columns five and six. The stopping criterion of 5% leads to 34 failures. By removing all points with a distance of more than 5 m a lot of outliers are removed, so that the calibration pattern is often times among the remaining 5% of points. By lowering the threshold the success rate is increased to 48. All failure cases are correctly identified. Some examples are shown in Fig. 7.10.

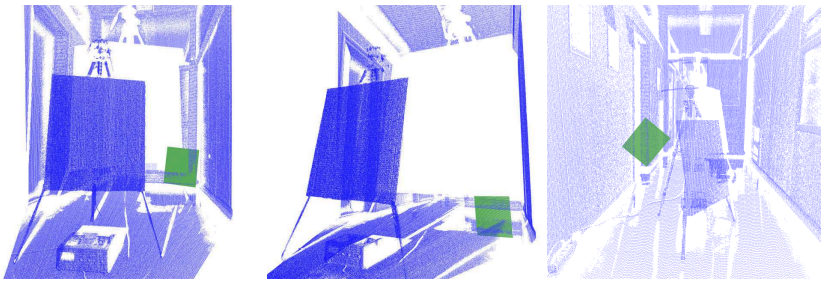
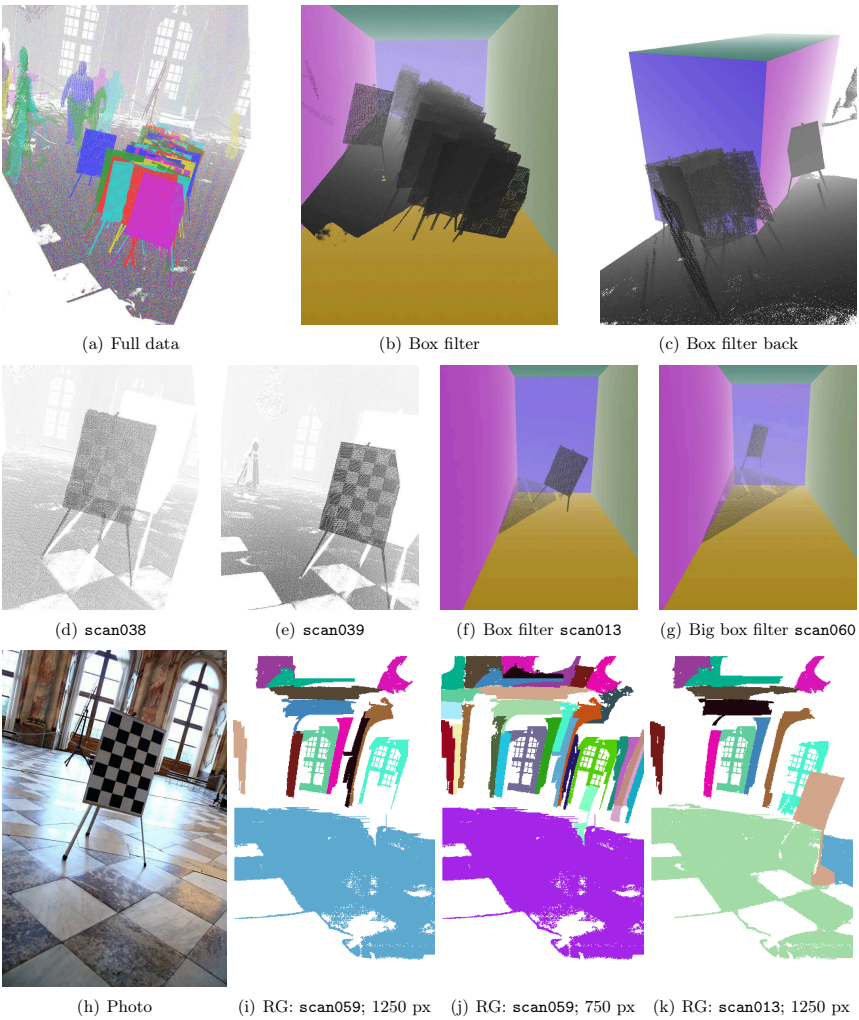


Fig. 7.10: True negatives in detecting the calibration pattern.

(a) (b) (c) (d)

Fig. 7.11: Plane detection in range images.

Even though the failure cases could be reduced to false positives it is desirable to find a method that is less dependent on the correct parameters. For this reason the region growing method was implemented with parameters that depend on the calibration pattern only and not on the scanned environment. Fig. 7.11 shows some examples of the plane detection in the panorama images. The calibration pattern is always correctly segmented. In some cases the tripod legs are partially included in the region of the calibration pattern. In Fig. 7.11(c) the pattern is cropped due to the minimum range of the scanner. The filters reduce the number of planes in question, and especially filter out planes that are too large and could be evaluated as successes when counting the number of point pairs. The effect is shown in Table 7.1. All 53 patterns are found. The two scans where the pattern is not entirely in the scan due to the minimum range of the scanner are also correctly classified as failures.



(a) Full data (b) Box filter (c) Box filter back

(d) `scan038` (e) `scan039` (f) Box filter `scan013` (g) Big box filter `scan060`

(h) Photo (i) RG: `scan059`; 1250 px (j) RG: `scan059`; 750 px (k) RG: `scan013`; 1250 px

Fig. 7.12: Calibration pattern detection in the Imperial Hall of the Würzburg Residence Palace. The full data set (a) and two sample scans (d) and (e). The box filter (b),(c),(f) and a slightly larger box filter (g). The planar patches found by region growing with at least $N_{\min} = 1250$ pixels (i) and (k) and $N_{\min} = 750$ pixels (j), respectively.

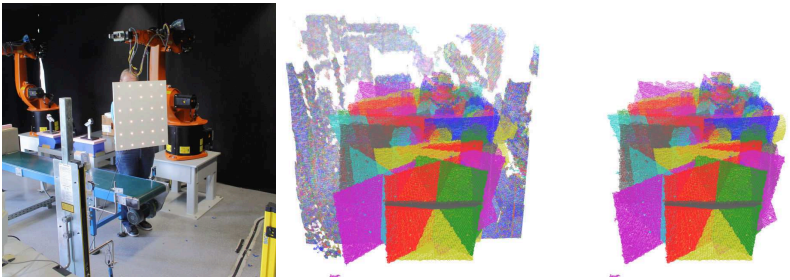
Table 7.2: Calibration pattern detection for RESIDENCE data set

	RANSAC	RANSAC Big	H 5 m 5%	H 5 m 2%	Region growing
Successes	52	56	55	69	70
Failures	20	16	17	3	2
True positives	52	51	55	69	70
True negatives	20	16	16	2	2
False positives	0	5	0	0	0
False negatives	0	0	1	1	0
Time in min	18.4	19.9	11.5	13.2	16.8

The second data set for evaluation is more challenging, the calibration data with the chessboard pattern from the RESIDENCE data set. The scanner was mounted on a tilted platform to allow for the DSLR camera to see more of the paintings on the ceiling. The scans have a FOV of $60^\circ \times 100^\circ$ and a resolution of 0.08° resulting in a maximum possible number of of $937,500$ points and the actual average number of $631,712$ points. Fig. 7.12 shows the chessboards pattern as seen from the camera mounted on top of the robot and the scans. The tilted scans cause obvious problems. This is accentuated with the axis aligned box. No single configuration allows for selecting all patterns without the floor. This is reflected in the results presented in Table 7.2. The box of $138 \times 200 \times 395\,\text{cm}$ (depicted in Fig.s 7.12(b),(c),(f)) was chosen to keep as many of the calibration patterns as possible without having a significant amount of points from the floor in the data. As a consequence the pattern could not be identified in 20 scans. Increasing the box to $150 \times 200 \times 450\,\text{cm}$ (cf. Fig. 7.12(g) and column "RANSAC Big" in Table 7.2) results in an increasing number of false positives because with increasing size of the floor area, the model of the calibration pattern is more likely to fit completely into that area, i.e., in a way that all points from the model have a corresponding point on the floor.

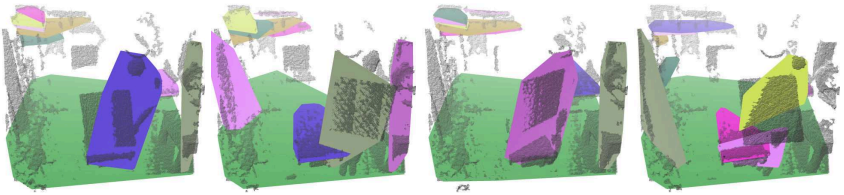
Due to the lack of planar walls and the large amount of open space in the Imperial Hall where the calibration data was collected it is impractical to apply the Hough Transform without using a range filter. Again, aborting the plane detection with only 5 % of the points left leads to many failures (cf. Table 7.2). Adopting the stopping rule to 2 % leads to the detection of 69 patterns. The region growing approach has one more true positive. The two failure case for region growing, `scan039` and `scan059` are depicted in Fig. 7.12(e), (i) and (j). In `scan039` the upper left corner is cropped by the FOV of scanner. `scan059` is at a large distance and in an unfavorable angle to the scanner, so that it falls below the minimum number $N_{\min} = 1250$ of pixels required in the plane detection step. Decreasing the limit to $N_{\min} = 750$ would have solved this problem. For reference a successful calibration pattern detection is shown in Fig. 7.12(k). With both Hough Transform variants as well as RANSAC one detection was marked as a failure even though it was correctly identified as verified by visual inspection. Fig. 7.12(d) shows the scan in question. The upper right corner is very close to the edge of the measurement range leading to a single missing correspondence.

The third data set stems from the PROJECTOR data set. As opposed to the previous data sets this one was not recorded with a laser scanner but with an Asus Xtion RGB-D camera. The fast data acquisition time of 30 Hz allows to collect the calibration data without a tripod.



(a) Photo (b) Full data (c) Box filter

Fig. 7.13: Calibration pattern detection for the PROJECTOR data set. (a) A photo of the scene, (b) the full data set and (c) the data cropped with a box of $135 \times 130 \times 250\,\text{cm}$.



(a) `scan000` success (b) `scan041` success (c) `scan001` failure (d) `scan003` failure

Fig. 7.14: Plane detection using the RHT in the PROJECTOR data set. The clutter in the noise leads to many false detections. The examples show that the correct detection of the plane does not determine the success of the calibration pattern detection as in one of the success cases (b) and in one of the failure cases (c) the found plane is aligned with the calibration board while in the other success case (a) and the other failure case (d) the plane intersects with the board instead.

Instead a person moves a light bulb pattern to different poses. The fast acquisition time comes at the expense of lower accuracy and a smaller field of view of $58° \times 45°$. With a resolution of 640×480 pixels up to $307,200$ points, on average $240,871$ points are captured. The full calibration data set is depicted in Fig. 7.13. The scene is the work cell of a robotic manipulator and full of clutter. The 3D camera is tilted to capture as much of the work area as possible. This makes it again hard to find an axis aligned box for cropping the clutter (cf. Fig. 7.13(c)). Furthermore, the person holding the calibration pattern is often visible in the data. The clutter and the noise of the camera make removing a predefined number of prominent planes unfeasible. Fig. 7.14 shows some results of applying the RHT and demonstrates how unreliably the board is detected in this kind of data. Up to 10 planes are detected. The floor plane is the only

Table 7.3: Calibration pattern detection for the PROJECTOR data set

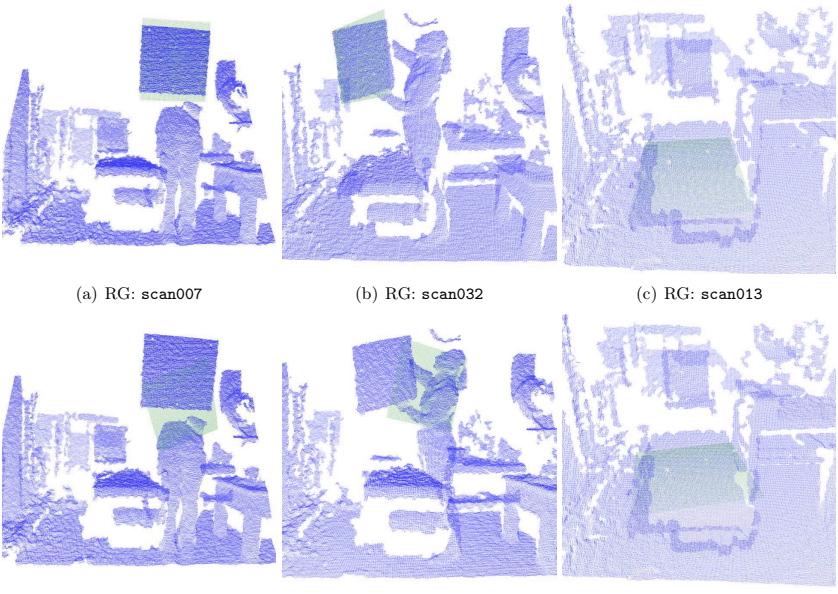
	RANSAC	H 5 m 5%	H 5 m 2%	Region growing
Successes	42	36	32	48
Failures	12	18	22	6
True positives	42	36	32	48
True negatives	11	–	–	4
False positives	0	0	0	0
False negatives	1	–	–	2
Time in min	20.7	60.0	84.9	27.8

prominent plane in all data and is thus removed for the evaluation seen in Table 7.3. In `scan041` and `scan001` the plane containing the board is detected, while in `scan000` and `scan003` a random plane that intersects with the board is detected. However, the fact whether the board lies on the detected plane or intersects with it does not determine the success of the calibration pattern detection. The convex hull includes also the points in the vicinity of the calibration pattern, thus the center of the convex hull determines how close to the calibration pattern the model plane will be projected, which has – together with the density of the falsely included points – a major impact on the success of the calibration process. Thus the distinction between true and false negatives is omitted in Table 7.3. All successes are evaluated as true positives by visual inspection. With a success rate of 59.25% for the 5% variant and 66.67% for the 2% stopping rule the method clearly fails for this setup.

Cropping the data with a box of $135 \times 130 \times 250$ cm and applying RANSAC afterwards to detect the most prominent plane increases the success rate to 77.78% (cf. Table 7.3). The region growing approach, however, achieves once again the best results. The failure cases are visualized in Fig. 7.15 and 7.16. In SCAN007 and SCAN032 the pattern is cropped due to the limited field of view of the camera. Similar to the examples shown for the Hough Transform also with RANSAC the model plane is not centered on the calibration pattern while with the region growing approach the model lies centrally on the calibration pattern. In `scan013` the board lies on the conveyor belt and is hardly distinguishable from it. Due to a small incident angle the edge is very noisy. Here both methods fail. These three cases are marked as true negatives in Table 7.3. In `scan034` many points on the person holding the board and on the conveyor belt are coplanar to the calibration pattern (Fig. 7.16(b)) leading to a large planar patch (Fig. 7.16(a)) that is filtered by the 1250 pixel criterion. The two last failures for region growing are due to noise.

For the RANSAC approach most failure cases are due to the initial position estimate of the board between the person and the board as exemplarily shown in Fig. 7.16(c). Even in cases where the planar patch contains part of the person (cf. Fig. 7.16(e)) the better orientation estimate leads to successes for region growing while RANSAC fails.

The methods have not been optimized for time efficiency as the calibration is a step that is performed once before the actual data collection. The crucial point is the ability to detect the patterns fully automatically. All methods are comparable in the order of magnitude of their time requirements. In the cluttered environment of the PROJECTOR data set the RHT needs

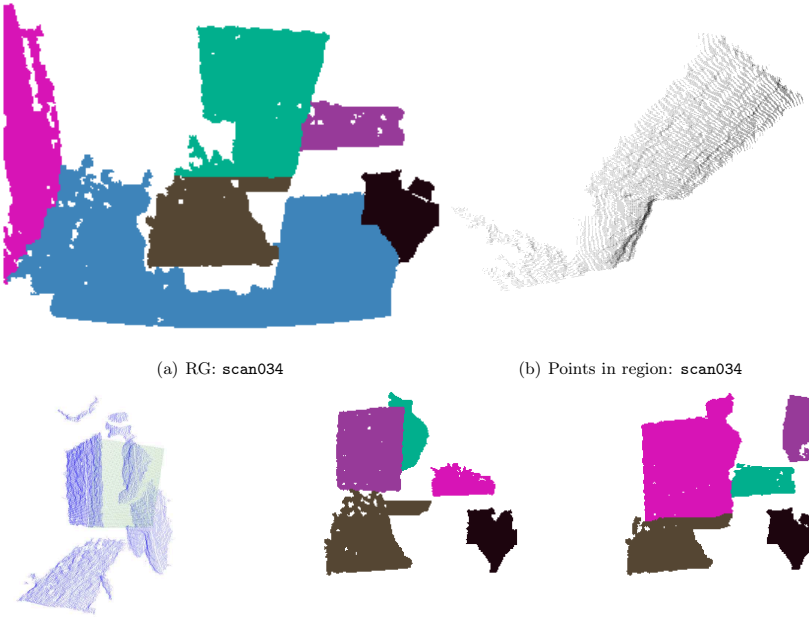


(a) RG: `scan007` (b) RG: `scan032` (c) RG: `scan013`

(d) RANSAC: `scan007` (e) RANSAC: `scan032` (f) RANSAC: `scan013`

Fig. 7.15: Three failure cases in the `Projector` data set. In `scan007` and `scan032` the data is calibration pattern is cropped. In `scan013` it is hardly distinguishable from the conveyor belt and the top is ragged due to the low incident angle.

significantly longer, due to the larger number of hypotheses that need to be evaluated in the refinement step. The single plane RANSAC approach works well in open space environments where a range or box filter removes all obstacles. For more restricted environments the setup of these filters takes up some time and requires the sensor to be correctly aligned with the environment. In cases of dominant planar structures in the environment the parameter search can be reduced by detecting several planes and removing the most dominant planes, keeping only few small candidates as hypotheses that need to be further evaluated. The more planar the environment is, the better this method performs. For the `Residence` data set where the only unwanted data was the smooth floor, the results were almost perfect, while for the `Zagreb` data set the number of successes was significantly lower. For the cluttered data from the work cell of the industrial robot the method failed. However, the most crucial value, the number of false positives remained 0 even in this scenario. The region growing approach performed best on all data sets. Without the need to change the initially set parameters the method performed



(a) RG: `scan034` (b) Points in region: `scan034`

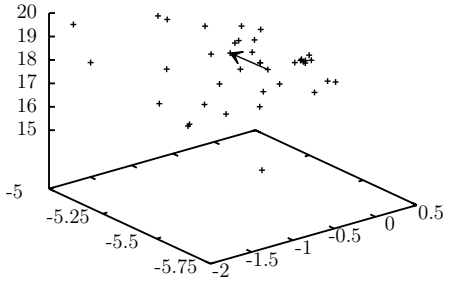
(c) RANSAC: `scan005` (d) RG: `scan005` (e) RG: `scan001`

Fig. 7.16: For `scan034` the RG approach fails because the coplanarity of the points not belonging to the board (b) leads to a planar patch that is larger than 1250 pixels and is thus filtered out (a). (c) shows a failure of the pattern detection with RANSAC while (d) and (e) show two successful cases for RG.

superior to the others on all three data sets. There was not a single false detection and the number of successes was higher than with any of the other methods. Only in the last scenario some false negatives appeared but with the low acquisition time of the calibration data in this scenario the issue is easily overcome by increasing the number of data pairs.

7.2.3 Estimation of the extrinsic calibration parameters

The extrinsic calibration is achieved by first finding the best transformation for each pair of image and laser scan given the detected features in image and laser scan and the intrinsic calibration of the camera. The transformation is defined as the translation t_x, t_y, t_z and the rotation θ_x, θ_y, θ_z. The evaluation in this section is carried out using two instances of calibration data from the Automation Lab data set, first the one from the experiments in [38] and second the one



(a) Translation in cm

Fig. 7.17: Distribution of the translation (in cm) calculated from the sensor data pairs.

from the experiments in [39]. Especially for low resolution images these N transformations are scattered, as shown exemplarily in Fig. 7.17 for the first calibration data set consisting of 37 data pairs. The translation differs by several cm in each coordinate. For best calibration results the transformation has to be chosen that minimizes the reprojection error for all image pairs. The reprojection error r_k of the transformation calculated from data pair k is defined as

$$r_k = \sqrt{\sum_{i=1}^{N}\sum_{j=1}^{M} \frac{||\hat{\boldsymbol{m}}_{i,j,k} - \bar{\boldsymbol{m}}_{i,j}||^2}{M \cdot N}}, \tag{7.8}$$

where N is the number of data pairs, M is the number of feature points on the calibration pattern, $\bar{\boldsymbol{m}}_{i,j}$ is the jth point on the calibration pattern in the ith thermal image and $\hat{\boldsymbol{m}}_{i,j,k}$ is the jth point on the calibration pattern in the ith scan when projected onto the image using the kth transformation.

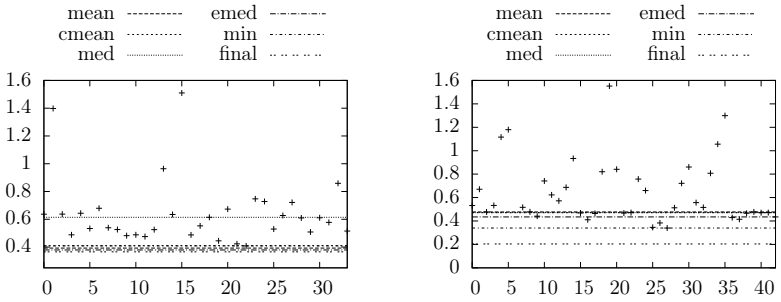
Different methods have been applied to find the best transformation.

mean The average over all translation and rotation vectors is calculated.

emed The transformation is determined as the element-wise median of all transformations, i.e., the median of all values x, and of all values y, z, θ_x,θ_y, θ_z, respectively.

med The transformation is chosen that contains the median of the translation vectors. The median translation vector is the vector with the minimum summed distances to all other vectors.

cmean The transformation vectors are filtered based on the distance of the translation vectors



(a) Calibration 1

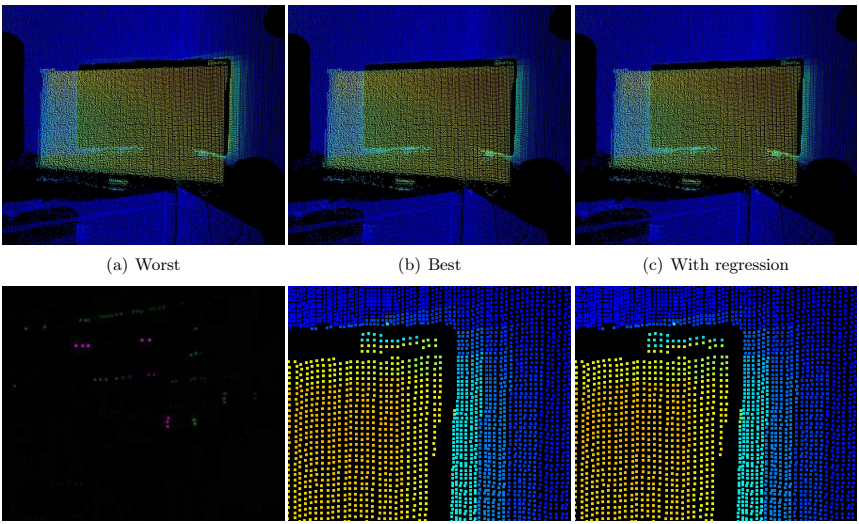
(b) Calibration 2

Fig. 7.18: Reprojection error for extrinsic calibration. The x-axis represents the indices of the data pairs.

to the median. The transformation is then chosen as the mean of the remaining vectors.

min From all transformations the one is chosen, that has the minimum reprojection error. Each translation vector is used to project all feature points onto their corresponding images. The reprojection error for a translation vector is the average distance between feature points in the image and the corresponding projected scan features.

Fig. 7.18 shows the results for the different methods for the two calibration data sets. The reprojection error for each of the N image and scan pairs is plotted as crosses. Here the deviation of each hypothesis becomes obvious that is due to the low resolution of the thermal image. The lines represent the different methods. From only two examples it becomes clear that no single method is always advisable to use. While in the first experiment all methods except the median perform better than all of the individual transformations, in the second data set several of the individual data points give better results than the joint methods. To yield the best calibration results in a general case, it is suggested to compare the single transformations to the joint methods and chose the best one. In [24] we decided against the use of any advanced regression methods for finding the optimal transformation for several reasons. The best results for the used methods lie close together. Therefore it is very likely that they actually represent a very good transformation. Second, the results depend heavily on the quality of the input data. As the input data will always be noisy it is expected that the benefits in terms of quality are not worth the extra effort of a reliable optimization method. To support this claim a regression method was implemented. Starting from the best transformation from the previously described methods, subsequently all transformation parameters are altered with a small value $\pm\varepsilon_t$ for the translation or $\pm\varepsilon_r$ for the rotation. For each transformation parameter first the sign is determined that leads to the largest improvement in reprojection error. Then the parameter is altered by $\pm\varepsilon$ repeatedly until no further improvement is achieved. The same procedure is applied to the



(a) Worst (b) Best (c) With regression

(d) Difference (e) Best (f) With regression

Fig. 7.19: Comparison of calibration using the first data set. Above: a computer monitor in the lab. Below: zoomed in to the top right corner of the monitor.

remaining transformation parameters. Once one parameter has been changed, the others need to be tested again. This procedure is repeated until no further improvement is achieved and a local minimum is reached. The risk of reaching only a local and not a global optimum is reduced by using a large number of data pairs for the calibration and by ensuring a good distribution of the pattern over the measurement range to avoid systematic errors.

Fig. 7.19 shows an example of the projection with different calibration results from the first calibration data set. The left image shows the result using the transformation with the highest reprojection error, the center image the projection using the transformation with the lowest reprojection error from any of the methods. The image on the right is the result with the transformation from the regression. It is obvious that the lower reprojection error also produces visibly better results. The image shows a computer monitor. In the right image the wall behind the right edge of the monitor is assigned warmer temperatures while in the left image the hot spot aligns nicely with the monitor edge in the point cloud. To see the difference between the best results from the regression method, the second row is zoomed in to the upper right corner of the monitor. One more line of warmer points is on the wall when not using the regression. This is highlighted in the difference image. As expected from the plots of the reprojection error (Fig. 7.18), the difference is minor.

For the second data set Fig. 7.20 shows the difference between the highest (top) and lowest

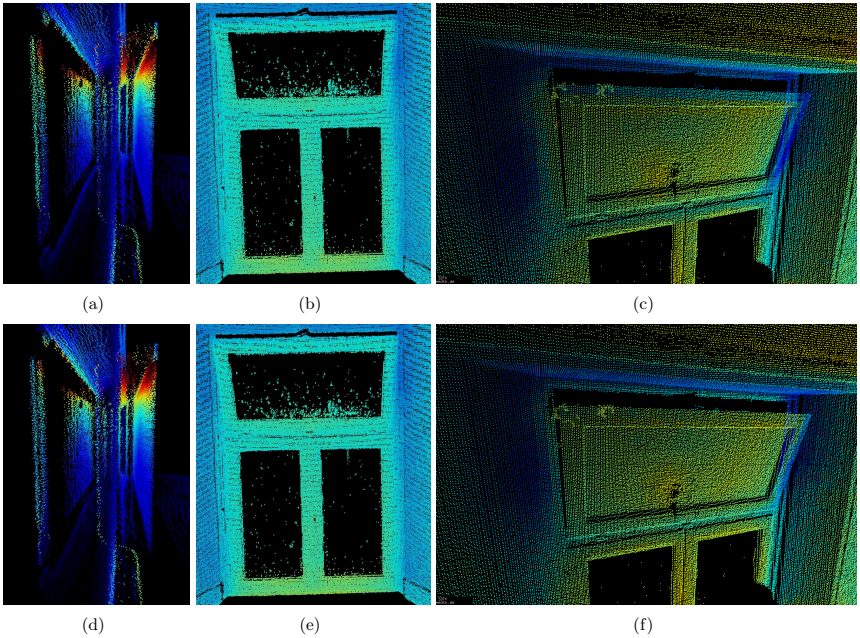
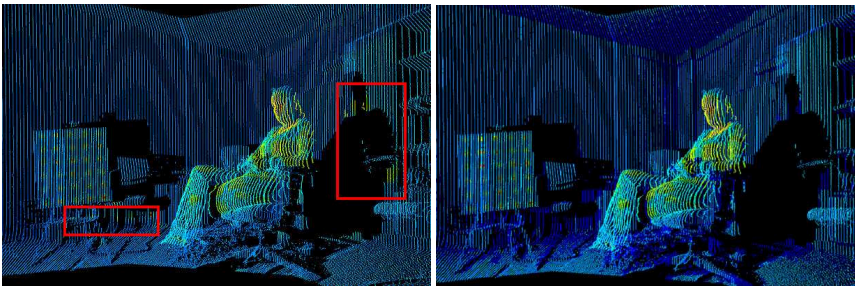


Fig. 7.20: Comparison of calibration using the second data set. Above: projection using transformation with highest reprojection error. Below: using transformation with lowest reprojection error.

(bottom) reprojection error after regression. The left image is a side view of a radiator. In the top image more of the high temperature values are projected onto the points from the wall (left). This suggests that the thermal image is projected slightly too low. This suggestion is verified when looking at two windows. Again, when using the transformation with the highest reprojection error, the thermogram does not align well with the window frames.

7.2.4 Color mapping and occlusion detection

The color or temperature mapping is visually inspected on several data sets. Here a few examples are selected to demonstrate the effectiveness of the occlusion detection methods. The fast method was used on an earlier data set with the low resolution thermal camera, while the raytracing approach is used later on. An exemplary result is shown in Fig. 7.21. The heuristic based on distance to the 3D scanner and size of the cluster determines effectively which points are considered and enhanced with color information. This removes also some correct color information but the improvement prevails. The heat from the calibration pattern that is projected



(a) No correction (b) With correction

Fig. 7.21: Before (a) and after (b) the occlusion correction algorithm.

onto the floor without correction is removed by the procedure as well as the person's body heat that is projected onto the wall without correction.

Approaches commonly used when combining laser scan data and optical images employ expensive high resolution cameras. However, the resolution of the thermal camera was only 120×160 pixels. This suggests that the approach is also applicable for low resolution optical cameras. To verify this we used a low-cost webcam in our experiments and created an image pyramid of the optical images to compare the results achieved with different resolutions, i.e., 1200×1600, 900×1200, 600×800, 300×400, and 120×160. Experiments showed that internal and external calibration could successfully be performed for all resolutions given careful positioning of the calibration pattern during calibration. At distances between 50 and 100 cm the distance was large enough to capture the entire board reliably with the scanner and the pattern was still detectable in the camera images. Results achieved with different resolution images show that the quality decreases and error-proneness increases with decreasing resolution. This effect is diminished by smart application of the error correction algorithm depending on the scenario. Even at resolutions as low as 120×160 an outdoor scene benefits from the added color information. Fig 7.22 presents results of this experiment. The raw point cloud is presented and the point cloud colored with different resolution images. The bottom left corner of each image shows the scaled input images. The robustness of the approach can be improved by using more data pairs for the calibration process. For the lowest resolution, i.e., 120×160, only 21 images could be used for the intrinsic calibration, and 15 data pairs for the extrinsic calibration. Given the higher noise level in the feature detection due to the low resolution it is obvious that more data pairs are necessary to minimize the error. Applying the correction algorithm with varying thresholds for decreasing resolution further improves the projection results.

The effectiveness of the ray tracing procedure is demonstrated with an exemplary scan of the Imperial Hall in Fig. 7.23(a) and (b). Especially behind the tripods and the chandeliers a lot of errors are removed after ray tracing.

Fig. 7.22: From top left to bottom right: (1) 3D Point cloud without color information. (2) Point cloud coloring with using the full 1600×1200 color image. (3)-(6) Image pyramid with the corresponding point cloud coloring. (best viewed in color)

(a) No ray tracing (b) Ray tracing

Fig. 7.23: One scan from the Imperial Hall colored with the information from the photos without (a) and with (b) correction from using ray tracing. It becomes evident that the ray tracing methods removes wrongly colored points.

7.2.5 Exploiting the 3D geometry

One of the main advantages of combining 3D laser scanners with thermal imaging devices is the known geometry of the scene. As illustrated in Fig. 2.23(a) the surface radiation of an object is influenced not only by the object itself but also by the environment. To avoid wrong measurements due to reflections precautions have to be taken during the data acquisition. As illustrated in Fig. 7.24 the angle at which an image is taken plays an important role especially for high-reflective surfaces [74]. As previously mentioned in Section 7.1.1 Luhmann et al. exploit the atmospheric temperature for the calibration of thermal cameras [131]. Their calibration pattern consists of areas with high and areas with low reflectivity. The parts with high reflectivity reflect the temperature of the sky. The same holds true for roofs or other surfaces made of high-reflective materials such as stainless steel. The resulting temperature readings will be highly biased towards the sky temperature which typically ranges between 0 °C and -20 °C. A bias towards the other direction occurs from sun glints when taking measurements during the day. To prevent biased measurements the angle between the optical axis of the camera and the normal vector of the surface should not exceed 60 °. On the other hand, straight-on measurements at close range might cause the camera to see reflections of itself. [74]

When taking measurements the view angles are difficult to approximate. Here the measurements benefit from the known geometry of the 3D laser scans. After generating the thermal point cloud all points are removed that violate the 60 ° rule. First, for each point the 20 nearest neighbors are determined using a kD-tree. By fitting a plane through these neighbors, as described in Section 4.3.1, the surface normal $\boldsymbol{n}$ at the point is estimated. Let $\boldsymbol{p}_n$ be the normalized direction vector between the point and the origin of the local scan coordinate system, then all points with $|\cos^{-1}(\boldsymbol{n} \cdot \boldsymbol{p})| > 60°$ are discarded. The results are shown for the Bremen City data set. As the distance between camera and surfaces is large the straight-on condition was neglected. After cleanup $62,272,650$ points remain of the $81,398,810$ points from the initial thermal point cloud. This means approximately 76.5% of the points were collected within the best view angle. Fig. 7.25 shows the original point cloud (a) with the points marked for removal

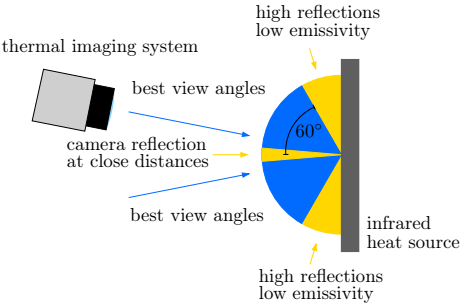
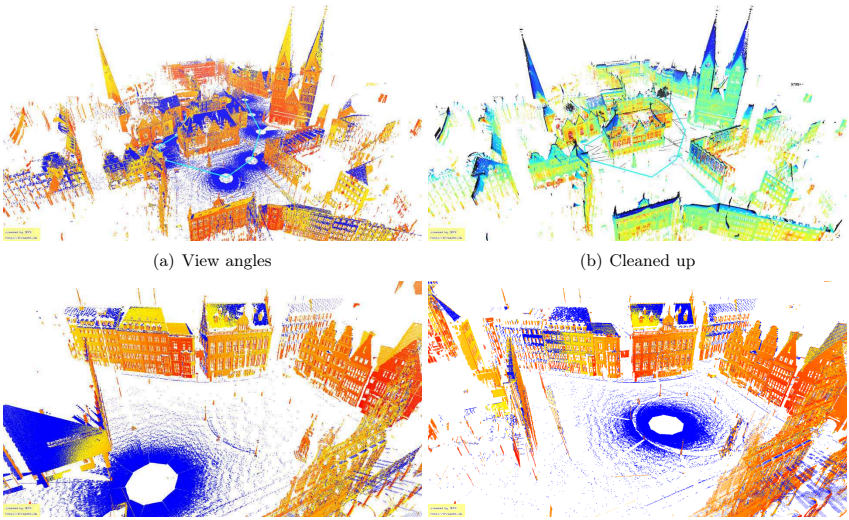


Fig. 7.24: Best view angles for thermography. Straight-on measurements on high reflective materials at close distances cause direct camera reflection, oblique angles are prone to errors due to overall reflection. Reproduced based on [74].



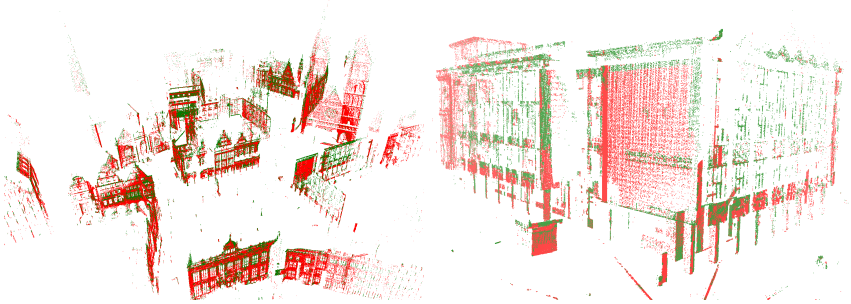
(a) View angles (b) Cleaned up

(c) `scan003` (d) `scan004`

Fig. 7.25: The complete scene from the Bremen City data set. (a) The points colored based on the angle between the normal and the view vector. Blue points have an angle greater than 60°. (b) The scene without the points with an angle greater than 60° colored based on temperature values. (c) + (d) Two scans from the Market square, showing the different view angles.

in blue and the cleaned-up point cloud (b) of the complete scene. As expected, many points on the rooftops and on the floor are removed but also of some other surfaces measurements were taken at unfavorable angles. Especially in narrow passages it is difficult to acquire data at the best angles. The comparison between two consecutive scans (c)+(d) that were collected at a distance of less than 25 m shows the different view angles for the measurements of the same surface.

To validate the measurements and to prevent misinterpretations due to reflections an approach is to take several measurements at changing angles and comparing the temperature values. Naturally this approach is affected by changing temperature conditions, but assuming that measurements were collected at mostly stable conditions the effects should be minor. For this approach the octree data structure (cf. Section 4.1.1) is used as it is designed to store additional information apart from the point coordinates. The procedure determines for each point the nearest neighbor in each of the other scans with a maximum distance of 1 cm. The point is discarded if the temperature differs more than 1 K from the median of the neighbors. Table 7.4 lists the number of points for the original data set and for the best view data set. Regardless of the previous processing step $\approx 57\%$ of the points with neighbors at less than 1 cm are discarded.



(a) Full scene (b) *Haus der Bürgerschaft*

Fig. 7.26: Points with neighbors at a distance of 1 cm from the BREMEN CITY data set. Points with a temperature deviation of > 1 K are marked in green.

Table 7.4: Point removal based on neighboring temperatures

	total # points	# points with neighbors	# points removed	# final points
All points	80,270,636	10,439,942	5,941,710	74,328,926
Best view	61,325,127	8,769,834	5,024,158	56,300,969

Fig. 7.26 shows the results for the full data set. It becomes evident that there are some sparse areas, that have only been seen from few scans or at a large distance leading to a low point density. Some parts of the full point cloud show not at all, meaning that they have only been captured by a single scan. In other areas the points with neighbors are evenly distributed. The points that are removed due to a temperature deviation occur often near edges. Noticeable is the large number of points marked for removal at the *Haus der Bürgerschaft*. While the market square consists mostly of historic buildings, the *Haus der Bürgerschaft* was built in 1966 and is covered with facades made from glass and aluminum causing reflections.

7.3 Experimental results

In the following a few more remarkable scenarios are presented to outline the application areas of the previously presented approach. The main focus of this work is on 3D thermal modeling. In Chapter 6 an example of indoor thermography was presented that incorporated the data collection with an autonomous mobile robot. An image of the final point cloud enhanced with reflectance, color and thermal information is given in Fig. 7.27. The color scale is adjustable for a good view of the temperature distribution in the current data. In all images depicted here, blue corresponds to cold temperatures while dark red corresponds to warm temperature values. Switching between the different views in the viewer from 3DTK enables the user to detect sources of wasted energy and to locate them clearly in the 3D view. One example of outdoor thermography, the BREMEN CITY data set, was used in the previous section to show the benefit

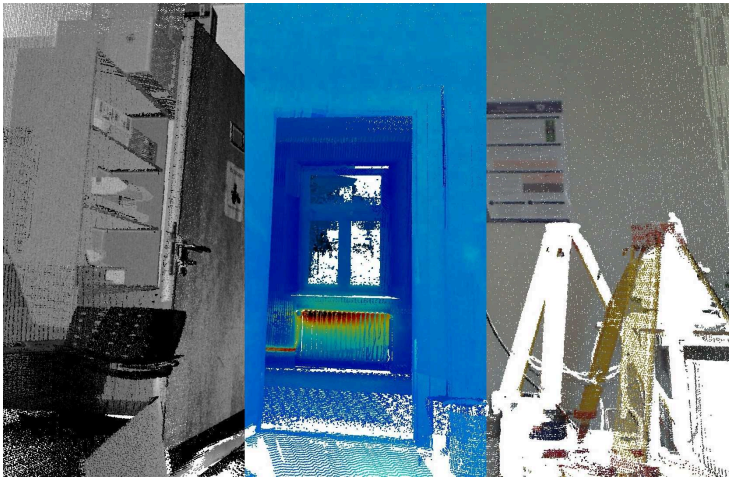


Fig. 7.27: Laser scan with reflectance (left), thermal (middle) and color (right) information.

of the 3D geometry for thermography. Fig. 7.28 shows another example from the JACOBS data set. The data in the left image consists of laser scans, photos, and thermal images recorded at 13 different positions in the early afternoon. It strikes immediately that the building to the left appears significantly warmer than the building in the back. Even diffuse sunlight on a cloudy day distorts the measurements in a way that a meaningful analysis becomes impossible. After the conclusions drawn from this data a second set consisting of scans and thermal images from 15 different positions was recorded at night. From the result shown on the right it becomes clear that the correct temperature values allow for an analysis of the data.

(a) JACOBS DAY (b) JACOBS NIGHT

Fig. 7.28: 3D thermal model of building from the JACOBS data set highlighting the effect of diffuse sunlight on thermal measurements.

Conventional methods for documenting archaeological excavations are based on hand drawings on graph paper and archaeological notes. They are time consuming and prone to human error. In industrial archaeology time is limited especially when the historical artifacts were found on a construction site. Often the findings will be destroyed during the construction and thus the notes are all that remains of them. Laser scanning is a means of preserving a 3D electronic copy of the historic findings. The color information is important for the archaeologist's work. In combination with data processing and storage tools laser scanning has the potential to be part of a workflow that improves the archaeological documentation and allows to present results in the same form as before [105]. Fig. 7.29 shows an example of an excavation within a construction site. A pause in the construction comes at large expenses and should be avoided.

Fig. 7.29: 3D model of the BRÄUTIGAM excavation within a construction site.

To attract visitors museums want to promote their exhibitions in a realistic way. A 3D color model created with Irma3D can be visualized on a computer screen and presents a good overview of an exhibition (see Fig. 7.30). This was demonstrated at the interactive science exhibition in the UNIVERSUM Bremen. The goal of the experiment was to create a complete digital 3D model of the special exhibition "*Your life in numbers*" that presented facts, sculptures and interactive stations demonstrating the statistics of human life. This experiment was a first test for solving the art gallery problem in 3D. In the art gallery problem Irma3D has to calculate the optimal next best scanning position to fully capture the environment. As many exhibits are hanging from the ceiling full 3D information is necessary. Despite the visitors the robot managed to explore large parts of the museum autonomously. A video demonstrating the experiment is available from [12].

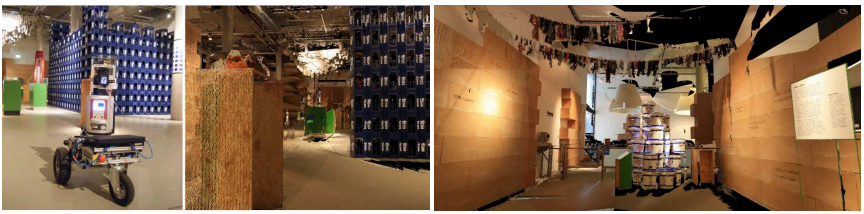
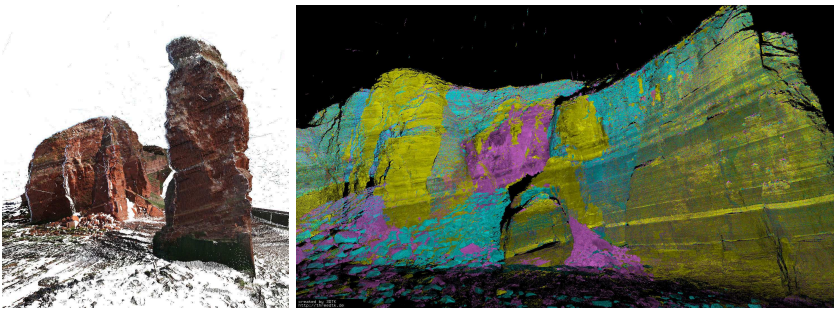


Fig. 7.30: Left: Irma3D collecting data in the UNIVERSUM Science Museum in Bremen. Right: Two views of the resulting 3D model.



(a) Scans 2014 (b) Comparison 2014 - 2016

(c) Birds (d) No birds

Fig. 7.31: HELGOLAND. (a) Color scans of the *Lange Anna* from 2014. (b) Comparison of scan position 4 from the different years. Scans are colored by reflectance information. Cyan, magenta and yellow indicate the different years. (c) Scan positions on the upper part from 2014. (d) Scan positions on the upper part from 2014 with a bird filter.

An application in the field of geology is shown in Fig. 7.31. The *Lange Anna*, a free standing rock column is the most famous landmark on HELGOLAND. The Buntsandstein the *Lange Anna* and most of the island consists of is extremely susceptible to erosion. To measure the change on the rock surfaces is therefore important for the safety on the island. Fig. 7.31 shows the colored scans from the data collection in 2014 as well as one selected scanning position from all three years. The initial pose estimates acquired using GPS and a compass were sufficient to register the data with a complete graph for the global optimization algorithm described in Section 5. The comparison of the data from scan position 4 clearly shows the difference. Large amounts of the rock have come down from the top. The 3D data also contains the information needed for a quantitative analysis.

The large number of birds flying around the *Lange Anna* are clearly visible in Fig. 7.31(c). To prepare the data for geological analysis a bird filter is apllied. This filter works by creating an octree of adaptable size and removing all voxels where the number of points falls below a

threshold. For the scan positions on the upper part of the island a voxel size of 2 m and a threshold of ten points per scan give the results depicted in Fig. 7.31. Most of the birds and only small areas of the rocks are removed. An alternative approach with higher computational complexity removes all points that have no neighbor within a small area in any of the other scans.

7.4 Summary

This chapter describes the principal component of multi-modal 3D mapping, the combination of the different modalities via calibration. Determining the transformation between the sensors allows to enhance the point cloud with color and thermal information from the cameras. The calibration is a semi-automatic process. The calibration pattern has to be moved into a different position for each data pair. The calibration patterns are a chessboard pattern for color cameras and a regular grid of light bulbs for thermal cameras, both fixed on a board. A region growing approach recognizes the board reliably in the point cloud. That way the correspondences in the date result in the transformation between the sensors.

When assigning color and thermal information to a point cloud occlusions need to be considered. This is achieved by using a ray tracing procedure in the kD-tree. The 3D geometry allows to discard data that was recorded under unfavorable cirumstances. Numerous examples demonstrate successful application of the approach.

Chapter 8

Applications

The previous chapters explain the generation of multi-modal 3D models, from autonomous data acquisition over post-processing to joining the data of different sensors at various positions into one common model. This chapter outlines applications of the methods presented so far. Section 8.1 uses the Hough Transform from Chapter 4.3 to detect the structural elements of building interiors and to reconstruct occluded parts. Section 8.2 goes one step further by classifying windows as open, closed or damaged. Section 8.3 gives an example of projecting the thermal information back into the scene for inspection purposes. Section 8.4 reconsiders the color assignment to point clouds through radiometric alignment while Section 8.5 analyses the use of SfM for generating point clouds from photo collections. Registering these point clouds to laser scans relates the camera poses to the coordinate system of the laser scans and allows to color the scans based on the photo collection.

8.1 Interior reconstruction using the 3D Hough Transform

3D point clouds represent direct distance measurements of a scene. As beneficial as the raw data is for calculations, the ultimate goal is often a more semantic representation of the environment. This section presents an application for the 3D Hough Transform presented in Chapter 4.3.4 for interior reconstruction [59]. According to [1] the previous attempts to automate interior and exterior reconstruction have rather focused on creating realistic looking models than on preserving geometric accuracy. Examples include [80, 96, 186, 196]. With the improvement in 3D laser scanning technology, the goal to automatically extract correctly scaled models has moved more into the research focus. Since architectural shapes of environments follow standard conventions arising from tradition or utility [73] one can exploit knowledge for reconstruction of indoor environments. An interesting approach is presented in [44] where sweeping planes are used to extract initial planes and a geometric model is computed by intersecting these planes. In outdoor scenarios precise facade reconstruction is popular including an interpretation step using grammars [22, 23, 166, 171].

The research proposed in [59] is an extension to previous work in [1, 2, 32] and combines the reconstruction of interiors that are suspect to occlusion and clutter with plane extraction to reconstruct a scene at high architectural level. A point cloud of an indoor environment acquired

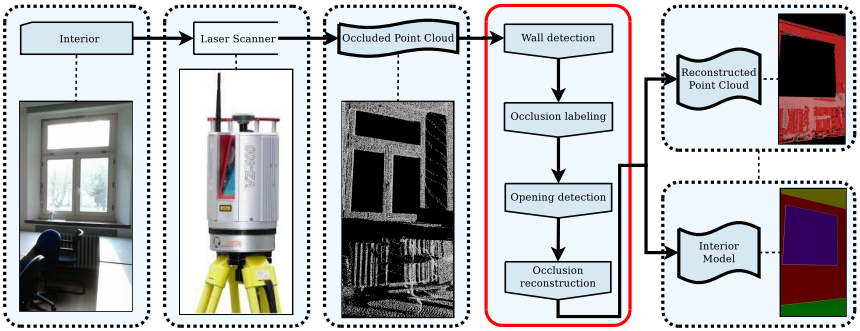


Fig. 8.1: Reconstruction pipeline.

with a 3D laser scanner suffers from occlusions. To convert it into a full reconstruction of the interior or even an interior model Adan et al. propose four steps, namely wall detection, occlusion labeling, opening detection and occlusion reconstruction [2]. The following demonstrates the interior reconstruction outlined in Fig. 8.1 on one scan from the BASEMENT data set. In Section 8.1.5 the approach is confirmed on the SEMINAR ROOM data set. The point clouds were reduced applying the octree based reduction to one point per cubic centimeter resulting in point clouds of approximately 5 million points per scan.

8.1.1 Wall detection

The wall detection starts by applying the RHT. Fig. 8.2 gives the initial planes that are analyzed to determine the main structural elements of the room. Assuming that the scanner was approximately horizontally aligned during scan acquisition, predefined epsilon values allow to classify the planar patches as vertical and horizontal surfaces, discarding at the same time those that do not fit into either category. From the remaining planes the highest and lowest horizontal planes are labeled as floor and ceiling, while the vertical planes are wall candidates. The implementation of the Hough Transform as presented in chapter 4.3.4 returns planar patches as regions of connected points. There the convex hull is used as it is clearly defined. For the purpose of interior reconstruction, however, the convex hull has the clear disadvantage that it might contain large areas that are not covered by points.

The concave hull gives a vague notion of the shape of the surface and enables the detection of structural elements in concave rooms. The alpha shapes implementation of *CGAL* [50] is used in this work. The concept is depicted in Fig. 8.3. Consider a fully connected graph. The concave hull reduces the graph to all edges that form the polygon that encloses all nodes. Figuratively speaking the alpha shapes algorithm can be thought of as a circular eraser with radius $\sqrt{\alpha}$. Starting from the convex hull it erases as much of the area of the polygon as possible. The nodes are obstacles over which the eraser cannot pass. Straightening the edges of the remaining area by using graph edges gives the concave hull. Two examples using different values for α are compared

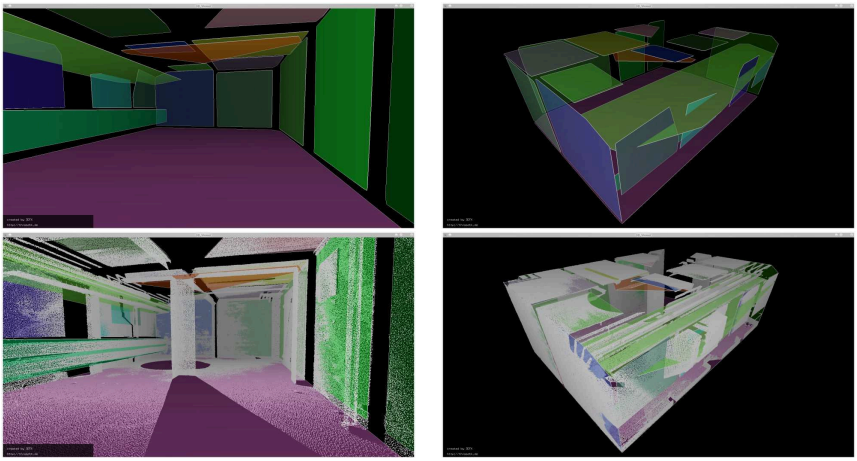
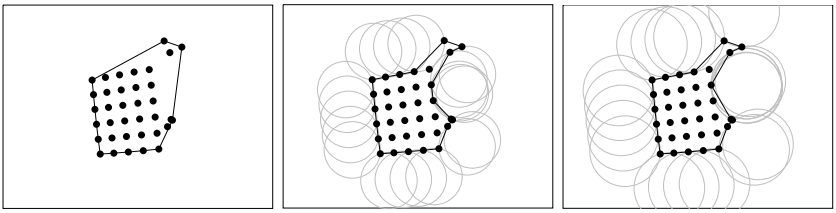


Fig. 8.2: Plane detection in the fragmented basement room. Above: Detected planes. Below: Planes and 3D points.

to the convex hull in Fig. 8.3. Using a very small α the solution consists of the unconnected point set. A very large α results in the convex hull.

When a plane is detected and the corresponding planar patch is determined, all points are projected onto the plane and the 2D concave hull is calculated. By grouping similar planes and averaging over the parameters of the wall candidates within such a group the actual walls are determined. Intersecting each of the thus calculated wall candidates with the floor and the ceiling plane yields the corners of the room. The concave hull of the corners on the respective planes give the planar patches representing the actual structural elements of the room as shown exemplarily in Fig. 8.4.



(a) convex hull $\sqrt{\alpha} = \infty$ (b) concave hull $\sqrt{\alpha} = 40$ (c) concave hull $\sqrt{\alpha} = 50$

Fig. 8.3: The Alpha Shapes algorithm using varying parameters.

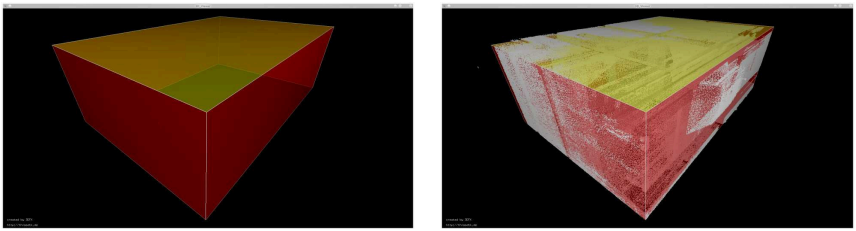


Fig. 8.4: Detected walls in the basement scan.

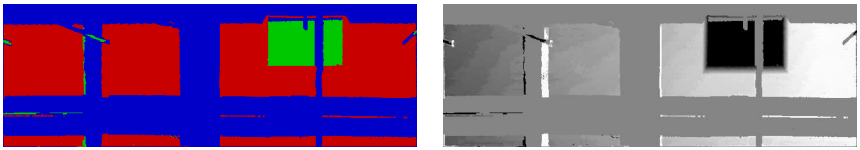


Fig. 8.5: Left: Labels (red `occupied`, blue `occluded`, green `empty`). Right: Corresponding depth image.

8.1.2 Occlusion labeling

For occlusion labeling the data is converted into an octree (cf. Section 4.1.1). The voxels that intersect with the surfaces are assigned one of the three labels `occupied`, `empty` or `occluded`. Voxels that contain points are marked as `occupied`. Ray tracing from the scanner position to the surface voxel using the Bresenham line algorithm [43] distinguishes between `empty` and `occluded` voxels. Merging data from several scanning positions and merging the labels into one unified representation helps to improve the labeling. This step in the algorithm calculates two images for each surface, a label image and a depth image as shown in Fig. 8.5. In the processing pipeline that is based on the ideas from [1] the two images form the input for the opening detection in the next step where a support vector machine (SVM) is trained to classify openings. Three of the 14 features for the SVM are taken from the depth image while lines in the depth image build the opening candidates.

8.1.3 Opening detection

The opening detection aims at classifying openings in the surfaces to distinguish between actual openings such as windows and doors and those areas that are occluded and should be filled in during the reconstruction. Following [1] an SVM is trained to classify opening candidates followed by k-means clustering. The input for the SVM is generated using image processing methods from OpenCV [156] to the depth and the label image. Applying the Canny edge detection algorithm [45] and the Sobel operator [52] in the horizontal and vertical direction on the depth image I results in three images I_1, I_2 and I_3. Using both the Canny edge detection and the Sobel operator makes the algorithm more robust. Merging these images by applying

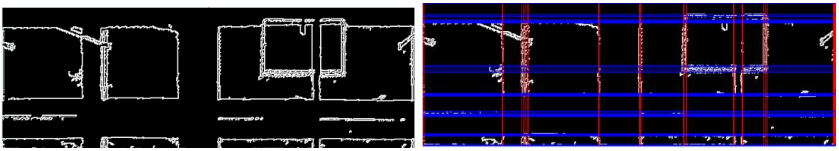


Fig. 8.6: Line detection in generated range images. Left: Image I_4. Right: Hough lines.

a threshold to each non-zero pixel yields a binary image I_4 with enhanced edges. The Hough line detection determines all the lines in image I_4. All non-horizontal and non-vertical lines are removed. An example of the merged image I_4 and the detected horizontal and vertical lines is given in Fig. 8.6 using one wall from the BASEMENT data set. Horizontal lines are drawn in red while vertical lines are drawn in blue. All possible rectangles are constructed from the remaining lines by determining the rectangle that is composed by taking combinations of two horizontal and two vertical lines. A filter removes rectangles based on criteria such as minimum and maximum area of the rectangle resulting in the opening candidates. Due to the nature of the Hough Transform redundant candidates appear where large clusters of horizontal or vertical lines have been detected.

Following [1] each opening candidate is described by 14 features:

1. candidate absolute area,
2. candidate width/height ratio,
3. candidate width/surface width ratio,
4. candidate height/surface height ratio,
5. distance from upper side of candidate to upper side of surface,
6. distance from lower side of candidate to lower side of surface,
7. distance from left side of candidate to left side of surface,
8. distance from right side of candidate to right side of surface,
9. root mean squared error to the best plane fitted to the rectangle surface,
10. percentage of occupied area,
11. percentage of occluded area,
12. percentage of empty area,
13. number of interior rectangles,
14. number of inverted U-shaped rectangles (specific to doors).

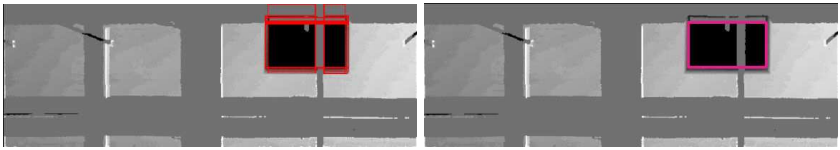


Fig. 8.7: Left: SVM learner applied to the Hough lines as presented in Fig. 8.6. Right: Result of k-means clustering and check for overlapping candidates.

Fig. 8.8: Left: Inpaint mask. Right: Final 2D opening.

Features (10), (11) and (12) are determined from the labels image while the other features are computed from the results of the line detection algorithm. A support vector machine is a machine learning approach that solves classification tasks by constructing hyper planes in multi-dimensional space. Here a Radial Basis Function (RBF) is used as a kernel in a binary SVM model. The SVM is trained for the opening features using a small database of actual openings from various buildings across the campus of Jacobs University Bremen. For classification each opening candidate is input to the trained SVM which classifies the candidate as opening or not based on similarity to the training examples. Many of the resulting rectangles that are classified as openings represent the same opening. k-means clustering on the center and the absolute area of the candidates groups those candidates into clusters. The number of clusters k is identical to the number of actual openings on the surface. For each cluster the representative is chosen that has the largest `empty` and the smallest `occupied` area. Fig. 8.7 shows the procedure for the example wall. The one opening is correctly classified. The various representations of the opening in the left image are clustered and the final representative of the opening is shown in the right image.

With the computed final openings in the 2D image the label image is updated. All `empty` labels outside the openings are changed to `occluded`. The result is the inpaint mask I_m that is used to fill the gaps in the surfaces [1, 175]. The inpaint mask I_m, depicted in Fig. 8.8, shows the patches of the surface that are missing information and need to be filled. The high degree of fragmentation in the Basement data set caused mainly by pipes in the environment leads to surfaces with approximately 50% occlusion. The high amount of reconstruction required for this data set proves challenging for reconstruction. Apart from the 2D inpaint mask this component results in a complete 3D model of the interior including the openings. The next component reconstructs reconstructing the actual 3D data from the 2D depth map and the inpaint mask.

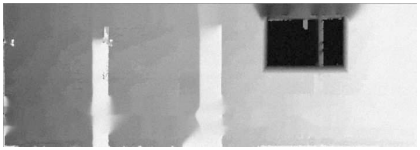


Fig. 8.9: Reconstructed depth image.

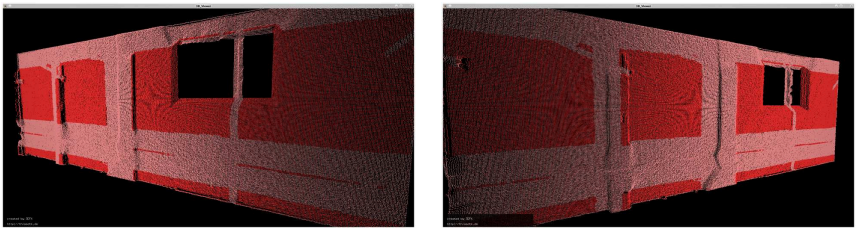
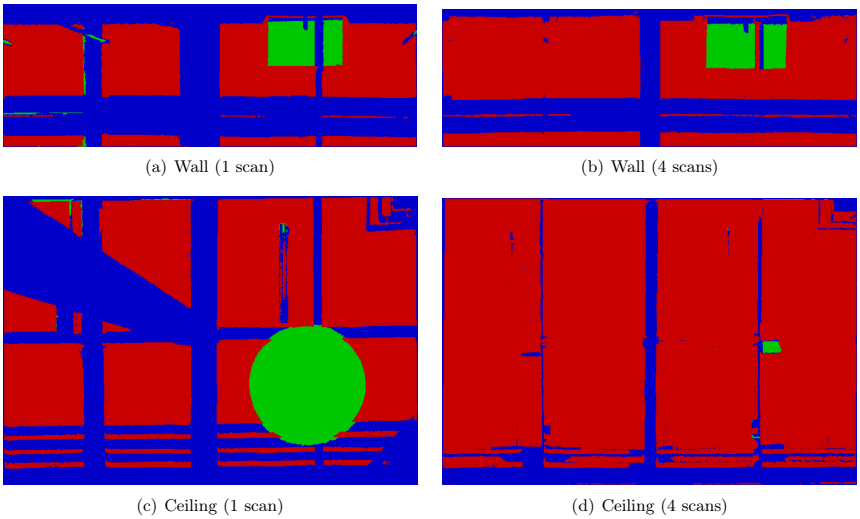


Fig. 8.10: Views of the reconstructed 3D point cloud. The light red points have been added through the inpaint process.

8.1.4 Occlusion reconstruction

The last component generates a reconstruction of the original 3D point cloud based on the 2D depth map and the inpaint mask. It uses a gap filling algorithm to fill in the missing information in the point cloud induced by occlusions. Inpainting fills holes in images using the information from surrounding areas. This is commonly used for restoration of historic photographs or removal of objects in an image. Here the OpenCV [156] implementation of the Navier-stokes method [15] is applied to the original depth image I of each surface using the previously generated inpaint mask I_m. The method is based on fluid dynamics. The intensity is thought of as a stream function of an incompressible flow that is transported into the empty region. The gradients at the boundaries to the inpaint region determine the direction and the magnitude of the flow. As the intensity of the depth image is proportional to the depth values this method yields directly a reconstructed 2D depth image I_r where the empty regions have been filled with information computed from the inpainting.

To achieve a more natural look and feel of the inpainted parts the mean and standard deviation of the original depth image in the depth axis are computed and Gaussian noise with the same mean and standard deviation are added to the reconstructed image I_r, thus yielding the final reconstructed depth image I_f of the surface as shown in Fig. 8.9. From I_f the reconstructed 3D point cloud of the surface is directly extrapolated using the plane parameters of the surface and the depth for the distance to the plane. Two perspectives of the final point cloud are shown in Fig. 8.10. The light red color indicates points that were added by the reconstruction pipeline.



(a) Wall (1 scan) (b) Wall (4 scans)

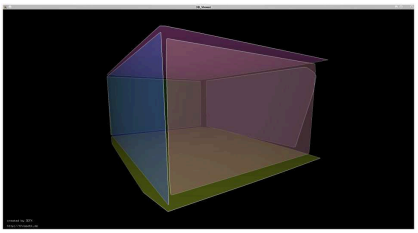
(c) Ceiling (1 scan) (d) Ceiling (4 scans)

Fig. 8.11: Comparison of the BASEMENT room labeling between a single and four registered scans.

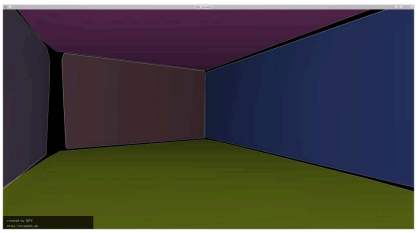
8.1.5 Results and discussion

The individual steps of the reconstruction pipeline were demonstrated on one scan from the BASEMENT data set. The data set proved ideal for testing the pipeline as the walls were heavily fragmented due to occlusions from the pipes. Nevertheless, all six surfaces of the interior were accurately detected. The label image contained large areas of all three labels. The `occluded` areas coming from the pipes running across the entire wall pose a challenge as well as a small vertical pipe that occludes the only opening in the wall. The ability of the algorithm to cope with these challenges suggests that it can also deal with windows consisting of several smaller window panes. It avoids to detect several small windows in this scenario by checking for overlapping candidates and choosing the candidate with the largest `empty` area. Naturally the algorithm benefits from the availability of several registered scans that reduce the occlusions in the original point cloud. This is shown in Fig. 8.11 by comparing the label images of the previously evaluated wall and the ceiling with the model generated from all four registered scans. The occlusions on the wall coming from the pillar in the room are reduced as well as the occlusions on the ceiling introduced by the steal girders. The remaining `occluded` areas are caused by elements that are close to the wall and cannot be omitted. The large opening in the ceiling caused by the limited field of view of the scanner also disappears.

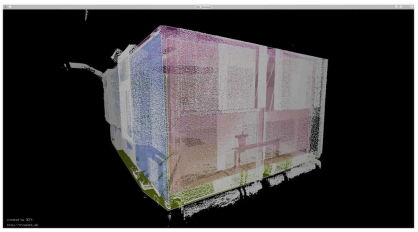
To verify the pipeline and to ensure that the SVM is not overfitted results from the SEMINAR ROOM data set are presented in Fig. 8.12 and 8.13. The 3D point cloud of the seminar room



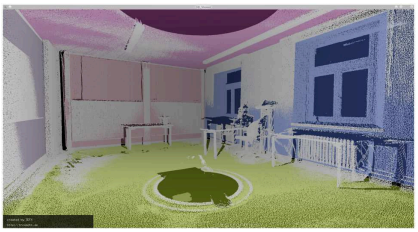
(a) Planes exterior



(b) Planes interior



(c) Planes and points exterior

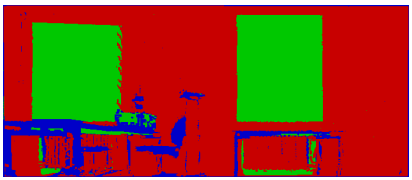


(d) Planes and points interior

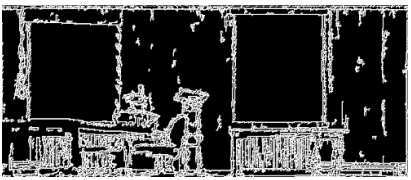
Fig. 8.12: Detected planes in the seminar room (above) and as overlay to the 3D point cloud (below).

(Fig. 8.12) has a simpler structure than the basement room. All walls and the ceiling are directly detected as individual planes due to the lack of larger occlusions. Only smaller obstacles occlude the structural elements of the room. The wall to the right in Fig. 8.12(d) has two windows composed of three window panes and is occluded by several objects. These clearly show up in the depth image I_4 and effect the image processing pipeline in the intermediate results presented in Fig. 8.13. The obstacles lead to a large amount of horizontal lines. The resulting opening candidates are correctly removed in the classification step so that the result from the SVM consists only of correct opening candidates from which the two correct openings are chosen after clustering. The inpaint mask aims at covering all the small occlusions caused by the various objects and yields a complete wall as reconstructed point cloud.

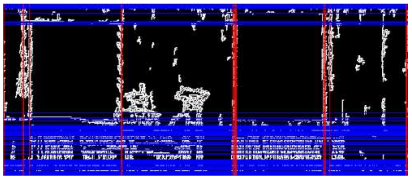
The presented system for architectural 3D interior reconstruction from 3D points clouds is implemented in 3DTK using image processing components from the OpenCV and CGAL libraries and was successfully tested in two scenarios. The four main components rely each on the output of the previous one. The modular nature of the pipeline presented here allows to extend or replace each of the individual components by alternative solutions to the respective problem. Based on the fast plane detection the wall detection and labeling components are highly reliable. The opening detection and the scene reconstruction, however, require specific fine tuning. A typical drawback when using machine learning methods is the dependence on the training data



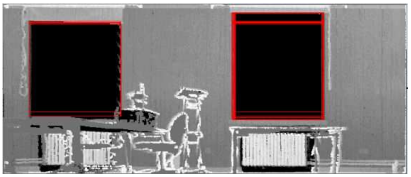
(a) Labels



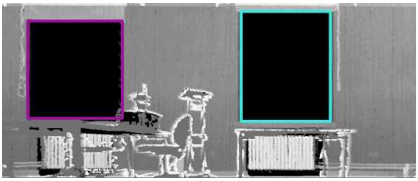
(b) Image I_4



(c) Horizontal and vertical Hough lines



(d) Opening candidates

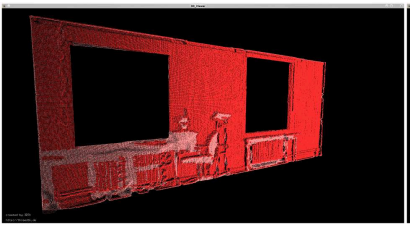


(e) Final openings after k-means clustering

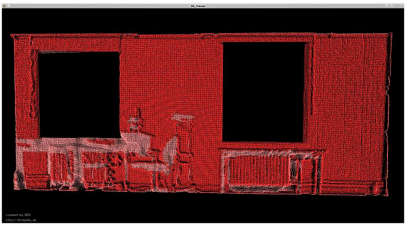
(f) Inpaint mask

(g) Finals 2D openings

(h) Reconstructed depth image



(i) Reconstructed 3D point cloud



(j) Reconstructed 3D point cloud

Fig. 8.13: The image processing pipeline for the surface from the seminar room containing two windows yields the reconstructed 3D point cloud of the wall.

for the SVM. The method is currently limited to rectangular shaped windows, a valid assumption, but for differently shaped windows this component needs to be extended. The reconstruction component fills the areas indicated by the inpaint mask. This approximates what the surface would look like based on the surroundings. Needless to say, it is impossible to know what the occluded areas actually look like. Therefore, inpainting should be reduced to small occluded areas by using exploration methods and NBV planning.

8.2 Window classification

To use 3D models of buildings efficiently in maintenance and inspection, preservation or other building related tasks the association of semantics to the raw sensor data is important. Without this additional information inspection tasks are merely moved from the inspected object to the computer. Interpretation of the data enables the system to guide the user to the points of interest. This section builds upon the results of the thermal modeling described in Chapter 7 by presenting a pipeline, illustrated in Fig. 8.14 to infer semantic information to 3D thermal models. Similar to the previous section a sequence of algorithms first differentiates between objects in a room and objects that constitute the room, e.g., floor and windows. In the main step of the workflow the thermal information is used to construct a stochastic mathematical model – namely Markov Random Field (MRF) – of the temperature distribution of the detected windows. As a result, the MAP (maximum a posteriori) framework is used to further label the windows as either `open`, `closed` or `damaged`. Fig. 8.15 shows examples of the different classes. Classification approaches that use training require a huge training data set. The benefit of modeling the classification problem instead eliminates this requirement.

8.2.1 Problem definition

Inferring higher level knowledge about the properties of an object consists of two highly related subproblems. First, the detection of objects that belong to a certain class emphasizes on understanding and modeling of time-invariant properties of a certain class of objects. The properties are used to recognize an object of that class, e.g., all windows are made of glass. Second, inferring about the object, rather than the class, by recognizing properties of an object that are observed in a certain time frame under a certain condition, gives information about the state of the object rather than the class it belongs to. The solution space of the first problem is the domain/problem space of the second problem. In the following, window detection, a problem of the first class is followed by a problem of the second class, the labeling of windows as `open` (**O**), `closed` (**C**) or `damaged` (**D**), i.e., a window without the proper insulation.

The goal is to find pre-processing steps that enable one to reduce the first task, i.e., window detection, to the estimation of a mapping function $f(M)$. M is the fully registered thermal 3D model of the room and the output $t_s = f(M)$ where $t_s \in \mathbb{R}^N$ is the temperature distribution that gives a temperature value for each of the N 3D points representing the window. Considering the 3D points as random variables that take a temperature value from $\mathbb{R}$ assigning the label `closed`, `opened` or `damaged` to the window is formulated as a probability distribution modeling problem. The discriminative model approach models the posterior probability $P(Y|X)$ directly from the data. As this approach has superior predictive capabilities compared to the generative model

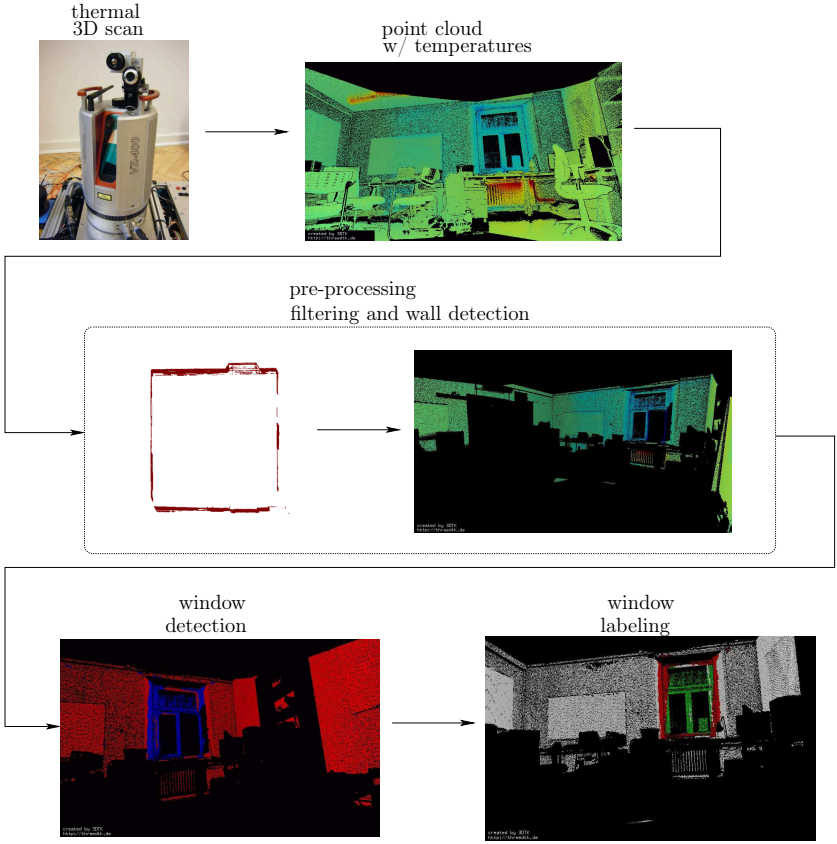


Fig. 8.14: Overview of window detection and labeling pipeline.

approach that attempts to model the joint probability $P(X, Y)$ [111] the posterior probability of a label $l \in \{\mathbf{C}, \mathbf{O}, \mathbf{D}\}$ is modeled as

$$P(l|t_s) = \frac{p(t_s|y)P(l)}{p(t_s)}, \tag{8.1}$$

Due to the lack of prior information $p(t_s)$ is assumed to be exactly the same for all the labels and has no effect on the label specific posterior. Likewise the probability of a window to be in

Fig. 8.15: Photo of eight windows (left) and the corresponding infrared image (right). In the thermal image heat losses are clearly detectable. In the top row the windows show different temperature distributions. The first window on the left belongs to an empty office which is not heated. The top part of the second window is open. The third window is closed and has proper insulation. The fourth window shows unusually high temperatures. This was caused by a partially detached rubber insulation strip that allowed the air to flow through.

either of the states is assumed to be identical. Therefore, the modeling task is simplified to

$$P(l|t_s) \approx p(t_s|l). \tag{8.2}$$

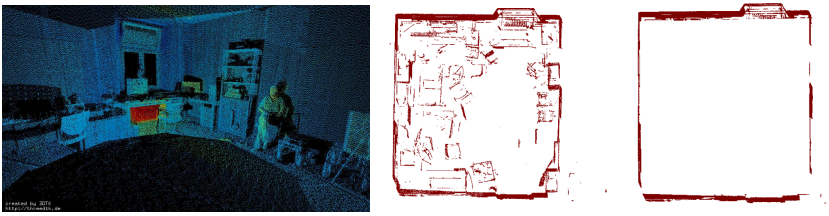
Labeling a window as `open`, `closed` or `damaged` is thus formulated as estimating the conditional probability distribution of every label, followed by a decision rule. To minimize the expected error the decision is based on the MAP

$$\hat{l} = \arg\max_{l_i \in Y} P(l = l_i|t_s), \tag{8.3}$$

where $\hat{l}$ is the final label assigned to the temperature distribution of a window $t \in \mathbb{R}^n$, and $L = \{\mathbf{C}, \mathbf{O}, \mathbf{D}\}$ [19]. The following sections present a solution to the problem formulation given in Eq. (8.1).

8.2.2 Pre-processing

For the window detection in this section the thermal properties as measured with a thermal camera are exploited. The core idea is to exploit the difference in thermal conductivity between the material of the wall and the window as well as the window frame. Typical materials for window frames are wood, fiberglass, vinyl or aluminum while walls are made of bricks, concrete or various multi-layer materials including insulation layers. Consequently a temperature difference occurs that can be used to distinguish one from the other. This requires the removal of all the objects, clutter as well as the floor and the ceiling from the point cloud data leaving only the walls and the windows. To achieve this task the method from section 8.1 can be used. An alternative solution converting the data into a 2D representation and detecting the walls as lines is presented in [54] and shown exemplarily for the test data set in Fig. 8.16. After detecting the lines representing the walls all points that are further than a distance threshold from the walls



(a) (b) (c)

Fig. 8.16: (a) A 3D thermal model of the AUTOMATION LAB at Jacobs University Bremen. (b) 2D projection of the room with the xy-axis colored in red. (c) The detected outer rectangular shape, or walls of the room.

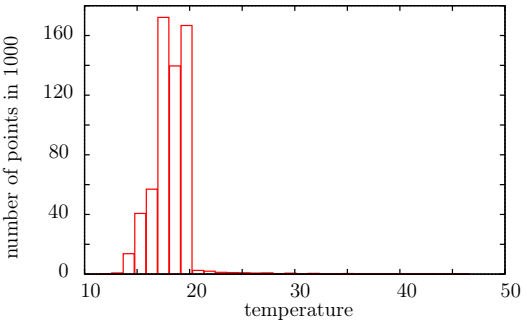


Fig. 8.17: Typical temperature distribution of a 3D scan with walls and windows only.

are removed. The clutter in the room is ignored during further computations. The result of the pre-processing step is a 3D model of the walls of the room.

8.2.3 Window detection

The remaining point cloud is dominated by points from the wall which typically have a similar temperature. In Fig. 8.16(a) the 3D points on and near the window show a considerable difference in temperature compared to the wall points. This difference is the main feature for the window detection technique. Creating a histogram of the temperature values of a point cloud consisting only of walls and windows gives a bell curved shape as depicted in Fig. 8.17. The wall points constitute the peak of the curve while temperature peaks in the data are reflected in the flat ends of the temperature distribution. The higher temperatures at the end of the curve correspond to warm objects near the wall such as computer monitors or heaters while the lower end corresponds to cooler objects, i.e., mostly windows. This is of course only valid for the cold season when the outside temperature is significantly lower than the indoor temperature. During the warm season

the usage of air conditioning units instead of heaters requires a different interpretation. Let μ be the mean temperature of the 3D points. The mean and standard deviation of the temperature distribution allow to define a thresholding constant independent of the exact shape of the curve:

$$\text{temp}_{\pm} = \frac{1}{N}\sum_{j=1}^{N} t_j \pm \sqrt{\frac{1}{N}\sum_{j=1}^{N}(t_j - \mu)^2}, \tag{8.4}$$

that identifies points with uncommon temperature values. t_j is the temperature value for each of the N points. For the cold season, the scenario considered here, the lower boundary detects points with low temperatures and thus potential window points. If heaters are switched off, the distribution at the upper end will change but this has no impact on the window detection as long as the difference between the inside and outside temperature remains significant. All points that fall below the lower threshold temp_ are clustered based on spatial distance. Under the assumption that the points were recorded in an ordered structure, as typically done by a laser scanner, the clustering is achieved by starting a cluster with a randomly selected point. Further points are randomly selected and added to an existing cluster if their distance to any of the points is below a threshold. Otherwise they are the first point in a new cluster. For arbitrary point clouds clustering methods that emphasize compactness and connectivity, such as OPTICS (Ordering Points To Identify the Clustering Structure) [5] are necessary.

Each cluster with a sufficient number of points represents a window candidate that is further processed individually. Fitting a plane through the points in a cluster (cf. Section 4.3.1) allows to determine the boundary size of the window. As an approximation the problem is reduced to axis-aligned computations. Ignoring the component with the smallest variance the minimum and maximum of the remaining components are determined by finding the points with the maximum distance to the mean of the cluster. The boundary points are directly calculated from the extreme points. The size of the boundary is used to determine the voxel size for octree based point reduction (cf. Chapter 4.1) with the goal to achieve an identical number of points per window independent of the window size.

8.2.4 Modeling state of a window

For the classification a window is considered to be in three distinctive states or labels, namely `closed`(**C**), `opened`(**O**), or `damaged`(**D**). A damaged window refers to a window without the proper heat insulation. The probability distribution of the window states given the 3D thermal point cloud thereof are computed using MRF.

Markov Random Fields

The concept of Markov Random Fields (MRF) has been investigated intensively in the 1970s [47]. MRF are part of a class of models used to express the statistical dependencies between several random variables arranged in a form of spatial configuration, often expressed as a graph. Their main property, the Markov property, simplifies the joint distribution modeling by inferring conditional independence on an undirected graph. First, any two subsets are conditionally independent given a separating subset. Second, any two non-adjacent variables are conditionally independent

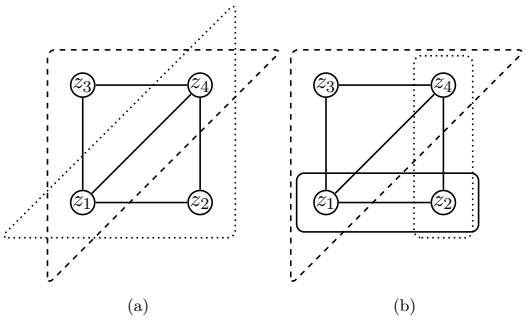


Fig. 8.18: Two different sets of cliques in a graph corresponding to the factorizations (a) $P(z_1, z_2, z_3, z_4) = \phi_1(z_1, z_2, z_4)\phi_2(z_1, z_3, z_4)$ and (b) $P(z_1, z_2, z_3, z_4) = \phi_1(z_1, z_3, z_4)\phi_2(z_1, z_2)\phi_3(z_2, z_4)$.

given all other variables. Third, any variable is conditionally independent of the other variables given its neighbors.

According to the Hammersley-Clifford theorem a joint probability distribution of several random variables $\{z_1, \cdots, z_n\}$ can be factorized over the cliques of the graph as $P(z_1, \cdots, z_n) = \prod_{j=1}^{n} \phi_j(\psi_j)$, where ϕ_j is any real valued function defined for each clique, that describes the compatibility of the random variables within the clique. If the positivity criterion is fulfilled, i.e., every joint realization of the random variables has a strictly positive probability value over the entire domain, this yields a representation of the MRF. An example graph is shown in Fig. 8.18, with two different clique sets corresponding to two factorizations for $P(z_1, z_2, z_3, z_4)$.

Converting the compatibility functions into potential functions in exponential form results in an alternative representation using the Boltzmann or Gibbs distribution:

$$p(t; \theta) = \frac{1}{Z(\theta)} \exp\left(\frac{-E(t, \theta)}{T}\right) \tag{8.5}$$

where Z is the normalization constant, T is a controlling constant, t is a certain realization of the random variables, and $E(\cdot)$ is an energy function specific for the system being modeled. θ is a set of free constant parameters of the system, that will be discussed later.

The Boltzmann distribution is the essential part of modeling a specific system as a MRF and thus the mathematical properties, i.e., the exact relationship between the dependent probability distribution $p(\cdot)$ and the two independent variables $E(\cdot)$ and T are crucial to understand. Most important is the relationship between the energy function and the probability distribution. By taking the negative of the energy function as input to the exponential function the energy function and the probability distribution have an inverse variation. With $E(\cdot) > 0$, the functions relate as $p(\cdot) \propto \frac{1}{E(\cdot)}$. Thus, the Boltzmann distribution favors states with low energy values. For the probability distribution and the controlling constant T one can see that when T gets small the standard deviation of the probability distribution gets small, thus $T \propto \sigma$, where σ is the standard deviation.

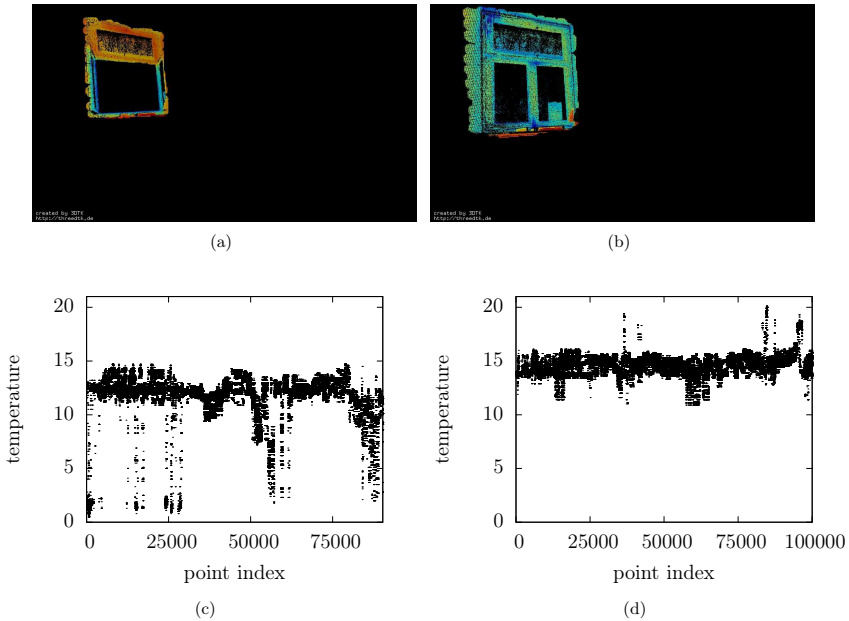


Fig. 8.19: (a) Open window colored according to the thermal distribution. (b) Closed window colored according to the thermal distribution. (c)+(d): the temperature distribution of each window, respectively.

Designing the energy functions

MRF in image analysis are often used for segmentation or classification tasks where each node represents a pixel and the random variables are the labels assigned to each individual pixel. In these scenarios one tries to find an assignment of labels that minimizes the energy function. However, for the window classification the problem is a different one. The random variables are the temperature values of the points measured by the thermal camera. Thus the goal is to design one energy function for each label **C**, **O** and **D** that takes the points with their temperature values as input and results in a high value for the probability of the Boltzmann distribution for the correct label and low probability values for the other labels. At the same time instances of a class should always minimize the energy function for the Boltzmann distribution of the class better than instances from other classes. Modeling the classification problem thus means to formulate energy functions that favor distinctive features of the individual window states.

The characteristics of the different window states were derived from thermal images of windows as shown in Fig. 8.15. A `closed` window is expected to have a temperature distribution that exhibits a very small variance in temperature while an `open` window has a comparably

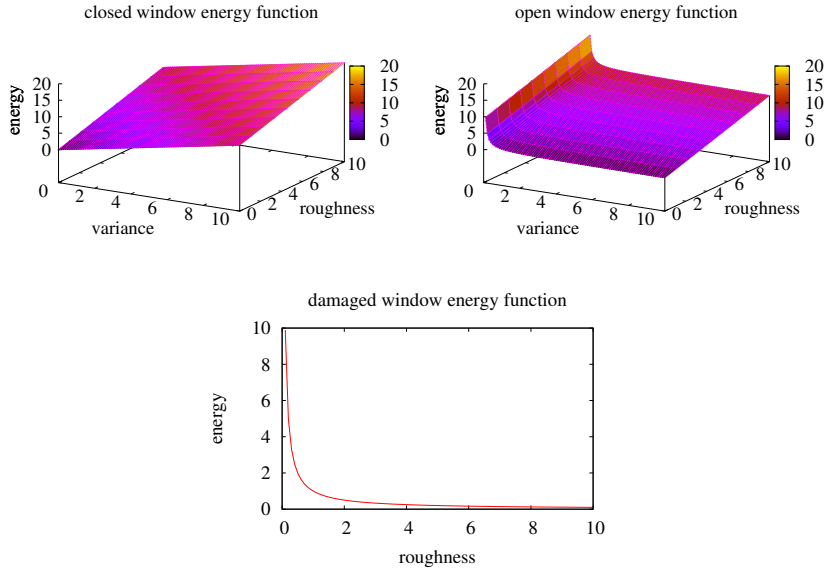


Fig. 8.20: The energy functions defined on variance and/or smoothness where all the weighting constants are assigned to 1. Top Left: `closed` window's energy function, Top Right: `open` window's energy function, Bottom: `damaged` window's energy function.

higher variance. This has shown to be a very robust feature to distinguish one from the other invariant to the maximum temperature value (cf. Fig. 8.19). `damaged` windows are particularly detectable because of their non-smooth temperature distribution, i.e., large temperature differences between neighboring points, as opposed to the smooth distribution of `open` and `closed` windows with small temperature differences between neighboring points. Energy functions ought to be designed such that the function is close to the maximum for weak cases of the designated feature and reaches its minimum for representatives of the class, respectively. Since the features of each state should be distinctive they must be different from each other and the design of the energy function is state specific. Thus the variance and the smoothness are used to model the energy function of `open` and `closed` windows while for the `damaged` windows the non-smoothness is the main characteristic. The following equations model the derived energy function for each state:

$$E_C(t_s, \theta_c) = \alpha_{1c} \sum_{j=1}^{N} \frac{(t_j - \mu)^2}{N} + \alpha_{2c} \sum_{j=1}^{N} \frac{d(x_j, K)}{N} \tag{8.6}$$

$$E_O(t_s, \theta_o) = \alpha_{1o} \frac{N}{\sum_{j=1}^{N} (t_j - \mu)^2} + \alpha_{2o} \sum_{j=1}^{N} \frac{d(x_j, K)}{N} \tag{8.7}$$

$$E_D(t_s, \theta_d) = \alpha_{1d} \frac{N}{\sum_{j=1}^{N} d(x_j, K)}, \tag{8.8}$$

where $\theta_c = (\alpha_{1c}, \alpha_{2c})$, $\theta_o = (\alpha_{1o}, \alpha_{2o})$, $\theta_d = (\alpha_{1d})$ are weighting constants for each term of the energy functions. N is the number of random variables x_j (3D points) representing the window. d is a function that calculates the average temperature difference to the k nearest neighbors of x_j based on spatial relation. Finally, t_j is the temperature value of the jth point x_j of the window and μ the mean temperature value. The first term encourages small variance for the `closed` window energy function (8.6), and a high variance for the `open` window energy function (8.7). The second term for both `open` and `closed` windows emphasizes smoothness. The `damaged` window energy function (8.8) has only one term encouraging a non-smooth temperature distribution. The energy functions are visualized in Fig. 8.20. They are defined on a very high dimensional space and computing the normalization constant Z for the Boltzmann distribution analytically is mathematically complex and therefore not done in practice. Hence, it has to be approximated with an appropriate numerical method, the Monte Carlo integration.

Monte Carlo integration

Monte Carlo integration attempts to approximate a result from an experiment. Monte Carlo methods are usually better for the approximation of higher dimensional integrals than numerical methods [132]. From equation (8.5) it is easily seen that the partition function

$$Z(\theta) = \int_{\mathbb{R}^N} \exp\left(\frac{-E(t, \theta)}{T}\right) \mathrm{d}t \tag{8.9}$$

describes the normalization constant Z of the Boltzmann distribution for N continuous random variables defined on the same probability space under the condition that $\int p(t, \theta) = 1$ holds true. As the energy functions E_C, E_O and E_D are high dimensional and a numerical solution to the integral is not readily available Monte Carlo integration with importance sampling is utilized. Intuitively the process computes the area under the function randomly and averages over the results. By definition the expected value $E[g(x)]$ of a function $g(x)$ is calculated as

$$E[g(x)] = \int_{\mathbb{R}^N} g(x) p(x) \mathrm{d}x,$$

where $p(x)$ is the probability density function. Using the law of large numbers, instead of integration the expected value is approximated randomly choosing samples based on the probability and averaging their results:

$$E[g(x)] \approx \frac{1}{M} \sum_{i=1}^{M} g(x_i).$$

For $M \to \infty$ the error between the approximation and the expected value converges to zero. Importance sampling corresponds to a change of integration variables. Without loss of generality equation (8.9) can be expressed as:

$$Z = \int_{\mathbb{R}^N} \frac{\exp\left(\frac{-E(t,\theta)}{T}\right)}{\tilde{p}(t)} \tilde{p}(t)\,\mathrm{d}t = E_t\left[\frac{\exp\left(\frac{-E(t,\theta)}{T}\right)}{\tilde{p}(t)}\right] \tag{8.10}$$

where $\tilde{p}(t)$ is a positively valued probability distribution with $\int \tilde{p}(x)\mathrm{d}x = 1$ that may be interpreted as a probability density function. $t \in \mathbb{R}^N$ is the realization of the random variable where N is the number of random variables or 3D points. $E_t[\cdot]$ is used to represent the expectation of the function over the random variables. The law of large numbers assures the approximation of the expected value with a large number of samples and thus Eq. (8.10) is further reduced to:

$$E_t\left[\frac{\exp\left(\frac{-E(t,\theta)}{T}\right)}{\tilde{p}(t)}\right] = \frac{1}{M}\sum_{j=1}^{M} \frac{\exp\left(\frac{-E(t_j,\theta)}{T}\right)}{\tilde{p}(t_j)}. \tag{8.11}$$

$t_j \in \mathbb{R}^n$ is sampled according to $\tilde{p}(t)$ M times, the larger M is the better the approximation will be. For facility of notation the substitution $b(t,\theta) = \exp\left(\frac{-E(t_j)}{T}\right)$ is used in the following.

The major practical difficulty in using the Monte Carlo integration is the design of $\tilde{p}(t)$ especially for a function defined in a large space. Assuming, for instance, $\tilde{p}(t)$ to be uniformly distributed the fraction $b(t,\theta)/\tilde{p}(t_j)$ becomes almost infinity since $\tilde{p}(t)$ will be extremely small due to the high dimensionality of the space. Theoretically, this is solved by taking an infinite amount of samples from $\tilde{p}(t)$. In practice a hand designed probability distribution $\tilde{p}(t)$ has to be found that tracks the energy function very well, where $\tilde{p}(t_j)$ is small when $b(t,\theta)$ is small and vice-versa. As a result, the fraction $b(t,\theta)/\tilde{p}(t_j)$ becomes a more pragmatic number that contributes to the estimation significantly. Moreover, a well-tracking probability distribution allows for a more accurate expectation and in effect a better estimate with a smaller number of samples, comparatively.

Designing a probability distribution

The energy function of each label is inherently different, and thus, the design of the probability distribution is also label specific. The goal is to design a probability distribution that tracks each energy function well, which means $K \cdot \tilde{p}(t) \propto b(t,\theta)$. Apart from the first condition, the good tracking capability, a well designed probability distribution has to satisfy a second condition, $\tilde{p}(t)$ should not be negligibly small regardless of its tracking capability, which is a characteristic of most probability distributions defined in a vast space. To meet the first condition a Markov chain is designed as follows:

$$\begin{aligned} X_{i+1} &= X_i + W \cdot \mathcal{N}(0,\sigma^2) \\ X_0 &= \mathcal{N}(k,\sigma^2), \end{aligned} \tag{8.12}$$

where X is a random variable taking a temperature value from $\mathbb{R}$, i is the index of the random variable and X_0 is the initial state of the process. W is a weight used to control the rate in

variation, k is a constant that can be tuned accordingly and σ is the standard deviation of the Gaussian distribution $\mathcal{N}$. To generate a temperature distribution of a window with N random variables or 3D points the discrete time stochastic process (8.12) is ran N times.

Based on equation (8.12) and using the Markov property to simplify the computation of the joint distribution the probability of a temperature distribution instant $t = \{x_n \cdots, x_0\}$ is calculated as:

$$\begin{aligned} p(x_n, \cdots, x_0) &= p(x_n, \cdots, x_1|x_0)p(x_0) \\ &= p(x_n, \cdots, x_2|x_1, x_0)p(x_1|x_0)p(x_0) \\ &\vdots \\ &= p(x_n|x_{n-1}) \cdots p(x_2|x_1)p(x_1|x_0)p(x_0) \end{aligned} \tag{8.13}$$

with:

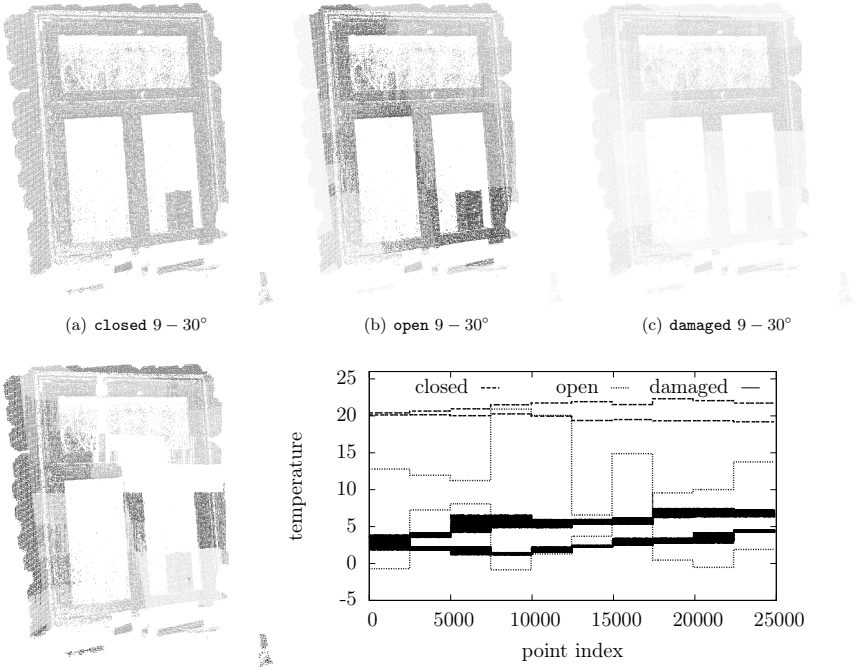
$$p(x_i|x_{i-1}) = \frac{1}{\sigma\sqrt{2\pi}} e^{r^2/2\sigma^2}, \tag{8.14}$$

where r is a number sampled from the Gaussian distribution $\mathcal{N}(0, \sigma^2)$ at the ith step. Thus, equation (8.12) describes a probability distribution that is able to generate samples $t = \{x_n, \cdots, x_0\}$ that track the energy function of each label. The second condition for the probability distribution is not to be negligibly small. The solution to this problem is to cluster points according to their spatial location using an octree, and treat each cluster as a random variable, i.e., every point in the octree voxel (cluster) will have the same temperature value. This, in effect, will reduce the domain space from the number of points to the number of clusters which means the joint distribution is highly scaled.

Examples of the sample data are shown in Fig. 8.21. The free parameters W and σ of equation (8.12) have to be tuned according to the specific energy function. For the `closed` window $W = 1$ and $\sigma^2 = 0.3$ are assigned. A small variance leads to high values in equation (8.14) and (8.13). Less variance between consecutive points x_{i-1} and x_i by itself encourages smoothness. For the `open` window temperature distribution the variance is much higher than for the `closed` window, but the smoothness should be exactly the same. Hence, setting $W = 20.0$ in the `open` window case aims at having a higher variance between clusters, but not between points, as points within a cluster are still assigned the same temperature. As a result equation (8.14) and (8.13) will be high valued for states with high variance yet smooth temperature distribution. Increasing W proposes states with higher variance without causing the probability distribution to shrink which would happen if simply increasing the variance. Thus, the variance is left as $\sigma^2 = 0.3$.

The most distinctive feature of a `damaged` window is the non-smoothness of the temperature distribution. The design of the probability distribution for the `damaged` window energy function is based on equation (8.12) with modifications on the handling of points inside a cluster. For the `open` and `closed` windows each point in a cluster is assigned exactly the same temperature but for the `damaged` window the assignment is done as follows:

$$t_j = X_{i-1} + W_2 \cdot \mathcal{U}(0, 1) \cdot r, \tag{8.15}$$

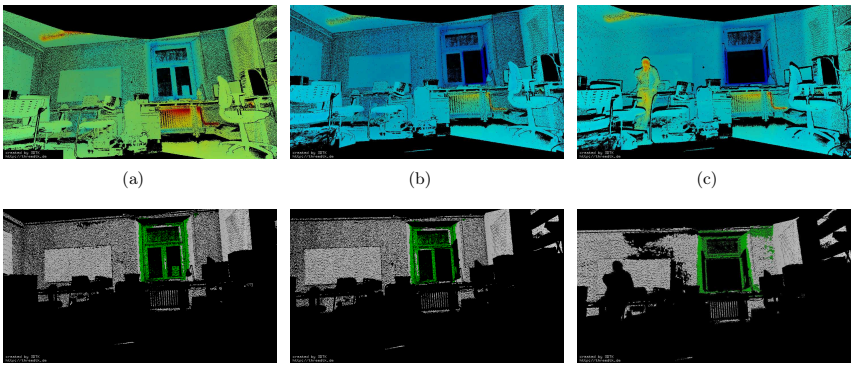


(a) `closed` $9-30°$ (b) `open` $9-30°$ (c) `damaged` $9-30°$

(d) `closed` $18-20°$ (e) Temperature distribution

Fig. 8.21: Examples of the randomly generated temperature distributions for the labels `closed`, `open` and `damaged`. (a) to (d) show the window with points colored in gray scale according to the temperature. The characteristics modeled into the probability distribution functions are clearly visible. The `open` window has a large variance compared to the `closed` and `damaged` window. The spatial clusters, i.e., octree voxels show up as squares of the same temperature. (e) shows exemplary temperature distributions for each label. This plot points out the identical temperatures within a cluster for the `open` and `closed` window and the uniformly distributed temperature values within a cluster of the `damaged` window.

where t_j is the temperature value for the point $\boldsymbol{p}_j$ in a cluster with value X_i, X_{i-1} is the value of the previous cluster, r is a sample generated from $\mathcal{N}(0, \sigma^2)$ at the ith step, and $\mathcal{U}(0, 1)$ is a uniform distribution that is sampled iteratively $\forall \boldsymbol{p}_j \in X_i$. W_2 is a weighting constant that amplifies non-smoothness. Unlike for `open` and `closed` windows, each point in a cluster will have a different temperature value. The probability of a point is exactly the same as the probability of the cluster, since every value is sampled from $\mathcal{U}(0, 1)$ with a probability of one. This property enables the proposed distribution to represent a very non-smooth temperature distribution, without



(a) (b) (c)

(d) (e) (f)

Fig. 8.22: (a)-(c): Three original thermal 3D point clouds with temperature values. (d)-(f): The detected windows under different circumstances, i.e., (d) closed, (e) semi-open, and (f) fully open are shown in green.

flattening the probability distribution. Temperature distributions generated with this method are shown exemplarily in Fig. 8.21 demonstrating that the hand designed probability distributions model the characteristics of the energy functions well.

Apart from the probability distribution the normalization constant depends also on the number of 3D points representing a window. In practice the number of points on a window varies based on the resolution of the capturing device and the distance to the window. To overcome this issue the number of 3D points on the window is kept constant by using an octree based subsampling thus enabling the offline computation of the normalization constant. For applications where speed is not an issue the normalization constant can be calculated online allowing to use a changing number of points on the window. However, apart from the obvious overhead in the online case no significant difference has been observed in the final performance.

8.2.5 Results and discussion

The entire pipeline is tested on six scans from the Automation Lab data set. Fig. 8.22 shows examples of the correctly detected windows in the laboratory. The room has one window of the typical window type present at Jacobs University. The data was collected in winter to reduce the interference of the sun on the thermal camera and to have an adequate temperature between room and outside temperature. Open, semi-open and closed windows are correctly detected.

The energy functions are modeled to represent the characteristics of each label, the parameters are set to decrease or increase sensitivity to the features that characterize the specific state. The values of these parameters that dictate the shape of the posterior probability distribution are hand tuned for this experiment. This has the advantage over other machine learning approaches that require thousands of samples for training. Optimizing the free parameters ensures opti-

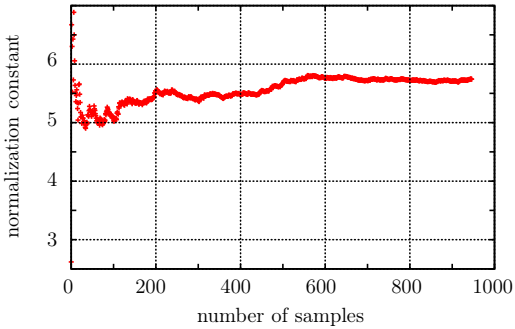


Fig. 8.23: The convergence of the mean sequence; the approximation of the normalization constant of a `closed` window with 1,000 samples.

Table 8.1: Summary of the approximated normalization value and error range.

Label	Number of samples	Normalization constant	Error range
`closed`	1,000	5.79517	0.0333378
`open`	1,000	5.26492	0.0231288
`damaged`	1,000	4.64347	0.0379469

mal performance of the specific energy functions. Determining the MLE (Maximum Likelihood Estimator) from the likelihood function, given a sufficient training data set, or using EM (Expectation Maximization) if there are missing variables from the data set would suffice for an ideal optimization scenario. However, as the normalization function is a function of the free parameters, the difficulty to determine the normalization constant analytically prevents the application of these methods. Alternatives like pseudo-likelihood, where the joint likelihood is approximated by disintegrating it into many spatially independent distributions [16] or convolutional neuronal networks, could be used if there was a huge training data set to learn from.

The normalization constant for each label is approximated with 1,000 samples randomly taken according to the designed probability distribution. The parameter k from equation (8.12) that initializes the temperature distribution is set to room temperature. Fig. 8.21(e) shows two samples for each label. Each sample and the corresponding joint probability equation are evaluated and summed up according to equation (8.11). Since the approximation of the mean gets closer to the true mean as $M \to \infty$, where M is the number of samples, the error is estimated with the variance of the following sequence V_h, that gets smaller and smaller as $M \to \infty$, cf. Fig. 8.23:

$$V_h = \sum_{i=1}^{h} \frac{b(t_i, \theta)}{h\tilde{p}(t_i)}, \tag{8.16}$$

where h goes from 1 to M. As shown in Table 8.1 the error range of the normalization constants Z for each label is very low. The experimentally determined parameters α_1, α_2, α_3, and T are

Table 8.2: The value of free parameters for each labels energy function.

Label	α_1	α_2	α_3	T
`closed`	0.05	1	N/A	1
`open`	2	1	N/A	1
`damaged`	N/A	N/A	0.1	1

given in Table 8.2. Despite the hand tuning of the free parameters the algorithm performed as expected. Exemplary results achieved with these parameters are detailed in Table 8.3. All examples are correctly labeled by the algorithm. The probabilities for each label and also the probability to choose a wrong label are given. `closed` windows are reliably detected. The probability to mislabel an `open` window is much higher. This is due to the fact that in these cases the non-smoothness increases the probability that the window is `damaged`. This suggests that the smoothness function is not optimal for window labeling. Nevertheless, even in these cases the correct label was chosen. This study is a proof of concept for the window labeling task. However, an extensive evaluation using a larger data set containing windows of different types is necessary. Future work should further assess and try to improve the methods presented here. This includes a thorough evaluation of the performance under changing temperature distributions. Especially transferring the approach to examine rooms with air conditioning rather than heating systems is a goal. Different energy functions should be evaluated in order to improve the reliability of the labeling and a comparison to methods using training rather than modeling for the classification task should be performed. Lastly, extending the processing pipeline to label other heat sources and to detect poor insulation in buildings addresses an open topic.

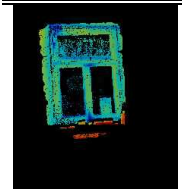
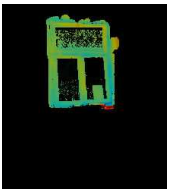
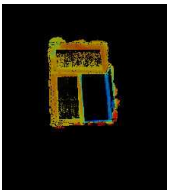
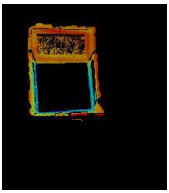
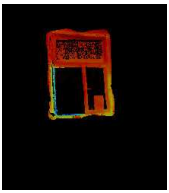
8.3 Spatial projection of thermal data

For some applications the temperature information has to be available immediately and the creation of 3D models is not an option. To overcome this problem we propose a portable system that combines thermal imaging with Augmented Reality (AR) [36]. The idea of the approach is to project the gathered temperature information back into the scene to facilitate visual inspection.

To document the energy efficiency of existing buildings and for quality management during the construction phase for new buildings 3D models are a means to study the thermal characteristics of the building thoroughly. For other applications, where blocked pipes are detected or for the maintenance of electric systems and air conditioning technology the detection speed might be the key to ensuring safety. During manufacturing processes the heat signatures of parts changes rapidly, thus a timely availability of the data is crucial.

The term Augmented Reality refers to enhancing a user's view of the environment with additional computer-generated graphical information. Those virtual additions need to be seamlessly integrated into the environment, that is appear at the right location and with the right orientation, and be updated in real-time in order for the enhancement to be perceived as realistic. Different methods exist to achieve this augmentation: graphics are rendered into video streams on stationary monitors or hand-held devices like smartphones and tablets, or are presented on a head-mounted display integrated into a helmet. While these approaches allow for a realistic

Table 8.3: A summary of experimental windows labeling and probability of making an error

segmented window	Probability of `closed`	`opened`	`damaged`	making error	**Final Label**
	0.642699	0.303224	0.054077	0.357301	**Closed**
	0.731546	0.1543682	0.0247715	0.15684535	**Closed**
	0.308042	0.423237	0.1568721	0.576763	**Opened**
	0.271198	0.477311	0.251491	0.522689	**Opened**
	0.347431	0.466858	0.185712	0.533143	**Opened**

three-dimensional enhancement, they also limit the user's view to the screen of the augmented device, or require him to carry/wear the display hardware which may be cumbersome and might hinder him from performing his intended work. Furthermore, the enhancement is limited to those carrying the device.

Spatial Augmented Reality (SAR) [18] takes a different approach to realize those visualizations: projectors are employed to show the augmentation data directly in the user's surroundings. The user is freed from carrying any additional hardware and sees the data directly where he needs it: on the objects in his work environment. There is no need to (mentally) transfer the information from a visualization device to the real environment. While this approach limits augmentations to the setup area of the projector, multiple projectors can be used to increase the working range, or a portable projection system can be used, to change the display area to the need by hand.

In [36] we proposed a system that combines thermal imaging with AR similar to the one recently presented by Palmerius and Schönborn [159]. By co-calibrating a thermal camera with a projector the visual representation of the temperature information is directly reprojected into the scene. This enables the user to perform visual inspection on manufacturing processes, to examine the run of electric systems or pipelines and to localize thermal leaks in buildings more easily. To account for errors due to the different perspective of the thermal camera and the projector a 3D camera is integrated as an additional component. The system is modular, as the components are interchangeable, and portable, as each individual component is mounted on a tripod. With only a few steps the system can be set up in any location. Compared to [159] it is easier to set up and covers a larger area, i.e., several meters. By using a pico projector it is technically possible to create a mobile system that works similar to a flashlight. However, current handheld projectors are limited in luminance and tracking is limited in accuracy, restricting the application areas of such a system. The following describes the general workflow and the necessary steps to set up a system for spatial projection of thermal data for visual inspection. The approach is evaluated based on a small test scenario.

8.3.1 Spatial Augmented Reality

Spatial Augmented Reality has been employed in a number of different application areas, ranging from prototyping [162] and design [133], museum installations [169], maintenance, medical imaging, training, to operation and control of CNC machines [155] or manipulator robots [126]. To confine this overview, it is limited to systems in combination with a depth camera, and some used for inspection. A system that combines projection based graphics with 3D-sensors is the RoomProjector/SLAMProjector by Molyneaux et al. [137]. Similar to our system, they use depth cameras to acquire depth data of the projection surface. Their projection system is handheld, i.e., it can be moved while using it. The movement of the projector is tracked with either multiple stationary depth cameras or a depth camera fixed to the projector, respectively. While offering great flexibility in choosing the augmented area, tracking accuracy of the handheld projector is limited in the first case, and prone to suffer from drift in certain cases in the second. This limits the achievable accuracy of the projected data, making this type of portability not ideal for our visual inspection purposes. The Beamatron system by Wilson et al. [203] combines a depth sensor with a projector mounted on a pan-tilt platform. While this steerable display

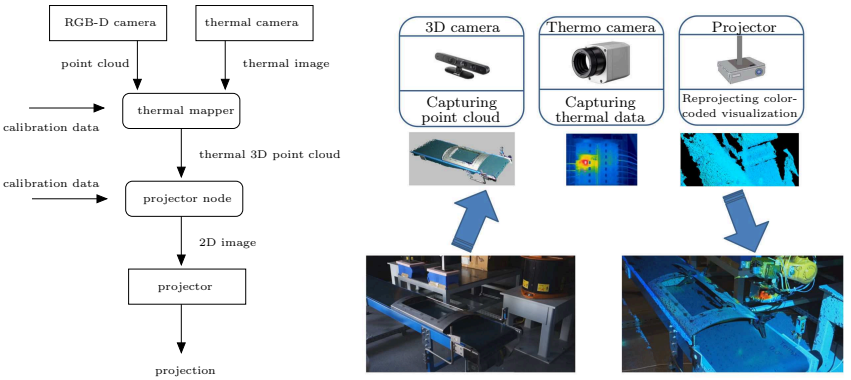


Fig. 8.24: Data flow diagram of the system.

setup is more complicated from a hardware point of view, it allows shifting the projection area and even tracking a moving user with high accuracy. Wilson uses the depth data for rendering of perspectively correct 3D objects (requiring constant tracking of the user's position) as well as an environmental physics model, and allows speech and gesture guided interaction with the environment. An SAR system used for quality inspection of spot welding points in automotive manufacturing was presented by Zhou et al. [219]. The system allows to easily locate welding spots on the real car body by highlighting them with several SAR visualization options, avoiding the necessity to locate them on a technical drawing or monitor and to transfer that knowledge to find the corresponding spots in the work surroundings. Olwal et al. [155] use SAR in combination with a see-through glass panel with CNC-machinery. Real-time process data or occluded tools are made visible to the operator, allowing inspection and a better understanding of the process, and simplifying operation of the mill. Leutert et al. [126] use a SAR-system to display data in the workcell of an industrial manipulator. The processing path and the logged data are visualized directly on the workpiece, making the expected processing result visible before execution and allowing for easy inspection and modification with relation to the real work environment.

8.3.2 System overview

The system for spatial projection of thermal data for visual inspection consists of three main hardware components. The system is used to generate the Projector data set. A thermal camera perceives the infrared radiation emitted from observed objects that corresponds to their temperature. A depth camera records the 3D geometry of the scene while the projector projects the observed information back into the scene to make the temperature values visible to the user. Each component defines its own coordinate system. The essential point is to determine the transformation between these coordinate systems and to transform the data accordingly. Fig. 8.24 shows the data flow in the system. The system is designed to be portable. Thermal

camera, depth camera and projector are each individually mountable on tripods. Alternatively they can be fixed in a working environment. After each setup a one time calibration is necessary to calculate the new coordinate system transformations. Afterwards the system runs continuously. The central link is the thermal 3D point cloud. It is linked to the global coordinate system. For ease of implementation in our test scenario the global coordinate system is identical to the coordinate system of the depth camera. However, integrating a given global coordinate system of the working environment requires only one further transformation. The depth camera acquires point clouds and the thermal camera acquires thermal images. These are combined into the thermal 3D point cloud by the thermal mapper node. The projector node uses the calibration data for the projector to transform the point cloud into a 2D image that is projected into the scene. The following describes first the different calibration processes and then the workflow during runtime.

8.3.3 Calibration

Combining the individual hardware components of the system requires knowledge of their geometric properties. This is achieved by geometric calibration. The calibration consists of two parts, namely intrinsic and extrinsic calibration as described in detail in Chapter 7. The system for spatial projection of thermal data consists of three hardware components. The thermal camera and the projector need to be intrinsically calibrated. As depth cameras are designed to measure geometric properties they are already geometrically calibrated. For the extrinsic calibration both the projector and the thermal camera are calibrated to the depth camera. The intrinsic calibration of the thermal camera and the extrinsic calibration between thermal camera and depth camera are performed according to the methods described in Chapter 7. Using an actively heated pattern dispenses of requiring any special environment conditions. This is ideal for a portable system.

As is fairly common in SAR systems, the projector in our system is modeled as an (inverse) pinhole camera. This allows for applying the same well-researched algorithms for calibration and image formation as are used in camera systems. The input required for determining the internal and external calibration parameters of the device is a number of 2D/3D point pairs. In case of the projector this specifies where the light ray of an illuminated 2D pixel hits the corresponding 3D point in the environment. Since the internal parameters of a projector need to be determined only once, a more elaborate but precise method for calibration was chosen here. The point pairs were acquired by fixing and measuring an electronic board with light sensitive photo resistive elements on a precise positioning device, an industrial robot with absolute positioning error $< 0.1\,\text{mm}$. The light sensor was moved to different positions in the projection volume of the projector, and the corresponding 2D pixel illuminating the photo resistor at that position was determined using binary search in the projector image. This method of point pair determination requires specialized hardware. If not available, other, more easily realizable approaches to calibrate the projector might also be used, for example using a pre-calibrated camera [115].

Several different approaches already exist to calibrate the extrinsic parameters between a camera and projector pair. Those involve, for example, using a pre-calibrated camera to find correspondences for camera and projector in world coordinates of some calibration object. Other approaches are to adjust a projected pattern iteratively until it matches a printed one, or to find

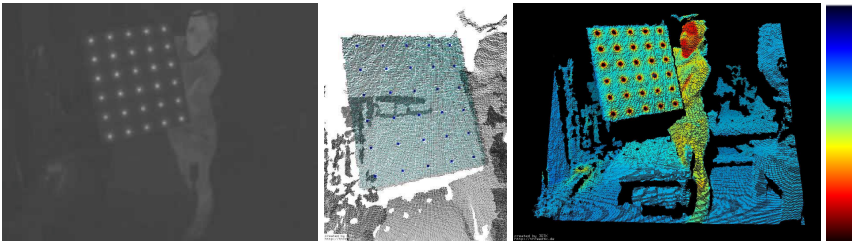
a homography transformation between a calibration plane and the projector image plane. Our system uses the approach proposed in [141], where corners of a chessboard pattern are detected in the camera and projector image, and local homographies for each detected corner are computed. Using a printed chessboard pattern increases complexity of the calibration procedure slightly, but allows to account for non-linearities in the optics of the projector and results in high accuracy calibration data. During this procedure, images from the RGB-D camera are used to determine chessboard corners and to capture structured light illumination from the projector. This allows to detect the pixel location of those corners with subpixel accuracy from the camera as well as from the projector, resulting in high quality input for the calibration algorithm that computes the transformation data between projector and RGB-D camera. An alternative approach for the extrinsic calibration uses again the industrial robot. The projector is extrinsically calibrated to the industrial robot. Measuring the corners of the calibration pattern for the thermal camera with the industrial robot reveals the transformation between the robot. Again, this procedure requires additional hardware and manual control of the robot.

8.3.4 Thermal projection pipeline

After calculating the calibration parameters the projection pipeline as depicted in in Fig. 8.24 is straightforward. Given the calibration of the thermal camera with respect to the RGB-D camera the point cloud is projected onto the image plane to determine for each point the corresponding temperature value. The point cloud is then transferred to the projector node where each point is transformed into the projector coordinate system. To create a 2D image each point is projected onto the projector plane. On each pixel a z-buffer is applied that removes all points except the closest point accounting for occlusions in the environment from the perspective of the projector. For those pixels containing a point the color is calculated from the temperature value. Humans typically associate colors in the red spectrum to warm temperatures and colors in the blue spectrum to cold temperatures. Based on this association an adjustable color scale is developed, as depicted in Fig. 8.25(d). For the point cloud seen in Fig. 8.25(c) the temperature limits are set to $10^\circ\,\mathrm{C} \ldots 40^\circ\,\mathrm{C}$. Pixels without a point remain blank. The resulting image is sent to the projector output. This procedure is repeated in real time to show dynamic changes in the environment continuously.

8.3.5 Combination with a tracked input device

In order to not only see the temperature data visualized in the environment, but also take precise measurements at specific spots of interest in the work area (as with a probe) the operator needs to be able to specify the 3D coordinates of the desired measurement spot. For this purpose, different tracking systems and input devices could be used, ranging from simple tracking of gestures up to using commercially available high-precision tracking systems. In the system presented here we used the already available pen tracking system from [126]. For RGB-D cameras skeleton tracking is already available. Another possibility would be to build upon this by using the operators hands as system input. This option comes with no additional hardware but is limited in terms of accuracy. The specified probe coordinates must then be transformed to the coordinate system of the point cloud which is easily possible, since all involved systems have been calibrated



(a) (b) (c) (d)

Fig. 8.25: The generation of the thermal point cloud. (a) Thermal image in gray scale (white $\hat{=}$ 0 °C, black $\hat{=}$ 100 °C). (b) The calibration procedure. The model (cyan) is fitted to the data to determine the position of the light bulbs (blue). (c) The points are colored based on the scale from 10 °C to 40 °C shown in (d).

beforehand.

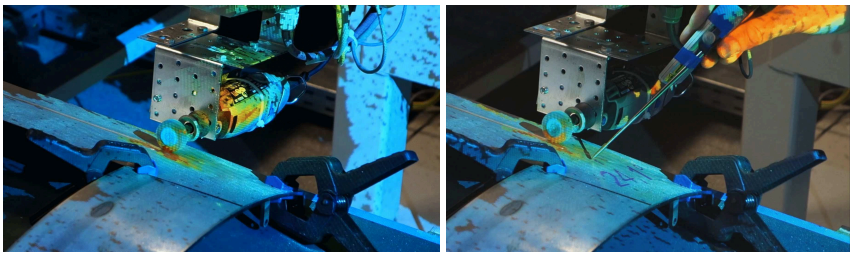
For the global probe coordinates, the closest point in the acquired point cloud of the environment is determined. If point data is available and sufficiently close to the coordinates indicated by the user, the precise temperature and spot the data is taken from is marked and displayed in the projection image. By highlighting the measurement spot again the user can be certain of where precisely a temperature reading is taken from, which might be important if a less precise tracking system were to be used. This allows the user to not only see the general heat distribution in the environment, but also take precise numerical readings at certain critical spots, and enhances the functionality of the system significantly.

8.3.6 Results and discussion

The proposed approach is evaluated using the small test case described in section 3.8. For a first evaluation, a small processing task was set up. The industrial robot was programmed to polish a metal workpiece. This workpiece, a metal oven door, was fixed in the work cell. For the actual processing, a small grinding tool was mounted onto the manipulator, and a simple processing path specified for the robot. The heat inspection system was set up and calibrated to capture the work execution.

During the evaluation runs, the developing heat when the tool was in contact with the workpiece, grinding the metal surface, was clearly made visible as seen in Fig. 8.26(a). Due to a relative high amount of pressure during processing, a lot of strain was put on the electromagnetic motor of the small processing tool, resulting also in a high internal temperature, which could readily be seen in the reprojection. Precise temperature readings were taken using a tracked probe system as described earlier. The increase in temperature at the grinding spot, although small due to the limited size and power of the employed processing, and subsequent cooling down of the metal could be visualized and measured numerically (cf. Fig.8.26(b)).

In a second experiment the spatial accuracy of the reprojection system is tested and visualized.



(a) (b)

Fig. 8.26: (a)Display of temperatures during processing: the heat developing on the surface due to the grinding, as well as in the motor of the tool becomes clearly visible. (b) Measurement of surface temperature of the metal workpiece shortly after processing, using a tracked probe to specify the measurement point.

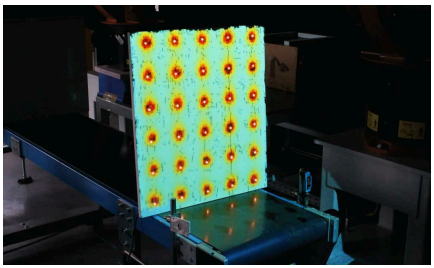


Fig. 8.27: The thermal reprojection of the temperature distribution of the calibration pattern for the thermal camera. It becomes clear that the area around the small light bulbs is heated up.

The wooden calibration pattern featuring small light bulbs in a regular distance grid was placed in front of the system. Since the wood stays relatively cool compared to the light bulbs, the measured heat distribution around the light centers, and thus the spatial accuracy of the thermal inspection system could be visually verified as exemplarily shown in Fig. 8.27.

From 32 frames of a video recording featuring the board at different positions between 164 cm and 281 cm from the depth camera, the light bulbs and their projection were manually selected. After applying a perspective transform to deskew the pattern an average reprojection error of 11.198 mm was computed. Analyzing the results revealed a small systematic error. As the default calibration between the depth and the color image from the RGB-D camera was used in this experiment the overall spatial accuracy is expected to increase when applying a custom refinement of the calibration.

The experiments showed that the system is capable of projecting the thermal information into the scene facilitating visual inspection. In both experiments the system had to deal with

dynamic changes in the environment. An average frame rate of approximately 13 Hz was suitable to visualize these changes in real time.

8.4 Radiometric alignment of 3D point clouds

When adding color information to 3D laser scans by projecting photos of an additional camera to the 3D scans the capturing process is time consuming and therefore prone to changes in the environment. Commonly several pictures per scan are required to cover the field of view of the laser scanner. Besides dynamic objects in the scene the appearance of the colored point cloud is mainly affected by changes of lighting conditions and corresponding camera settings such as exposure and white balance of the captured images.

The RGB values of overlapping images are related to each other by the scene radiance. For short time intervals e.g. between two consecutive images, the scene radiance is assumed to be constant, so color changes are caused by the imaging system. The vignetting, described in Section 2.2.6, causes a radial decline towards the edges of the lens, which is usually modeled by a vignetting term. The camera response function models non-linear processes in the camera. The two components are determined either by pre-calibration or simultaneously during the image alignment and describe the relation between the radiance and the corresponding image point. At the latest when combining several colored 3D scans collected over a period of minutes, hours or even days in an uncontrolled area, the environmental conditions and thus the radiance within the scene changes. For instance Fig. 8.28 depicts two images of a wall from different perspectives and the corresponding point cloud from the Ostia data set. The scan positions are marked in the point cloud. The large exposure difference between both images, that were captured on different days, is caused by the different lighting conditions, i.e., the position of the sun, clouds and the different orientation of the camera with respect to the sun.

In case of panorama images these exposure variations are typically corrected by radiometric aligning the input images to each other. In this section an approach is presented that adopts existing methods for panorama optimization to correct the coloring of point clouds [124]. To this end corresponding pixels from overlapping images are selected by using geometrically closest points of the registered 3D scans and their neighboring pixels in the images. Utilizing the capabilities of most contemporary consumer cameras to store images in a raw format allows for subsequent correction of large exposure differences due to the higher dynamic range of the camera in comparison to 8bit RGB images.

8.4.1 Related work

A few approaches have been presented for color correction in point clouds. Kanzok et al. correct color locally while rendering images of a point cloud [113]. They benefit from the fact that, while rendering, several pixels are projected onto the same pixel. Using all points that fall into one pixel with the same depth value they average over the luminance channel within the HSV color space. Hue and saturation remain unchanged, even though, they are prone to variation if the data was collected over a longer time period. Taking neighboring pixels into consideration by applying a Gaussian filter smoothes the image locally but has no global effect. For the problem at hand this method is not applicable without loss of information, as the resolution of the photo

Fig. 8.28: Point cloud of a wall from the Ostia data set (above). The images used for colorization (below) are not radiometrically aligned so the differences in exposure are clearly visible. The red line in the point cloud connects the two scan positions.

camera is often much higher than the resolution of the point cloud. A two-stage approach for color texture fusion on triangle meshes of range images is presented by Agathos and Fisher [3]. In the global step they determine point correspondences in the overlapping range images and compute a linear transformation matrix in RGB space to transform the color of one image to the other, thus eliminating global discontinuities between the images. In a local step the boundary areas are smoothed by diffusing the color difference from the outline of the boundary towards the interior.

For panorama stitching and high dynamic range imaging radiometric alignment of images is a much more studied problem. To compute seamless high dynamic range panoramas Eden et

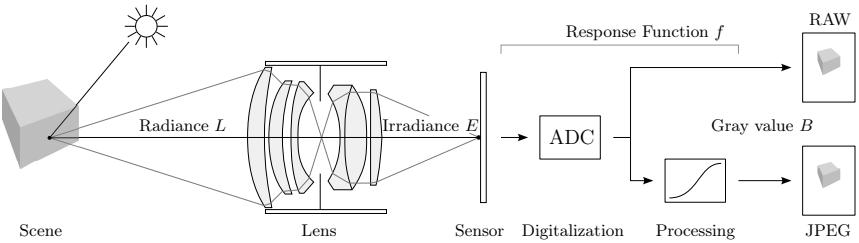


Fig. 8.29: Simplified model of radiometric image formation.

al. [60] radiometrically align the input images based on radiance maps. Using the inverse response function of a pre-calibrated camera and geometrically aligned images, they compute the radiance value at each pixel in overlapping areas and solve for the difference in white balance between the images. While determining the white balance gains between radiance maps of overlapping images vignetting is not considered. An alternative to computing radiance maps is presented by d'Angelo [51]. A gray value transfer function is estimated from corresponding point pairs that allows to recover the corrected exposure values, the camera response function and the vignetting parameters. A simple and fast method is to model the exposure differences between two images as well as the vignetting by a polynomial, thus avoiding to use the radiance space [57]. By using only additive terms the solution is obtained from linear regression. Further images are consecutively aligned to already corrected images. No global optimization is applied to account for summation of small errors in this pair-wise approach.

Color correction of point clouds and panorama images is a similar task. However, the individual images for panorama stitching are typically acquired by simply rotating the camera as large positional offsets cause errors in the image. Thus the effects from changes in perspective are negligible while they play an important role when capturing point clouds for occlusion free models. Thus following an approach similar to [3] spatial proximity in the 3D model is used to find corresponding pixels from overlapping source images. These correspondences are radiometrically aligned exploiting the dynamic range of the raw images produced by the camera.

8.4.2 Image formation

For eliminating color artifacts such as seams and speckled patterns from overlapping image parts by globally aligning the images radiometrically a transformation has to be found that maps the color of one image to the color of the other image. This requires knowledge about the relationship between a point in the scene and its intensity in the image and about the relationship between points in overlapping images. The basic principles of radiometry are explained in Section 2.2.8.

Fig. 8.29 illustrates the image formation process in a simplified model. A point in the scene is lit by a light source and reflects the light towards the lens with radiance L. L is constant but depends on the angle under which the object point is seen, except for Lambertian surfaces which reflect light diffuse. The amount of light passing the lens and hitting the sensor with irradiance E

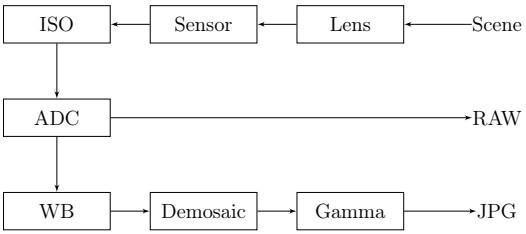


Fig. 8.30: Typical processing steps in the image formation.

is mainly affected by the aperture k and vignetting, a radial fall off. Due to the geometry of the lens the irradiance depends on the angle between the ray and the optical axis. In simple lenses the relation between E and L is often modeled as the $\cos^4(\theta)$ law from equation (2.21). A more general approach that also includes a fall off from other sources, models the spatial variation as a function $M(\bar{\boldsymbol{p}})$ with respect to the coordinates of an image point $\bar{\boldsymbol{p}}$, such that equation 2.21 generalizes to

$$E = M(\bar{\boldsymbol{p}})\frac{1}{k^2}L.$$

A common approach proposed in [89] models M as a sixth degree even polynomial

$$M(r) = 1 + \alpha r^2 + \alpha_2 r^2 + \alpha_3 r^6$$

where r is the distance with respect to the optical center.

The exposure H results from integrating E over the shutter time t at the sensor:

$$H = Et = eML,$$

where $e = \frac{t}{k^2}$ represents the exposure settings. The sensed signal is amplified and digitized and either stored in a raw image format or further processed before storing in a JPEG image. In case of raw images, the further image processing is postponed to a later time.

Fig. 8.30 sketches the major image processing steps, namely white balancing, demosaicking and dynamic range compression. White balancing refers to reducing the hue from colored light by balancing the color channels to move the white point of the image to pure white. Most often camera sensors use a color filter array in order to gather RGB information. Each pixel of the sensor provides only information for one channel, so the missing values have to be interpolated from neighboring pixels. Further image processing steps include a transform to a standardized color space and compression of the dynamic range. To boost the appearance of the images cameras often perform additional built-in image processing which is not further discussed here.

The aforementioned processing steps are camera specific and often non-linear and non-reversible. The relationship between the sensed exposure E_d and the final intensity I in the image is modeled by a camera response function f:

$$I = f(E_d) = f(eML).$$

Most commonly f is monotonically increasing and thus the inverse response function exists, which is used to compute the scene radiance. The relationship between two corresponding image intensities I_1 and I_2 regarding the radiance L is described by the brightness transfer function τ [51]:

$$I_1 = \tau(I_2) = f\left(\frac{e_1 M_1}{e_2 M_2} f^{-1}(I_2)\right). \tag{8.17}$$

For handling color images either a separate response function for each channel is maintained or a single response function is used with an additional weighting factor w for the white balance. Equation 8.17 then becomes

$$I_1 = \tau(I_2) = f\left(\frac{w_1 e_1 M_1}{w_2 e_2 M_2} f^{-1}(I_2)\right). \tag{8.18}$$

8.4.3 Radiometric alignment of point clouds

Radiometrically aligning a colored point cloud consists of two subproblems, aligning the photos collected at one scanning position and aligning the images from different scan positions. For the first problem the rotating scanner acts as a tripod as in the collection of panorama images. With a large rotational and a small translational movement between images the radiance of the overlapping images within the scene remains theoretically stable. For the second problem large changes in perspective and the time difference lead to changes of radiance between two corresponding points in different images. Assuming scenes with mostly Lambertian surfaces, however, the observed radiance remains approximately constant. As L is proportional to E the differences in radiance are interpreted instead as varying exposure settings, i.e., factors for shutter time and white balance.

Fig. 8.31 gives an overview of the processing steps for generating a color corrected points cloud, assuming that the extrinsic calibration between laser scanner and camera is solved, i.e., that the relations between laser point and image pixel coordinates are known. As the built-in JPEG-conversion of a camera contains processing steps such as automatic exposure correction that are hard to model and therefore not reproducible the color correction is carried out on RAW images using the batch processing function of UFRaw [81] which allows to adjust the brightness of an image in aperture stops (EV). The inital step converts the raw images with the minimal required steps to reduce the impact of the non-linear image processing and to ensure an identical response function throughout the processing pipeline. The next step selects a set of point correspondences from the point cloud as described in more detail in Section 8.4.4 and retrieves the color information from the raw images based on the extrinsic calibration.

Using these correspondences the optimization step aligns the images radiometrically using the method from [51] by minimizes the distance

$$\varepsilon = \mathrm{dist}(I_1 - \tau(I_2)) \tag{8.19}$$

between an image intensity I_1 and the transferred corresponding intensity I_2 using the method of Levenberg-Marquart. By estimating the function τ the exposure values and white balance factors of the aligned images, as well as the camera response function are recovered. The capability

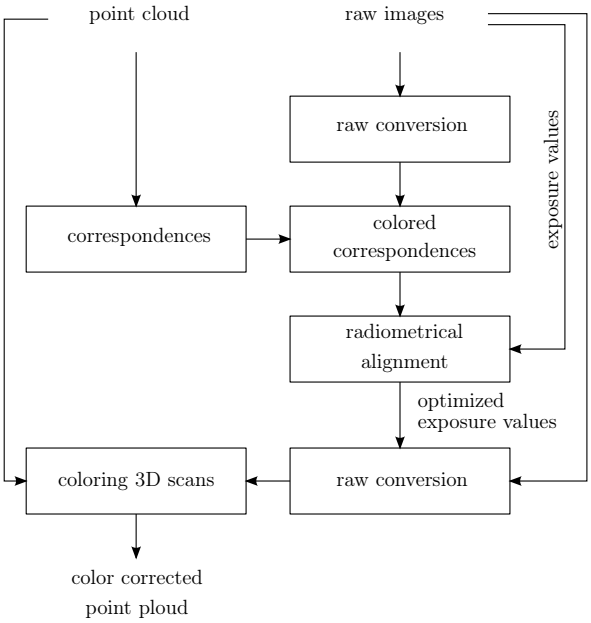


Fig. 8.31: Framework for radiometrically aligning point clouds.

of the method by d'Angelo [51] to estimate the vignetting simultaneously was not considered as the prerequisite of constant radiance does not hold true for multiple scans with large perspective displacement and time offset. Pre-calibrating the specific camera lens combination solves the vignetting integration for this scenario. Differences in the white balance are considered by introducing an additional white balance scaling factor w for the red and blue channel of each image in the function τ in equation 8.17. Note that this white balance factor differs from the white balance of the camera. As demosaicking algorithms expect the raw image to be white balanced UFRaw applies the white balancing first. Consequently adjusting the white balance factors of the camera in RGB space requires to apply the raw conversion for each image in each iteration. Applying a second white balancing to the interpolated RGB channels instead reduces the computational costs.

The EXIF data of an image stores information about the camera settings. Thus the initial guess for the exposure value of the images is derived from this information. Selecting one well exposed image manually as reference image the exposure value and white balance of this image

remain fixed while the other images are adjusted to align with the reference image. Finally, with the optimized exposure values and white balance factors the raw images are newly converted and the color corrected point cloud is computed from these images.

8.4.4 Selection of correspondences

When radiometrically aligning images during panorama generation typically corresponding pixels are sampled from the overlapping areas after aligning the images geometrically using feature based registration (cf. Chapter 5.3). In our scenario the 3D coordinates of each pixel are known from camera to scanner calibration and scan registration (cf. Chapter 5 and 7). Each 3D point from the laser scanner is assigned to one image pixel. Points that are projected onto more than one image yield directly the correspondences for overlapping images from one scan position. To find point pairs in overlapping images from different scan positions the nearest neighbor search finds the closest points based on euclidean distance below a threshold, e.g. 0.5 mm, in the kD-tree representation of the scans.

The radiometric accuracy of the selected points is mainly affected by three factors. First, the resolution of the camera and the laser scanner vary. When the pixel density of the camera images is higher than the point density of the laser scan the area surrounding one 3D point is represented by several image pixels. Thus corresponding points are likely not colored by corresponding pixels. Second, a displacement of the image caused by low accuracy of the calibration leads to the selection of geometrically correct but radiometrically incorrect point pairs. Third, inaccuracies in the scan registration cause a mis-selection of correspondences. To reduce the impact of these factors the mean RGB value of a small pixel neighborhood around the point candidate is used. Additionally point pairs where one image point has a high gradient and thus lies on an edge are discarded.

8.4.5 Experiments and results

Results are presented using the Bräutigam and the Ostia data sets. The camera images provide more than 90 million images points per scan, so approximately 5 candidate pixels per 3D point are available for coloring each scan.

During the several hour long data collection phase for the Bräutigam data set in an outdoor environment the changing position of the sun and more importantly clouds obscuring the sun influenced the exposure of the captured images. Fig. 8.32 illustrates the improvements of the radiometric optimization exemplarily. The scattered pattern on the wall in the original point cloud shows the overlapping area from two images and disappears in the corrected image. Furthermore the difference in brightness between the left and the right side of the point cloud is clearly reduced. To analyze this further Table 8.4 gives the correction parameters ΔEV for each source image of the point cloud resulting from optimization. Image 0 of scan 3 is the reference image. While the environmental conditions are quite stable throughout scan 4 and 7, they change rapidly for others. Note, for example, the distance of more than 1.5 EV between consecutive images 4 and 5 in scan 5. The seam marked with red arrows in Fig. 8.32 is caused by the exposure distance of 0.69 EV between the first and the last image of scan 5. It is not visible in the corrected point cloud anymore. As the homogeneity of the texture increases the

Fig. 8.32: Stephanitorzwinger, Bremen, Germany. Details from the point cloud before (left) and after optimization (right) with equal point density. Note the scattered dark pattern on the brick wall, which disappeared after exposure correction.

scattered pattern disappears. The remaining dark areas (exemplarily marked with green arrows) originate from images where these parts are in the shade while the major part of the image is in bright sunlight. Therefore the images are slightly darkened (Table 8.4, image 0 and 1 of scan 7) and the parts in the shade are not consistent with overlapping images. It should be noted here that shadow removal is a different problem not addressed here, as this work focuses on global not local radiation optimization over the entire image.

From the Ostia data set 13 scans were selected that cover the central room and the large hall. These scans were collected over the course of three days at varying times of day as listed in Table 8.5. Exposure settings were adjusted daily before starting the measurements and kept constant throughout the entire day. Due to the open ceiling and windows the changes in environmental conditions and the resulting difference to the optimal exposure settings were strong, occasionally even between consecutive images due to clouds.

From the 13 scans and their 117 source images a total of 111,999 corresponding point pairs are randomly selected for the exposure optimization. The original uncorrected point cloud is depicted in Fig. 8.33 on the left. Similar to the previous data set changes in lighting conditions express themselves in the scattered pattern in regions of equal point density as shown. At image boundaries and at the edges of occluded areas hard seams appear. An example is visible behind the right pillar in the upper images and in front of both pillars in the lower images. Comparing the original point cloud on the left to the optimized one on the right, the improvements are clearly visible. Despite some calibration inaccuracies, over- and underexposed images are adjusted, so the homogeneity in the brightness distribution increases. The scattered pattern does not disappear completely but the texture of the floor becomes visible and the seams near the pillars become less pronounced.

The paintings on the walls appear blurred both before and after exposure alignment. This is due to the fact that the calibration between camera and laser scanner was unfortunately not repeated every day and the accuracy suffered from the transportation of the equipment.

Table 8.4: Exposure value corrections applied to images for radiometric alignment for the BRÄUTIGAM data set.

	Scan							
Img	0	1	2	3	4	5	6	7
0	1.96	-0.25	0.13	0.00	0.05	1.13	0.97	-0.19
1	1.65	0.94	-0.10	-0.13	0.02	0.60	0.84	-0.07
2	1.64	0.88	0.32	-0.02	0.10	0.11	0.35	0.02
3	1.53	0.85	0.30	-0.05	0.28	-0.19	-0.13	0.01
4	1.39	0.60	0.02	-0.14	0.08	-0.07	-0.22	-0.18
5	1.47	0.38	-0.14	0.51	-0.17	1.69	0.03	-0.30
6	1.23	0.66	-0.27	0.64	-0.20	1.79	0.09	-0.13
7	0.87	1.00	-0.32	1.20	-0.19	1.82	0.22	-0.06
8	1.03	1.24	0.22	1.31	-0.08	1.96	0.09	-0.23
9	1.73	1.24	0.15	1.30	0.01	1.82	0.32	-0.21

Table 8.5: Scans from the OSTIA dataset used for optimization with the acquisition day (a for morning, p for afternoon).

Scan	000	001	002	003	004	010	017	018	019	020	025	026	026
Day	1	1	1	1	1	2a	2p	2p	2p	2p	3a	3a	3a

8.4.6 Discussion

This section presents a way of porting the radiometric image alignment from panorama images to point clouds. The main challenge is the selection of corresponding point pairs from the point cloud. In this work geometric closest points are chosen. Increased robustness is achieved by averaging over the neighboring image pixels. In the two scenarios evaluated here the appearance of the point cloud is visibly increased. It is shown that an accurate calibration between the camera and the laser scanner is crucial for the success of this method. Thus to further improve the results the selection of corresponding point pairs needs to be improved. An option is to employ feature based methods. To simplify the optimization problem the process could be separated into two stages. The first stage aligns the images from one scan radiometrically while the second stage considers each scan as one image and aligns only the differences between the scans.

8.5 Structure from Motion

Up to this point all methods in this thesis assume that the 3D geometry was acquired with a dedicated 3D sensor. These devices compute the 3D geometry of a scene directly but come with certain disadvantages. 3D laser scanners come at high prices while structured light systems are limited in range. Thus methods have been developed to recover the 3D geometry of a scene from photos using image features such as Sift features explained in Section 4.2.

For analyzing thermal data the availability of 3D information is a big advantage. It en-

Fig. 8.33: Ostia Antica. Details from the point cloud before (left) and after optimization (right). Over- and underexposed images are aligned radiometrically thus increasing the homogeneity in the brightness distribution. The point density is identical for all images.

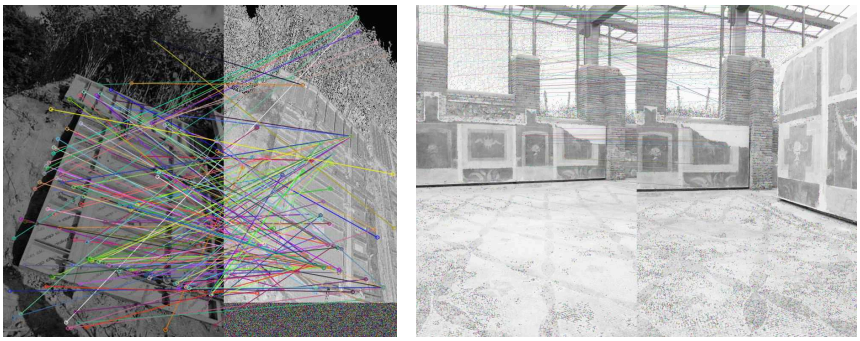


Fig. 8.34: Sift feature matching, the lines connect the best feature matches according to the descriptor matching. Left: Comparison of Sift features in the Bräutigam data set between a photo as gray-scale image and the intensity image of a laser scan. The lines connect the feature correspondences. Right: Two photos from the Ostia data set.

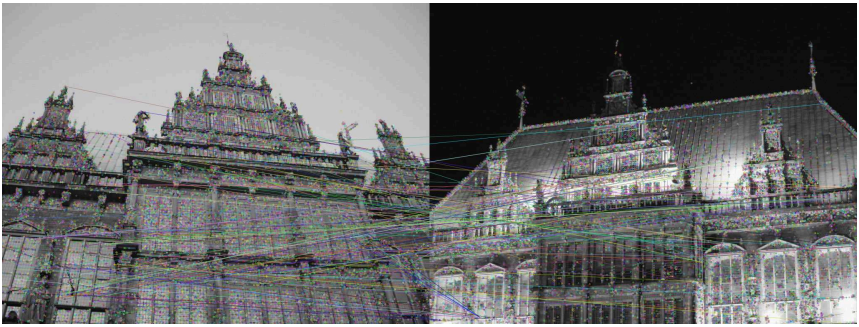


Fig. 8.35: Sift feature matching between day and night images. A similar area of the City Hall from the Bremen City data set is depicted. Among the best feature matches indicated by the lines, are mostly wrong matches.

ables the expert to look at the entire scene on a computer, to measure distances and to draw conclusions. To identify objects in the scene more easily a camera was added to the system in Chapter 6. This is a feasible solution for security related applications but to analyze the energy efficiency of buildings stable conditions are required (cf. Chapter 1). As demonstrated in Section 7.3, especially for the outdoor case, daylight prevents reasonable thermography. Photographs taken in the dark lack detail. To add color information to the 3D data an alternative needs to be found. An intuitive approach is to determine the correspondences to a laser scan based on features in the photos and the intensity images. This was successfully demonstrated on half-timbered houses [23] but lacks practicability in the general case as depicted exemplarily in Figure 8.34. The intensity of the reflected light of the laser scan depends on various aspects as outlined in Chapter 2, including the color, the material and the structure of the surface. Since feature matching is based on the gradient, in general environments the number of false matches exceeds the number of correct matches. Taking images at night and during the day and registering them based on Sift features is also not an option as exemplified in Figure 8.35. Most of the feature matches are wrong. The change in gradients is amplified by the artificial lighting at night. The final solution is to recover the 3D geometry directly from the images.

Assuming an intrinsically calibrated camera and landmarks with known position in the world coordinate system identifying the landmarks in the image allows to estimate the camera pose. This problem, known as the Perspective-n-Point problem (PnP) [119], is used for extrinsic camera calibration (cf. Section 7.1.2). A system of two cameras with known intrinsic and extrinsic calibration allows to retrieve the world coordinates from two images by feature matching. Without known calibration parameters it is still possible to use feature matching to compute the 3D geometry by resolving ambiguities through additional images and features. This procedure is known as Structure from Motion (SfM) or bundle adjustment. Given a set of images, the approach first discovers image features (cf. Section 4.2) in each image. The matching step tries to find for each feature the best matching features in all other images. By comparing several pairs

Table 8.6: From photos to points (SfM)

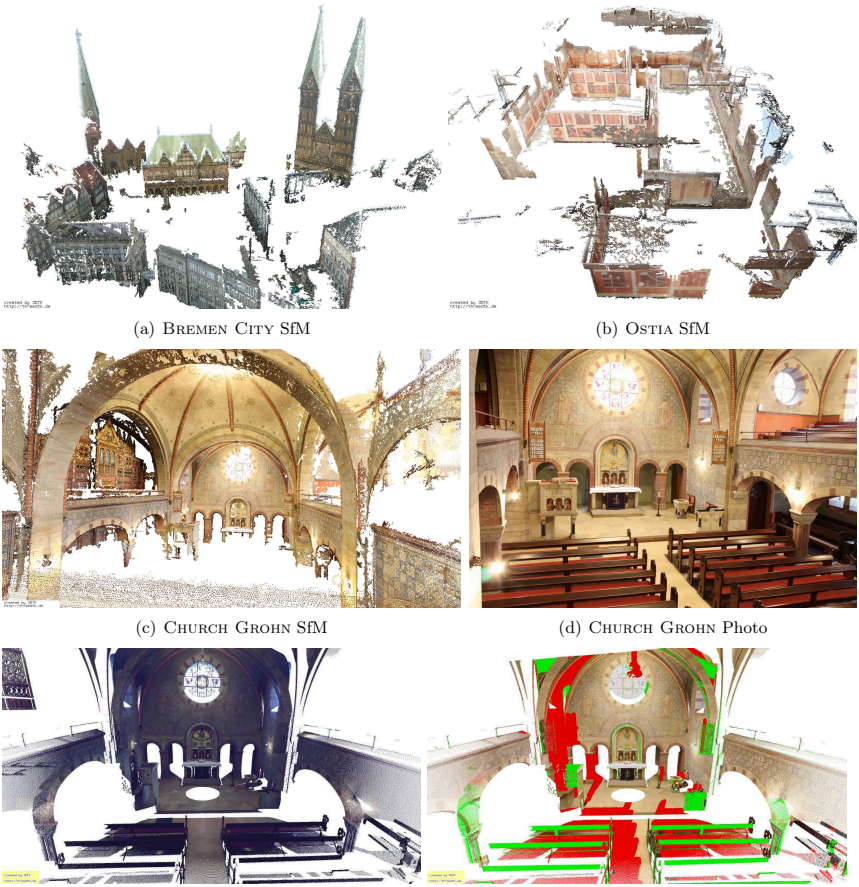
	# photos	# photos used	# points	# select pts	scale factor (m)
Bremen City	143	141	1,313,255	40	4.98
Church Grohn	121	100	4,697,368	6	4.33
Ostia	531	287	8,063,705	25	0.49

a plausibility check rejects outliers among the feature matches. With a RANSAC-like approach possible parameter configurations are determined and a bundle adjustment step minimizes the sum of all deviations between image coordinates and the back-projections of the resulting points.

In this thesis Bundler [184] and VisualSFM [205, 206] are used to generate point clouds using SfM. Both toolchains are based on feature detection using Sift features and incorporate CMVS/PMVS2 for multi-view stereo. These algorithms use the output from the SfM algorithm, cluster views and generate patches around the features to achieve a more accurate and dense reconstruction of the scene [83, 84].

Figure 8.36 shows the dense point clouds generated with the SfM software from the Ostia, Bremen City, Church Grohn data sets. Apart from the missing scale factor the main flaw is the faulty representation of structure-less areas, due to the lack of features. In all three examples the missing floor strikes immediately. The sett pavement in the city center of Bremen has little contrast and it is not surprising that no distinct features are found there. Also by trying to capture as much of the buildings as possible the camera was often tilted upwards thus neglecting the pavement. In the Church Grohn the pews are made of dark wood with little structure. Surprisingly also the mosaic floor in the garden house in Ostia is missing in the reconstruction. Most likely the features are too repetitive to be matched reliably. Applying Sift to two images from the data set shows in Figure 8.34 that no matches on the floor are among the best matches. Noticeably also in the overlapping area in the left image fewer features are marked, due to overexposure. Table 8.6 gives some quantitative information about the models. Almost all photos from the Bremen City data set have contributed to the model. In a wide scene with large opening angle there is a lot of overlap between the images that increases matchability. In the Church Grohn data set there are some smaller passages that lead to occlusions and to a higher number of rejected images. In the Ostia data set the photos were not specifically taken to perform SfM. Instead, the photos collected by the robot were used. To capture the small rooms completely with a minimum number of scans the scanning positions were preferably chosen in corners leading to detailed views of the corners with not much overlap to the other photos. Consequently 46% of the photos were not considered. This is also depicted in the final model where some walls are completely missing, especially the narrow hallways. For comparison refer to Figure 5.8. The protecting roof is also missing. The limited density for all three models is reflected in the number of points that range far below the number of points for a single typical scan.

One approach to solve the scale problem is by iteratively adapting the scale based on measurements of well-distributed reference distances. The results still deviate significantly compared to the the accuracy of a laser scanner [114]. Thus, the following suggests to solve the scale and the registration problem simultaneously by applying a point-based method to determine the transformation between the two models, namely the sICP algorithm (cf. Section 5.4).



(a) Bremen City SfM (b) Ostia SfM

(c) Church Grohn SfM (d) Church Grohn Photo

(e) Church Grohn Internal camera (f) Church Grohn colored point cloud

Fig. 8.36: Point clouds created with the (a) Bundler and (b)+(c) VisualSFM software. (d) A photo from the Church Grohn data set. (e) First scan colored with the internal camera. (f) First scan colored with (d). Green points have normals facing away from the camera. Red points are occluded.

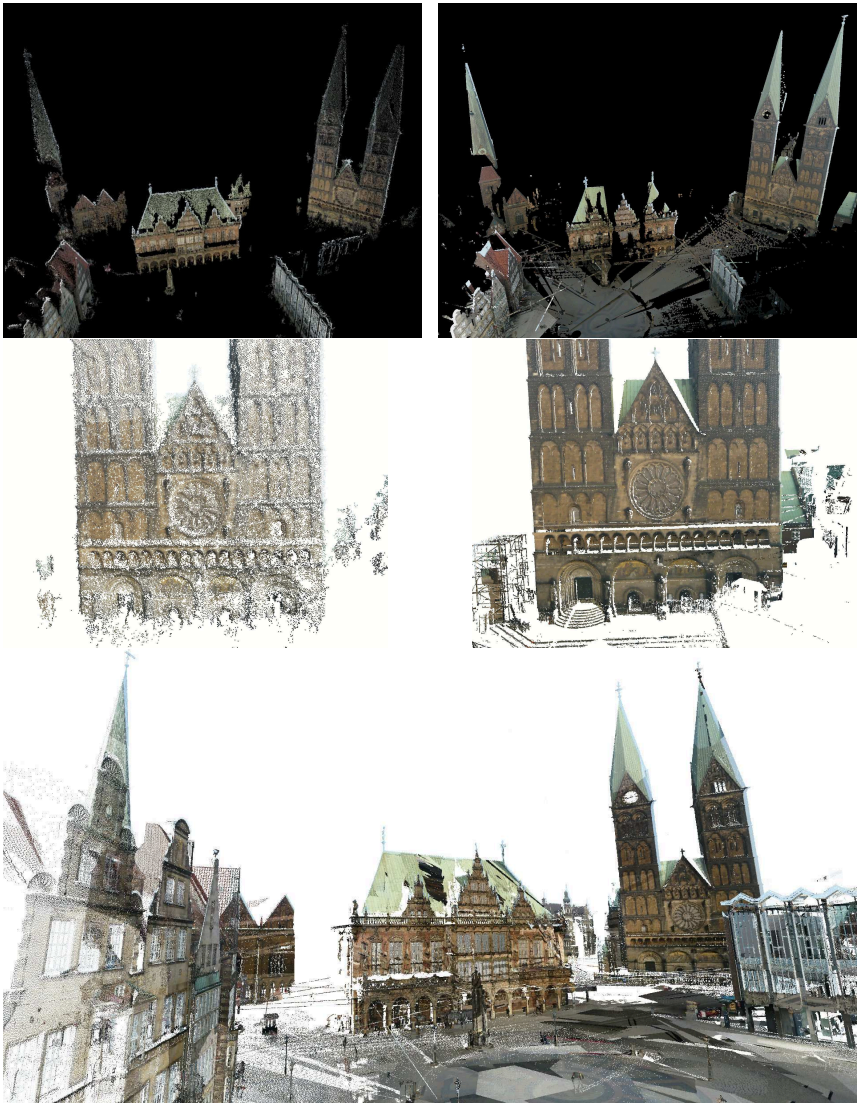


Fig. 8.37: Top: Comparison of the Bremen City models. Left: Results from SfM. Right: colored laser scans. Bottom: Three scans from the Bremen City data set colored with all photos.

As for the ICP the sICP requires initial pose estimates. The large difference in scale prevents the application without any pre-processing. Thus a number of point correspondences are manually chosen for the first iteration of the sICP. The number of correspondences and the resulting initial scale factor is given in Table 8.6. For the final transformation including scale the point correspondences are automatically found based on Euclidean distance. Once the transformation is known it is directly applied to the camera poses allowing to color the scan data. The general approach is identical to the one described in the previous chapter. However, due to the large offset between the camera and the scanner the occlusion check is not sufficient. To avoid coloring surfaces facing away from the camera, an angle check is introduced, similar to the one described in Section 7.2.5. Points are discarded if the angle between their normal vector and the viewing direction exceeds 120°. This is demonstrated in Fig. 8.36(f). Here a single scan is colored with a single photo (Fig. 8.36(d)). The scanning position is visible in front of the altar. Consequently all the pews and parts of the arches are facing away from the camera. The large amount of occlusions is also caused by the different viewing positions. For best results it is recommended to apply the method a complete point cloud model without any holes. Comparing the result to the scan colored with the internal camera of the Faro Focus3D (Figure 8.36) reveals the drawback of the internal camera. In environments with poor lighting the image quality is low.

Figure 8.37 shows the models generated with SfM and a single scan colored with five selected photos from Bremen City in comparison. Two major improvements are clearly visible. First, the density of the new model is significantly higher. It corresponds to the density of the scan which is configurable. The total number of pixels in the photos is typically much higher. This way precise details are preserved. Second, structure-less areas with few or no features are not omitted anymore. An example is the ground of the market square that appears now in the colored point cloud.

Figure 8.37 shows the joint point cloud from three scanning positions in front of the City Hall and the cathedral colored with all photos. The occlusion and the angle check are activated and for each point the photo with the minimum distance is chosen to reduce dynamic obstacles. When taking the photos the distance to the buildings was rather large to get good coverage. This should reduce the risk of ending up with a lot of small spots. Nevertheless, the edges between photos are clearly visible in some areas. On the cathedral spires the automatic camera settings caused the image brightness to change based on the amount of visible sky. The conspicuous patchwork on the ground has two reasons. First, as all camera positions are inevitably on the ground and distributed over the entire area, the closest camera changes frequently. Second, the ground points have the tendency to be on the image edges where small calibration errors have the largest effect. One last issue is caused by occlusions. Due to the small number of scanning positions and the restriction to the ground, holes caused by occlusions are inevitable. The number of camera positions is much higher and thus in some cases a camera position is chosen for an area that is not visible from that position but only appears through a hole in the point cloud. One prominent example is on the roof of the City Hall. Overall the result is good and especially the transformations are computed with high precision. A possible solution to the coloring problem is to compare the color values from all cameras that see a point and to perform radiometric alignment similar to the approach in the previous section. This should also detect wrongly matched images. The results from the SfM toolchain could be used to facilitate this step.

(a) SfM + Calibration (b) SfM (c) Calibration

Fig. 8.38: Comparison between the scan colored based on the SfM result and based on the calibration.

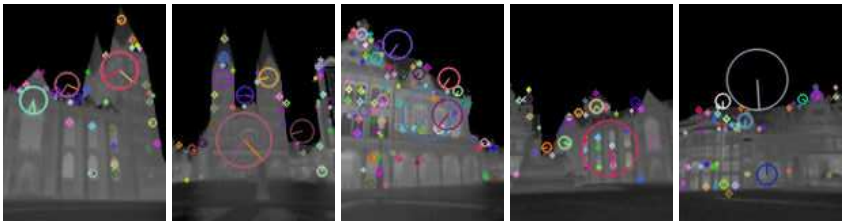


Fig. 8.39: Sift feature examples on 160×120 pixel thermal images.

The last example is the Ostia data set in Figure 8.38. In the examples for radiometric correction in the previous section small calibration inaccuracies were revealed. Here a direct comparison is performed between the first scan colored based on the SfM results and according to the calibration. Only the photos taken at this position were considered. Overlaying both scans gives a blurry image of the wall paintings. Looking at individual scans shows that in the SfM result the images fit the 3D structure of the wall better. The images align better with the top. For the calibration result the seam between the two images to the right is clearly visible. The Ostia data set was the first experiment with that specific setup. The small amount of calibration data comes at the price of low accuracy. Based on this experience larger calibration data sets were used for later experiments.

The experiments in this section show that it is possible to use SfM to generate the 3D information necessary to join camera and laser scanning data. In the examples used here the accuracy was generally satisfactory. The major problem that still remains is the risk of rejected images that cause holes in the data. With the runtime of several hours up to days for the complete toolchain it is difficult to rely only on this data. Furthermore the distance between the camera and the scanning positions causes problems in case of occlusion. To conclude it should be noted that this approach is not applicable for thermal data as the number of features is too low, especially for low resolution thermal images. A few examples from the Bremen City data set are shown in Figure 8.39.

8.6 Summary

Many applications are imaginable that build upon the foundations created by the methods developed throughout this thesis. This chapter successfully exploits some of them. The automatic extraction of semantic information from point clouds is one of the biggest challenges. A first step towards this goal is to identify the most prominent planes from building interiors and to extract the room structure. Even in rooms with large occluded parts the structural elements are correctly classified leading to a plane model of the room. Section 8.2 infers additional semantics by making use of the thermal information. By modeling the characteristics of open, closed and damaged windows explicitly the approach manages to correctly classify the samples into their category without the need of extensive training data.

Combining the 3D thermal mapping capability with a projector introduces the technology into the field of AR. The use of an RGB-D camera allows real time applications. By projecting the thermal information into the scene an operator sees the temperature directly on the object and can react to undesired behavior immediately.

Changing lighting conditions influence the color perception of cameras leading to visible edges and speckles in overlapping areas. Radiometric alignment of the images based on closest point in the point cloud data reduces this effect. In cases where the co-calibration of camera and range sensor is not applicable or where the calibration turns out to be inaccurate SfM offers an alternative. Point clouds generated with SfM suffer from low density and unknown scale but generally preserve the geometry quite well. Combining these two types of point clouds allows to apply the color from the photos to the dense laser scans.

Chapter 9

Conclusions

The work at hand exploits the combination of 3D point clouds with color and thermal camera images. The main focus lies on the co-calibration between a laser scanner and a thermal camera. For this purpose a calibration pattern consisting of a known light bulb pattern on a wooden board is designed that is reliably detected in both the point clouds and the thermal images. The calibration procedure is directly transferable to color images by replacing the light bulbs with chessboard corners. This way the system opens up for applications in other fields such as geology and archaeology. For the calibration pattern detection in the point cloud plane detection methods are evaluated. These methods are also highly relevant for further processing of the data. It is shown that plane detection methods that act on a global scale are ideally suited for detecting the major structural elements of building interiors. For detecting smaller structures, such as the calibration pattern, region growing approaches are preferable.

The sensor data that is combined into one model is acquired using optical sensors. The propagation of light can be described using general geometric rules. To obtain a true representation of the environment the internal geometric properties of the sensors need to be known. This is achieved via calibration. For laser scanners calibration is performed by the manufacturer. Modern cameras are built in a way that reduces distortions but due to the moving parts a user-side geometric calibration is necessary for precise measurements.

The semi-automatic calibration procedure is the essential step for transforming the sensor data into a multi-modal model. The remaining steps rely on 3D point cloud processing. Data structures are presented that enable efficient accessing of the point data. Panorama images are useful representations of single point clouds that reflect the acquisition procedure. They have their strengths when all points need to be accessed and neighborhood relations from individual laser scans are relevant, e.g., for feature-based registration or region growing approaches. For exploiting the full 3D geometry octrees or kD-trees are the preferred data structure. While the octree divides the space in a regular 3D grid, the kD-tree uses a split plane on each level. Considering that a laser scan captures only surface points, the octree contains many empty voxels while the kD-tree adjusts to the distribution of the data. This makes the kD-tree more suitable for nearest neighbor detection and ray tracing. The octree on the other hand is useful for point reduction and efficient occupancy grid representations.

The calibration procedure enables the main contribution of this work. With the known geo-

metric relation between the individual sensors the data from different sources is combined under consideration of occlusions. To create a full 3D model point clouds from different locations are registered into one common coordinate system. The sequential ICP algorithm and a GraphSLAM variant are evaluated to accomplish this task. Depending on the acquisition procedure the initial pose estimates are either determined from GPS measurements, 2D SLAM or feature-based registration.

The step that completes the automatic generation of a 3D model is the exploration. The mobile robot Irma3D is programmed to explore its environment. Initially it creates a 2D map and chooses the NBV position based on the expectancy to gather the most new information. Once an enclosure of the already explored space is detected, the algorithm switches to the 3D exploration mode that tries to find a position from where the most number of unseen voxels is perceived.

The technology was successfully applied to collect data in various scenarios, including archaeological and cultural heritage sites, geological formations, building interiors and cities. Several sample applications that make use of the data were tested. The 3D geometric information of a thermal model allows to remove unreasonable measurements caused by reflections or inappropriate sensing angles. Furthermore, by detecting structural elements of buildings, especially windows, and classifying them into categories such as open, closed and damaged, a first step is done towards automatic analysis of the data. Further applications include the use of AR to project the perceived temperature information back into the scene for inspection purposes, attempts to improve the color representation by radiometrically aligning the point clouds or by incorporating SfM approaches.

The technology for autonomously creating multi-modal 3D maps is extendable into many directions. The primary goal is further semantic analysis of thermal point clouds. A logical step is to detect potential points of interest, such as extrema or discontinuities in the data as suggested in [39]. Implementing a viewer that shows these points of interest to an expert one by one decreases the time needed for analysis. The method used for window classification can be extended to detect and classify other objects. By classifying different machines, pipes and cables in data centers or factories a robot can be applied for monitoring and detecting flaws in the workflow before they lead to failures. The generation of a multi-modal-spatio-temporal model will be considered in this context.

In the current setup the thermal camera acquires single images statically when used in combination with the laser scanner. By using the video streaming capability instead, the thermal camera could be added to aerial vehicles or personal laser scanning systems such as the one presented in [123]. This requires solving the synchronization problem as the scanner rotates with an angular velocity of $60\,^{\circ}/\mathrm{s}$.

Some attempts have already been made to use the combination of RGB-D and thermal camera for capturing 3D thermal videos. This is of interest for capturing the body temperature of athletes. An alternative project for this setup is to integrate a thermal camera into Kinect Fusion [109] to achieve real time generation of 3D thermal models.

Future goals in reconstruction of building interiors are to generate CAD or building information models and to integrate the thermal information into these. In cases where not only planar structures are of interest the thermal point cloud will be transformed into a polygon map with texture mapped to it similar to [170].

The calibration procedure is extendable to other sensors. In the field of geology hyperspectral cameras are used to determine surface materials while multispectral cameras are used for investigating art work. Adding such a device to the system opens opportunities for extended applications.

To determine the transformation between SfM and laser scan models the correspondences were determined manually. Future work will focus on evaluating automatic methods for the combination of the two point clouds [145, 212] and research further methods to achieve a reliable registration.

Bibliography

[1] A. Adan and D. Huber. 3D reconstruction of interior wall surfaces under occlusion and clutter. In *Proceedings of 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, May 2011.

[2] A. Adan, X. Xiong, B. Akinci, and D. Huber. Automatic creation of semantically rich 3d building models from laser scanner data. In *Proceedings of the International Symposium on Automation and Robotics in Construction (ISARC)*, June 2011.

[3] A. Agathos and R. B. Fisher. Colour texture fusion of multiple range images. In *3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on*, pages 139–146. IEEE, 2003.

[4] M. Đakulović, S. Ileš, and I. Petrović. Exploration and Mapping of Unknown Polygonal Environments Based on Uncertain Range Data. *AUTOMATIKA: Journal for Control, Measurement, Electronics, Computing and Communications*, 52(2):118–131, 2011.

[5] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. In *Proccedings of ACM SIGMOD '99 International Conference on Management of Data*, Philadelphia, 1999.

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

[7] Robotic Industries Association. Unimate – the first industrial robot. Webpage, visited May 2016. `http://www.robotics.org/joseph-engelberger/unimate.cfm`.

[8] B. Atcheson, F. Heide, and W. Heidrich. CALTag: High Precision Fiducial Markers for Camera Calibration. In Reinhard Koch, Andreas Kolb, and Christof Rezk-Salama, editors, *Vision, Modeling, and Visualization (2010)*. The Eurographics Association, 2010.

[9] M. Attene, B. Falcidieno, and M. Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer*, 22:181–193, 2006.

[10] AutomationAtJacobs. Automatic Markerless 3D Scanning in Ostia Antica. `http://youtu.be/sf-gq5xlaIc`, August 2012.

[11] AutomationAtJacobs. Mapping Bremen downtown with thermal information. `http://youtu.be/TPoCebERysc`, January 2012.

[12] AutomationAtJacobs. Robot Irma3D creates autonomously a digital 3D model of a special exhibition in the Universum Bremen. `http://youtu.be/p4I-aYEvrTo`, November 2012.

[13] U. Bauer and K. Polthier. Detection of Planar Regions in Volume Data for Topology Optimization. *Lecture Notes in Computer Science*, 2008.

[14] J. L. Bentley. Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517, 1975.

[15] M. Bertalmio, A.L. Bertozzi, and G. Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 01).*, pages 355–362, 2001.

[16] J. Besag. Statistical analysis of non-lattice data. *The statistician*, 24:179–195, 1975.

[17] P. Besl and N. McKay. A Method for Registration of 3–D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239 – 256, February 1992.

[18] O. Bimber and R. Raskar. *Spatial augmented reality: merging real and virtual worlds*. CRC Press, 2005.

[19] C. M. Bishop. *Pattern recognition and machine learning*, volume 1. Springer, New York, 2006.

[20] P. S. Blaer and P. K. Allen. Data acquisition and view planning for 3-D modeling tasks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 417–422. IEEE, 2007.

[21] M. Bleier and A. Nüchter. Low-cost 3D Laser Scanning in Air or Water using Self-Calibrating Structured Light. In *Proceedings of the 9th ISPRS International Workshop 3D-ARCH 2017: "3D Virtual Reconstruction and Visualization of Complex Architectures"*, February 2017.

[22] J. Boehm. Facade detail from incomplete range data. In *Proceedings of the ISPRS Congress*, Beijing, China, 2008.

[23] J. Boehm, S. Becker, and N. Haala. Model refinement by integrated processing of laser scanning and photogrammetry. In *Proceedings of 3D Virtual Reconstruction and Visualization of Complex Architectures (3D-Arch)*, Zurich, Switzerland, 2007.

[24] D. Borrmann, H. Afzal, J. Elseberg, and A. Nüchter. Mutual Calibration for 3D Thermal Mapping. In *Proceedings of the 10th Symposium on Robot Control (SYROCO)* , Dubrovnik, Croatia, September 2012.

[25] D. Borrmann and J. Elseberg. Global konsistente 3D Kartierung am Beispiel des Botanischen Gartens in Osnabrück. Master's thesis, Universität Osnabrück, 2006.

[26] D. Borrmann and J. Elseberg. Deforming Scans for Improving the Map Quality Using Plane Extraction and Thin Plate Splines. Master's thesis, Universität Osnabrück, 2009.

[27] D. Borrmann, J. Elseberg, P. N. KC, and A. Nüchter. Ein Punkt pro Kubikmeter – präzise Registrierung von terrestrischen Laserscans mit Scanmatching. In T. Luhmann and C. Müller, editors, *Photogrammetrie Laserscanning Optische 3D-Messtechnik, Beiträge der Oldenburger 3D-Tage 2012, Jade Hochschule*. Wichmann Verlag, February 2012.

[28] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter. A Data Structure for the 3D Hough Transform for Plane Detection. In *Proceedings of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV '10)*, Lecce, Italy, September 2010.

[29] D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, and J. Hertzberg. Globally consistent 3D mapping with scan matching. *Journal of Robotics and Autonomous Systems (JRAS)*, 56(2):130–142, February 2008.

[30] D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, and J. Hertzberg. The Efficient Extension of Globally Consistent Scan Matching to 6 DoF. In *Proceedings of the 4th International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT '08)*, pages 29–36, Atlanta, USA, June 2008.

[31] D. Borrmann, J. Elseberg, and A. Nüchter. Thermal 3D Mapping of Building Façades. In *Proceedings of the 8th Conference on Intelligent Autonomous Systems (IAS '12)*, Jeju Island, Korea, 2012.

[32] D. Borrmann, J. Elseberg, A. Nüchter, and K. Lingemann. The 3D Hough Transform for Plane Detection in Point Clouds – A Review and a new Accumulator Design. *Journal of 3D Research*, 2(2):1–13, March 2011.

[33] D. Borrmann, R. Hess, D. Eck, H. Houshiar, A. Nüchter, and K. Schilling. Evaluation of methods for robotic mapping of cultural heritage sites. In *Proceedings of the 2nd IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control (CESCIT '15)*, Maribor, Slovenia, June 2015.

[34] D. Borrmann, R. Heß, D. Eck, K. Schilling, and A. Nüchter. Robotic Mapping of Cultural Heritage Sites. In *Proceedings of the 5th ISPRS International Workshop 3D-ARCH 2015: "3D Virtual Reconstruction and Visualization of Complex Architectures"*, Avila, Spain, February 2015.

[35] D. Borrmann, H. Houshiar, J. Elseberg, and A. Nüchter. Vom Kombinieren von 3D-Modellen mit Farb- und Temperaturinformationen. In T. Luhmann and C. Müller, editors, *Photogrammetrie Laserscanning Optische 3D-Messtechnik, Beiträge der Oldenburger 3D-Tage 2013, Jade Hochschule*. Wichmann Verlag, February 2013.

[36] D. Borrmann, F. Leutert, K. Schilling, and A. Nüchter. Spatial Projection of Thermal Data for Visual Inspection. In *Proceedings of the XIVth International Conference on Control, Automation, Robotics and Vision (ICARCV 2016)*, Phuket, Thailand, November 2016.

[37] D. Borrmann, L. Leutert, I. Maurovic, M. Seder, and A. Nüchter. Automatische Grundrisserstellung mittels Laserscandaten. In T. Luhmann and C. Müller, editors, *Photogram-*

metrie Laserscanning Optische 3D-Messtechnik, Beiträge der Oldenburger 3D-Tage 2016, Jade Hochschule, pages 108–119. Wichmann Verlag, February 2016.

[38] D. Borrmann, A. Nüchter, M. Đakulović, I. Maurović, I. Petrović, D. Osmanković, and J. Velagić. The Project ThermalMapper – Thermal 3D Mapping of Indoor Environments for Saving Energy. In *Proceedings of the 10th Symposium on Robot Control (SYROCO)* , Dubrovnik, Croatia, September 2012.

[39] D. Borrmann, A. Nüchter, M. Đakulović, I. Maurović, I. Petrović, D. Osmanković, and J. Velagić. A mobile robot based system for fully automated thermal 3D mapping. *Advanced Engineering Informatics*, 28(4):425–440, October 2014.

[40] N. Boutros, M. R. Shortis, and E. S. Harvey. A comparison of calibration methods and system configurations of underwater stereo-video systems for applications in marine ecology. *Limnology and Oceanography: Methods*, 13(5):224–236, 2015.

[41] G. Bradski and A. Kaehler. *Learning OpenCV, Computer Vision with OpenCV library.* O'Reilly Media, first edition, 2008.

[42] Landesarchäologie Bremen. Ausgrabungen im Stadtbereich Bremen. Webseite, visited September 2017.
`http://www.landesarchaeologie.bremen.de/sixcms/detail.php?gsid=bremen167.c.1646.de`.

[43] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal 4*, 4(1):25–30, 1965.

[44] A. Budroni and J. Boehm. Toward automatic reconstruction of interiors from laser data. In *Proceedings of 3D-Arch*, Zurich, Switzerland, February 2005.

[45] J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), November 1986.

[46] A. Censi and S. Carpin. HSM3D: Feature-Less Global 6DOF Scan-Matching in the Hough/Radon Domain. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2009. (accepted).

[47] P. Clifford. Markov random fields in statistics. *Disorder in physical systems*, pages 19–32, 1990.

[48] Commission of the European Communities. Addressing the challenge of energy efficiency through Information and Communication Technologies. Communications from the Commisison to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions, COM(2008) 241 final, May 2008.

[49] D. González-Aguilera, P. Rodriguez-Gonzalvez, J. Armesto, S. Lagüela. Novel approach to 3D thermography and energy efficiency evaluation. *Energy and Buildings*, 54:436–443, 2012.

[50] T. K. F. Da. 2D alpha shapes. In *CGAL User and Reference Manual*. CGAL Editorial Board, 4.10 edition, 2017.

[51] P. d'Angelo. Radiometric alignment and vignetting calibration. *Proc. Camera Calibration Methods for Computer Vision Systems*, 2007.

[52] P. E. Danielsson and O. Seger. Generalized and Separable Sobel Operators. In H. Freeman, editor, *Machine vision for three-dimensional scenes*. Academic Press, 1990.

[53] G. G. Demisse, D. Borrmann, and A. Nüchter. Interpreting Thermal 3D Models of Indoor Environments for Energy Efficiency. In *Proceedings of the 16th Conference on Advanced Robotics*, Montevideo, Uruguay, November 2013.

[54] G. G. Demisse, D. Borrmann, and A. Nüchter. Interpreting Thermal 3D Models of Indoor Environments for Energy Efficiency. *Journal of Intelligent and Robotic Systems*, 77(1):55–72, January 2015.

[55] M. B. Dillencourt, H. Samet, and M. Tamminen. A general approach to connected-component labeling for arbitrary image representations. *J. ACM*, 39:253–280, April 1992.

[56] A. Doucet, N. de Freitas, and N. Gordan, editors. *Sequential Monte-Carlo Methods in Practice*. Springer Verlag, 2001.

[57] C. Doutre and P. Nasiopoulos. Fast vignetting correction and color matching for panoramic image stitching. In *2009 16th IEEE International Conference on Image Processing (ICIP)*, pages 709–712. IEEE, 2009.

[58] R. O. Duda and P. E. Hart. Use of the Hough Transformation to Detect Lines and Curves in Pictures. Technical Note 36, Artificial Intelligence Center, SRI International, 1971.

[59] R. C. Dumitru, D. Borrmann, and A. Nüchter. Interior Reconstruction using the 3D Hough Transform. In *Proceedings of the 5th ISPRS International Workshop 3D-ARCH 2013: "3D Virtual Reconstruction and Visualization of Complex Architectures"*, Trentro, Italy, February 2013.

[60] A. Eden, M. Uyttendaele, and R. Szeliski. Seamless image stitching of scenes with large motions and exposure differences. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2498–2505. IEEE, 2006.

[61] H. Ekbert and G. Schönfelder, editors. *Sensoren in Wissenschaft und Technik*. Vieweg + Teubner Verlag, Wiesbaden, 2012.

[62] A. Ekman, A. Torne, and D. Stromberg. Exploration of polygonal environments using range data. *Systems, Man, and Cybernetics, Part B: Cybernetics*, 27(2):250–255, 1997.

[63] J. Elseberg, D. Borrmann, and A. Nüchter. Efficient Processing of Large 3D Point Clouds. In *Proceedings of the XXIII International Symposium on Information, Communication and Automation Technologies (ICAT '11)*, Sarajevo, Bosnia, October 2011. IEEE Xplore.

[64] J. Elseberg, D. Borrmann, and A. Nüchter. Full Wave Analysis in 3D Laser Scans for Vegetation Detection in Urban Environments. In *Proceedings of the XXIII International Symposium on Information, Communication and Automation Technologies (ICAT '11)*, Sarajevo, Bosnia, October 2011. IEEE Xplore.

[65] J. Elseberg, D. Borrmann, and A. Nüchter. 6DOF Semi-Rigid SLAM for Mobile Scanning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '12)*, Lisbon, Portugal, October 2012.

[66] J. Elseberg, D. Borrmann, and A. Nüchter. Automatic and Full Calibration of Mobile Laser Scanning Systems. In *Proceedings of the 13th International Symposium on Experimental Robotics (ISER '12)*, Québec City, Canada, 2012.

[67] J. Elseberg, D. Borrmann, and A. Nüchter. Eine Milliarde 3D-Punkte mit Standardhardware verarbeiten – Processing One Billion 3D Points on a Standard Computer. *Allgemeine Vermessungs-Nachrichten (AVN)*, 119(1):11–23, January 2012.

[68] J. Elseberg, D. Borrmann, and A. Nüchter. Algorithmic Solutions for Computing Precise Maximum Likelihood 3D Point Clouds from Mobile Laser Scanning Platforms. *Remote Sensing*, 5(11):5871–5906, 2013.

[69] J. Elseberg, D. Borrmann, and A. Nüchter. One Billion Points in the Cloud – An Octree for Efficient Processing of 3D Laser Scans. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS), Special issue on terrestrial 3D modelling*, 76:76–88, February 2013.

[70] J. Elseberg, S. Magnenat, R. Siegwart, and A. Nüchter. Comparison on nearest-neigbour-search strategies and implementations for efficient shape registration. *Journal of Software Engineering for Robotics (JOSER)*, 3(1):2–12, 2012.

[71] FARO Technologies, Inc. `http://www.faro.com`, visited September 2017.

[72] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24:381 – 395, 1981.

[73] R. B. Fisher. Applying knowledge to reverse engeniering problems. In *Proceedings of the International Conference. Geometric Modeling and Processing (GMP '02)*, pages 149 – 155, Riken, Japan, July 2002.

[74] Inc. FLIR Systems. 5 factors influencing radiometric temperature measurements, November 2016.

[75] FLIR Systems AB. *Thermal imaging guidebook for building and renewable energy applications*. In cooperation with the Infrared Training Center (ITC), Sweden, 2011.

[76] W. Förstner and B. P. Wrobel. *Photogrammetric Computer Vision*, volume 11 of *Geometry and Computing*. Springer, 2016.

[77] N. A. Fouad and T. Richter. *Leitfaden Thermografie im Bauwesen – Theorie, Anwendungsgebiete, praktische Umsetzung*. Fraunhofer IRB Verlag, Stuttgart, 4th edition, 2012.

[78] M. H. Freeman and C. C. Hull. *Optics*. Buterworth-Heinemann, Edinburgh [etc.], 11 edition, 2003.

[79] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An Algorithm for Finding Best Matches in Logarithmic Expected Time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977.

[80] C. Frueh, S. Jain, and A. Zakhor. Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images. *International Journal of Computer Vision (IJCV)*, 61(2):159 – 184, 2005.

[81] U. Fuchs. Ufraw. Webpage, 2014. http://ufraw.sourceforge.net/.

[82] R. Furukawa and H. Kawasaki. Laser range scanner based on self-calibration techniques using coplanarities and metric constraints. *Computer Vision and Image Understanding*, 113(11):1118–1129, 2009.

[83] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. R.: Towards internet-scale multiview stereo. In *Proceedings of IEEE Computer Vision and Pattern Recognition*, 2010.

[84] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.

[85] J. Garcia and Z. Zalevsky. Range mapping using speckle decorrelation, October 7 2008. US Patent 7,433,024.

[86] V. K. Ghorpade, D. Borrmann, P. Checchin, L. Malaterre, and L. Trassoudaine. Time-of-flight depth datasets for indoor semantic SLAM. In *Proceedings of the International Symposium on Robotics Research*. (accpeted), November 2017.

[87] U. L. Glückert and R. Schmidt. Pyrometry and thermography. In O. Feldmann and F. Mayinger, editors, *Optical Measurements: Techniques and Applications*, Heat and Mass Transfer. Springer Berlin Heidelberg, 2 edition, 2001.

[88] D. B. Goldman. Vignette and exposure calibration and compensation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(12):2276–2288, 2010.

[89] D. B. Goldman and J.-H. Chen. Vignette and exposure calibration and compensation. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 899–906. IEEE, 2005.

[90] H. González-Baños and J.C. Latombe. A randomized art gallery algorithm for sensor placement. *SCG'01: Proceedings of the Seventeenth Annual Symposium on Computational Geometry*, pages 232–240, 2001.

[91] Google Earth. Helgoland, April 2016. 54°11′15.23″ N and 7°52′15.71″ E, Imagery Date: 1/1/2008.

[92] G. Grisetti, C. Stachniss, and W. Burgard. `http://openslam.org/gmapping.html`.

[93] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics*, 23:34–46, 2007.

[94] F. Grosan, A. Tandrau, and A. Nüchter. Localizing Google SketchUp Models in Outdoor 3D Scans. In *Proceedings of the XXIII International Symposium on Information, Communication and Automation Technologies (ICAT '11)*, Sarajevo, Bosnia, October 2011. IEEE Xplore.

[95] Khronos Group. OpenGL 2.1 Reference Pages. Webpage, visited December 2016. `https://www.opengl.org/sdk/docs/man2/xhtml/glFog.xml`.

[96] D. Haehnel, W. Burgard, and S. Thrun. Learning compact 3d models of indoor and outdoor environments with a mobile robot. *Robotics and Autonomous Systems*, 44(1):15 – 27, 2003.

[97] Y. Ham and M. Golparvar-Fard. An automated vision-based method for rapid 3D energy performance modeling of existing buildings using thermal and digital imagery. *Advanced Engineering Informatics*, 2013.

[98] E. Hecht. *Optics*. Pearson, Boston, 5 edition, 2017.

[99] Heinle Cengage Learning and Collins. *Collins COBUILD Advanced Dictionary of English*. Harper Collins Publishers, 2009.

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

[101] B. K. P. Horn, H. M. Hilden, and S. Negahdaripour. Closed–form Solution of Absolute Orientation using Orthonormal Matrices. *Journal of the Optical Society of America A*, 5(7):1127 – 1135, July 1988.

[102] A. S. Hornby. *Oxford Advanced Learner's Dictionary of Current English*. Cornelsen, Berlin, sixth edition, 2000.

[103] P. V. C. Hough. Method and Means for Recognizing Complex Patterns. US Patent 3069654, December 1962.

[104] H. Houshiar. *Documentation and mapping with 3D point cloud processing*. PhD thesis, Julis-Maximilans-Universität Würzburg, February 2017.

[105] H. Houshiar, D. Borrmann, J. Elseberg, A. Nüchter, F. Näth, and S. Winkler. Castle3D - A Computer Aided System for Labelling Archaeological Excavations in 3D. In *Proceedings of 25th CiPA Symposium, ISPRS Annals Photogrammetry and Remote Sensing Spatial Inf. Sci., II-5/W3*, pages 111–118, Taipei, Taiwan, September 2015.

[106] H. Houshiar, J. Elseberg, D. Borrmann, and A. Nüchter. Panorama Based Point Cloud Reduction and Registration. In *Proceedings of the 16th IEEE International Conference on Advanced Robotics (ICAR '13)*, Montevideo, Uruguay, 2013.

[107] H. Houshiar, J. Elseberg, D. Borrmann, and A. Nüchter. A Study of Projections for Key Point Based Registration of Panoramic Terrestrial 3D Laser Scans. *Journal of Geo-spatial Information Science*, 18(1):11–31, March 2015.

[108] J. Illingworth and J. Kittler. A Survey on the Hough Transform. *Computer Vision, Graphics, and Image Processing*, 44:87–116, 1988.

[109] S. Izadi, R. A. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, J. Shotton, A. J. Davison, and A. Fitzgibbon. Kinectfusion: Real-time dynamic 3d surface reconstruction and interaction. In *ACM SIGGRAPH 2011 Talks*, SIGGRAPH '11, pages 23:1–23:1, New York, NY, USA, 2011. ACM.

[110] C. G. J. Jacobi. Über ein leichtes Verfahren die in der Theorie der Säcularstörungen vorkommenden Gleichungen numerisch aufzulösen. *Journal für die reine und angewandte Mathematik*, 30:51–94, 1846.

[111] A. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, 14:841, 2002.

[112] H. Kälviäinen, P. Hirvonen, L. Xu, and E. Oja. Probabilistic and Non-Probabilistic Hough Transforms: Overview and Comparisons. *Image and Vision Computing*, 13(4), May 1995.

[113] T. Kanzok, L. Linsen, and P. Rosenthal. On-the-fly luminance correction for rendering of inconsistently lit point clouds. *Journal of WSCG*, 20(3):161–169, 2012.

[114] T. Kersten and K. Mechelke. Fort Al Zubareh in Katar – 3D-Modell aus Scanner und Bilddaten im Vergleich. In T. Luhmann and C. Müller, editors, *Photogrammetrie Laserscanning Optische 3D-Messtechnik, Beiträge der Oldenburger 3D-Tage 2012, Jade Hochschule*, pages 108–119. Wichmann Verlag, February 2012.

[115] M. Kimura, M. Mochimaru, and T. Kanade. Projector calibration using arbitrary planes and calibrated camera. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–2. IEEE, 2007.

[116] N. Kiryati, Y. Eldar, and A. M. Bruckstein. A Probabilistic Hough Transform. *Pattern Recognition*, 24(4):303–316, 1991.

[117] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf. A flexible and scalable slam system with full 3d motion estimation. In *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, November 2011.

[118] K. Konolige and A. Nüchter. Range Sensors. In Oussama Khatib Bruno Siciliano, editor, *Springer Handbook of Robotics*, chapter 31, pages 783–810. Springer, 2 edition, 2016.

[119] D. Kragic and K. Daniilidis. 3-D Vision for Navigation and Grasping. In Oussama Khatib Bruno Siciliano, editor, *Springer Handbook of Robotics*, chapter 32, pages 811–824. Springer, 2 edition, 2016.

[120] L. Krüger. *Model based object classification and localisation in multiocular images*. PhD thesis, Bielefeld University, 2007.

[121] R. Lakaemper and L. J. Latecki. Extended EM for Planar Approximation of 3D Data. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[122] H. A. Lauterbach. Farbkorrektur von Kamerabildern zur 3D Modellierung. Master's thesis, Julis-Maximilans-Universität Würzburg, September 2016.

[123] H. A. Lauterbach, D. Borrmann, R. Hess, D. Eck, K. Schilling, and A Nüchter. Evaluation of a Backpack-Mounted 3D Mobile Scanning System. *Remote Sensing*, 7(10):13753–13781, 2015.

[124] H. A. Lauterbach, D. Borrmann, and A. Nüchter. Towards Radiometrical Alignment of 3D Point Clouds. In *Proceedings of the 6th ISPRS International Workshop 3D-ARCH 2017: "3D Virtual Reconstruction and Visualization of Complex Architectures", International Archives of Photogrammetrie Remote Sensing and Spatial Information Science, XLII-2/W3*, pages 419–424, Nafplio, Greece, March 2017.

[125] K. Y. K. Leung, D. Lühr, H. Houshiar, F. Inostroza, D. Borrmann, M. Adams, A. Nüchter, and Javier Ruiz del Solar. El Teniente Underground Mine Dataset. *International Journal of Robotics Research (IJRR)*, 36(1):16–23, January 2017.

[126] F. Leutert, C. Herrmann, and K. Schilling. A spatial augmented reality system for intuitive display of robotic data. In *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*, pages 179–180. IEEE Press, 2013.

[127] M. Li. Camera calibration of a head-eye system for active vision. In J. O. Eklundh, editor, *Computer Vision - ECCV 1994. Lecture Notes in Computer Science*, volume 800. Springer, Berlin, Heidelberg, Germany, 1994.

[128] K.L. Low and A. Lastra. Efficient constraint evaluation algorithms for hierarchical next-best-view planning. In *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, pages 830–837. IEEE, 2007.

[129] D. G. Lowe. Distinctive Image Features from Scale-invariant Keypoints. *International Journal of Computer Vision*, 2004.

[130] F. Lu and E. Milios. Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4:333 – 349, April 1997.

[131] T. Luhmann, J. Piechel, J. Ohm, and T. Roelfs. Geometric calibration of thermographic cameras. *International Archives of Photogrammetry, Remote Sensing and Spatial Information, Commission V Symposium*, 38(5), 2010.

[132] N.N. Madras. *Lectures on Monte Carlo Methods*. Fields Institute for Research in Mathematical Sciences Toronto: Fields Institute monographs. American Mathematical Society, 2002.

[133] M. R. Marner and B. H. Thomas. Augmented foam sculpting for capturing 3d models. In *3D User Interfaces (3DUI), 2010 IEEE Symposium on*, pages 63–70. IEEE, 2010.

[134] J. Matas, C. Galambos, and J. Kittler. Progressive Probabilistic Hough Transform. In *Proceedings of the British Machine Vision Conference*, volume 1, pages 256–265, 1998.

[135] S. May, P. Koch, R. Koch, C. Merkl, C. Pfitzer, and A. Nüchter. A generalized 2d and 3d multi-sensor data integration approach based on signed distance functions for multi-modal robotic mapping. In *Proceedings of th 19th International Workshop on Vision, Modeling and Visualization (VMV '14)*, Darmstadt, Germany, October 2014.

[136] M. Mettenleiter, F. Härtl, S. Kresser, and C. Fröhlich. *Laserscanning*, volume 371 of *Die Bibliothek der Technik*. verlag moderne industrie, Munich, Germany, 2015.

[137] D. Molyneaux, S. Izadi, D. Kim, O. Hilliges, S. Hodges, X. Cao, A. Butler, and H. Gellersen. Interactive environment-aware handheld projectors for pervasive computing spaces. In *Pervasive Computing*, pages 197–215. Springer, 2012.

[138] S. J. Moorehead, R. Simmons, and W. L. Whittaker. Autonomous exploration using multiple sources of information. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '01)*, Seoul, Korea, May 2001.

[139] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 116–121, March 1985.

[140] H.P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2):61–74, 1988.

[141] D. Moreno and G. Taubin. Simple, accurate, and robust projector-camera calibration. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 464–471. IEEE, 2012.

[142] K. Murphy. Bayesian map learning in dynamic environments. In *In Proc. of the Conf. on Neural Info. Proc. Systems (NIPS)*, pages 1015–1021, Denver, CO, USA, 1999. MIT Press.

[143] C. R. Nave. Hyperphysics. Webpage, visited June 2017. `http://hyperphysics.phy-astr.gsu.edu`.

[144] Nikon Metrology. iSpace – Portable Metrology System User Manual and Startup Guide. Webpage, visited July 2014. `http://www.nikonmetrology.com`.

[145] D. Novák and K. Schindler. Approximate Registration of Point Clouds with large scale Differences. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, ISPRS Workshop Laser Scanning 2013*, II-5/W2, November 2013.

[146] A. Nüchter. Autonome Exploration und Modellierung von 3D-Umgebungen. Master's thesis, Universität Bonn, July 2002.

[147] A. Nüchter. *Semantische dreidimensionale Karten für autonome mobile Roboter*. Number 303 in DISKI, Dissertationen zur künstlichen Intelligenz. Infinix / Aka Verlag, Berlin, Germany, November 2006.

[148] A. Nüchter. *3D Robotic Mapping: The Simultaneous Localization and Mapping Problem with Six Degrees of Freedom*. Number 52 in Springer Tracts in Advanced Robotics. Springer, February 2009.

[149] A. Nüchter et al. 3DTK — The 3D Toolkit. `http://threedtk.de/`, visited September 2017.

[150] A. Nüchter, J. Elseberg, and D. Borrmann. Irma3D – An Intelligent Robot for Mapping Applications. In *Proceedings of the 3rd IFAC Symposium on Telematics Applications (TA '13)*, volume 3, Seoul, Korea, November 2013.

[151] A. Nüchter, J. Elseberg, P. Schneider, and D. Paulus. Study of Parameterizations for the Rigid Body Transformations of The Scan Registration Problem. *Journal Computer Vision and Image Understanding (CVIU)*, 114(8):963–980, August 2010.

[152] A. Nüchter, K. Lingemann, D. Bormann, J. Elseberg, and J. Böhm. Global Konsistente 3D-Kartierung mit Scanmatching. In T. Luhmann and C. Müller, editors, *Photogrammetrie Laserscanning Optische 3D-Messtechnik, Beiträge der Oldenburger 3D-Tage 2008, Fachhochschule Oldenburg/Ostfr./Whv.* Wichmann Verlag, January 2008.

[153] A. Nüchter and K. Lingemann. Robotic 3D Scan Repository. `http://kos.informatik.uos.de/3Dscans/`, visited September 2017.

[154] E. Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407. IEEE, May 2011.

[155] A. Olwal, J. Gustafsson, and C. Lindfors. Spatial augmented reality on industrial cnc-machines. In *Electronic Imaging 2008*, pages 680409–680409. International Society for Optics and Photonics, 2008.

[156] OpenCV. `http://opencv.org/`, visited September 2017.

[157] Optris GmbH. Basic Principles Of Non-Contact Temperature Measurement, March 2013.

[158] Ostia Antica. Soprintendenza speciale per i beni archeologici di Roma – Sede di Ostia. Webpage, visited July 2014. `http://www.ostiaantica.beniculturali.it`.

[159] K. L. Palmerius and K. Schönborn. Visualization of heat transfer using projector-based spatial augmented reality. In *International Conference on Augmented Reality, Virtual Reality and Computer Graphics*, pages 407–417. Springer, 2016.

[160] Panasonic. DMC-LS80 Specifications.
ftp://ftp.panasonic.com/pub/Panasonic/consumer_electronics/presskits/LS80_Spec_Sheet.doc.

[161] J. Poppinga, N. Vaskevicius, A. Birk, and K. Pathak. Fast Plane Detection and Polygonalization in noisy 3D Range Images. In *IROS*, 2008.

[162] S. R. Porter, M. R. Marner, R. T. Smith, J. E. Zucco, and B. H. Thomas. Validating spatial augmented reality for interactive rapid prototyping. In *Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on*, pages 265–266. IEEE, 2010.

[163] F. L. Predotti, L. S. Predotti, W. Bausch, and H. Schmidt. *Optik für Ingenieure: Grundlagen*. Springer, Berlin Heidelberg, 2 edition, 2002.

[164] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 1985.

[165] S. Prieto, B. Quintana Galera, A. Adan, and A.S. VÃ¡zquez. As-is building-structure reconstruction from a probabilistic next best scan approach. *Robotics and Autonomous Systems*, 94, 05 2017.

[166] S. Pu and G. Vosselman. Knowledge based reconstruction of building models from terrestrial laser scanning data. *Journal of Photogrammetry and Remote Sensing*, 64(6):575 – 584, 2009.

[167] F. Remondino and C. Fraser. Digital camera calibration methods: Considerations and comparisons. In *International Archives of Photogrammetry, Remote Sensing, and Spatial Information Sciences XXXVI(5)*, pages 266–272, 2006.

[168] A. Richardson, J. Strom, and E. Olson. AprilCal: Assisted and repeatable camera calibration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, November 2013.

[169] B. Ridel, P. Reuter, J. Laviole, N. Mellado, N. Couture, and X. Granier. The revealing flashlight: Interactive spatial augmented reality for detail exploration of cultural heritage artifacts. *Journal on Computing and Cultural Heritage (JOCCH)*, 7(2):6, 2014.

[170] K. O. Rinnewitz, T. Wiemann, K. Lingemann, and J. Hertzberg. Automatic creation and application of texture patterns to 3d polygon maps. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3691–3696, November 2013.

[171] N. Ripperda and C. Brenner. Application of a formal grammar to facade reconstruction in semiautomatic and automatic environments. In *Proceedings of AGILE Conference on Geographic Information Science*, Hannover, Germany, 2009.

[172] RoboticsAndTelematics. Kaisersaal. `http://youtu.be/jKVxlLvu7Pk`, May 2014.

[173] RoboticsAndTelematics. Kaisersaal 3D. `http://youtu.be/olzaronoC0E`, May 2014.

[174] RoboticsAndTelematics. Weißer Saal. `http://youtu.be/_wPug_So_iE`, May 2014.

[175] S. Salamanca, P. Merchan, E. Perez, A. Adan, and C. Cerrada. Filling holes in 3D meshes using image restoration algorithms. In *Proceedings of the Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, Atlanta, Georgia, United States of America, 2008.

[176] J. Salvi, X. Armangué, and J. Batlle. A comparative review of camera calibrating methods with accuracy evaluation. *Pattern Recognition*, 35(7):1617 – 1635, 2002.

[177] R. Schnabel, R. Wahl, and R. Klein. Efficient RANSAC for Point-Cloud Shape Detection. *Computer Graphics Forum*, 2007.

[178] N. Schuster and V. G. Kolobrodov. *Infrarotthermographie*. WILEY-VCH Verlag Berlin GmbH, 1st edition, 2000.

[179] M. Seder and I. Petrović. Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation (ICRA '07)*, pages 1986–1991, 2007.

[180] Senatskanzlei Bremen. Rathaus und Roland zu Bremen – Welterbe der Menschheit. Webpage, visited September 2017. `http://www.rathaus.bremen.de/welterbe___architektur-2085`.

[181] T. K. Sharpless, B. Postle, and D. M. German. Pannini: A New Projection for Rendering Wide Angle Perspective Images. In *International Symposium on Computational Aesthetics*, London, June 2010.

[182] M. Shortis. Calibration Techniques for Accurate Measurements by Underwater Camera Systems. *Sensors*, 15(12):30810–30826, 2015.

[183] M. R. Shortis, S. Robson, and H. A. Beyer. Principal Point Behaviour and Calibration Parameter Models for Kodak DCS Camera. *Photogrammetric Record*, 16(92), October 1998.

[184] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH Conference Proceedings*, pages 835–846, New York, NY, USA, 2006. ACM Press.

[185] Canon EOS 1000D Specifications. `http://web.canon.jp/imaging/eosd/eosdigital5/specifications.html`.

[186] I. Stamos, G. Yu, G. Wolberg, and S. Zokai. Application of a formal grammar to facade reconstruction in semiautomatic and automatic environments. In *Proceedings of 3D Data Processing, Visualization, and Transmission (3DPVT)*, pages 599 – 606, 2006.

[187] S. M. Stewart. Blackbody radiation functions and polylogarithms. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 113(3):232–238, February 2012.

[188] W. Sun and J.R. Cooperstock. An empirical evaluatoin of factors influencing camera calibration accuracy using three publicly available techniques. *Machine Vision and Applications*, 17(51), 2006.

[189] H. Surmann, A. Nüchter, and J. Hertzberg. An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Journal Robotics and Autonomous Systems (JRAS)*, 45(3–4):181–198, December 2003.

[190] ROS Robot Operating System. `http://www.ros.org`.

[191] Riegl Laser Measurement Systems. RiSCAN PRO. Software Description & User's Instruction, 2009. Version 1-5-0sp1.

[192] Riegl Laser Measurement Systems. http://riegl.com/nc/products/terrestrial-scanning/, visited September 2017.

[193] Riegl Laser Measurement Systems. Data Sheet, Riegl VZ-400. `http://riegl.com/nc/products/terrestrial-scanning/`, visited June 2017.

[194] F. Tarsha-Kurdi, T. Landes, and P. Grussenmeyer. Hough-transform and extended ransac algorithms for automatic detection of 3d building roof planes from lidar data. *ISPRS Volume XXXVI, Part 3 / W52*, 2007.

[195] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. Intelligent robotics and autonomous agents. the MIT Press, Cambridge (Mass.) (London), 2005.

[196] S. Thrun, C. Martin, Y. Liu, D. Haehnel, R. Emery-Montemerlo, D. Chakrabarti, and W. Burgard. A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots. *IEEE Transactions on Robotics*, 20(3):433 – 443, 2004.

[197] B. Tovar, R. Murrieta-Cid, and S.M. LaValle. Distance-optimal navigation in an unknown environment without sensing distances. *IEEE Transactions on Robotics*, 23(3):506–518, 2007.

[198] P. van Walree. Vignetting. `http://toothwalker.org/optics/vignetting.html`. Visited August, 2016.

[199] B. R. Vatti. A generic solution to polygon clipping. *Communications of the ACM*, 35(7):56–63, 1992.

[200] S. Vidas and P. Moghadam. HeatWave: A handheld 3D thermography system for energy auditing. *Energy and Buildings*, 66:445–460, November 2013.

[201] S. Vidas, P. Moghadam, and M. Bosse. 3D Thermal Mapping of Building Interiors Using an RGB-D and Thermal Camera. In *Proceedings of the IEEE Conference on Robotics and Automation (ICRA '13)*, Karlsruhe, Germany, May 2013.

[202] J. Weng, P. Cohen, and M. Herniou. Camera Calibration with Distortion Models and Accuracy Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):965–980, October 1992.

[203] A. Wilson, H. Benko, S. Izadi, and O. Hilliges. Steerable augmented reality with the beamatron. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 413–422. ACM, 2012.

[204] S. Winkelbach, S. Molkenstruck, and F. M. Wahl. Low-cost laser range scanner and fast surface registration approach. In Katrin Franke, Klaus-Robert Müller, Bertram Nickolay, and Ralf Schäfer, editors, *Pattern Recognition: 28th DAGM Symposium, Berlin, Germany, September 12-14, 2006. Proceedings*, pages 718–728. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[205] C. Wu. Towards linear-time incremental structure from motion. In *Proceedings of the 2013 International Conference on 3D Vision*, 3DV '13, pages 127–134, Washington, DC, USA, 2013. IEEE Computer Society.

[206] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *CVPR*, pages 3057–3064. IEEE Computer Society, 2011.

[207] O. Wulf, K. O. Arras, H. I. Christensen, and B. A. Wagner. 2D Mapping of Cluttered Indoor Environments by Means of 3D Perception. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '04)*, pages 4204 – 4209, New Orleans, USA, April 2004.

[208] O. Wulf, A. Nüchter, J. Hertzberg, and B. Wagner. Benchmarking Urban 6D SLAM. *Journal of Field Robotics (JFR)*, 25(3):148–163, March 2008.

[209] Würzburg Residence. Bayerische Verwaltung der staatlichen Schlösser, Gärten und Seen. Webpage, visited July 2014. `http://www.residenz-wuerzburg.de`.

[210] L. Xu, E. Oja, and P. Kultanen. A new Curve Detection Method: Randomized Hough Transform (RHT). *Pattern Recognition Letters*, 11:331–338, 1990.

[211] A. Ylä-Jääski and N. Kiryati. Adaptive Termination of Voting in the Probabilistic Circular Hough Transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9), 1994.

[212] R. Yoshimura, H. Date, S. Kanai, R. Honma, K. Oda, and T. Ikeda. Automatic registration of MLS point clouds and SfM meshes of urban area. *Geo-spatial Information Science*, 19(3):171–181, 2016.

[213] G. Yu, M. Grossberg, G. Wolberg, and I. Stamos. Think Globally, Cluster Locally:A Unified Framework for Range Segmentation. In *Proceedings of the 4th International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT '08)*, Atlanta, USA, 2008.

[214] L. Zagorchev and A. Goshtasby. A paintbrush laser range scanner. *Comput. Vis. Image Underst.*, 101(2):65–86, February 2006.

[215] T. Zaharia and F. Preteux. Shape-based Retrieval of 3D Mesh Models. In *Proceedings of the IEEE International Conferenc on Multimedia and Expo*, 2002.

[216] X. Zhang, S. Fronz, and N. Navab. Visual marker detection and decoding in ar systems: A comparative study. In *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, ISMAR '02, pages 97–, Washington, DC, USA, 2002. IEEE Computer Society.

[217] X. F. Zhang, A. Luo, W. Tao, and H. Burkhardt. Camera calibration based on 3D-point-grid. In Alberto Del Bimbo, editor, *Image Analysis and Processing: 9th International Conference, ICIAP '97*, pages 636–643. Springer Berlin Heidelberg, Florence, Italy, September 1997.

[218] Z. Zhang. A flexible new technique for camera calibration. *Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[219] J. Zhou, I. Lee, B. Thomas, R. Menassa, A. Farrant, and A. Sansome. In-situ support for automotive manufacturing using spatial augmented reality. *The International Journal of Virtual Reality*, 11(1):33–41, 2012.