Die approbierte Originalversion dieser Dissertation ist in der Hauptbibliothek der Technischen Universität Wien aufgestellt und zugänglich.
http://www.ub.tuwien.ac.at

The approved original version of this thesis is available at the main library of the Vienna University of Technology.
http://www.ub.tuwien.ac.at/eng

DISSERTATION

Autonomous Calibration of Depth Sensors For Robotic Vision

conducted in partial fulfillment of the requirements for the degree of a
Doktor der technischen Wissenschaften (Dr. techn.)

supervised by

Ao. Univ. Prof. Dipl.-Ing. Dr. techn. Markus Vincze
E376 Automation and Control Institute

submitted at the

TU Wien
Faculty of Electrical Engineering and Information Technology

by

Dipl.-Ing. Georg Halmetschlager-Funek
DOB 14.02.1987
Matr. Nr.: 0929189

Vienna, May 2019 G. Halmetschlager-Funek

Abstract

In the last few years the first generation of novel color and depth (RGB-D) sensors greatly pushed the frontier of robot perception. Dense depth data combined with color information offers the opportunity to directly relate rich sensor signals to the three dimensional operational space of the robot. Although RGB-D sensors are widely used, only little is known about their properties.

Knowing the exact error modalities is essential to develop a sensor simulation model which is necessary to test and desensitize algorithms for depth sensing errors with large-scale synthetic datasets. Often it is impractical to desensitize an algorithm, but at the same time it might be impossible to get decent depth estimates from the sensor. That motivates methods which enable a correction of the depth information in a post-correction step. If a correction of the depth measurements is not sufficient, i.e. if a precise pixel-wise fusion of color and depth information is required, an accurate calibration is necessary. However, the autonomous calibration of RGB-D setups is still an open problem. In order to relate the sensor information to other components of the robot, a holistic calibration should recover the rigid transformation between the sensor and a robot-fixed reference system.

This work addresses these open challenges with five approaches. First, it evaluates ten different RGB-D and depth sensors covering the three main sensor technologies: Structured Light, Active Stereo, and Time of Flight (ToF). Second, it continues with an intuitive and generic method to simulate characteristic systematic errors of the investigated state-of-the-art depth sensors. Third, two alternative camera-model-agnostic depth compensation methods are added that correct systematic depth sensing errors, without altering the sensors internal parameters. Fourth, the alignment of RGB and depth data is improved with a robust autonomous calibration algorithm that utilizes structure from motion (SfM) and incorporates plane priors in the optimization. Fifth, this work contributes an extrinsic pose calibration algorithm that uses the trajectory of the robot to find an optimal rigid transformation between the camera and a reference system.

The results of the sensor study show characteristic error modalities dependent on the sensor technology. The systematic depth error simulation model replicates the characteristic non-linear, and radial-shaped, local errors as well as linearly increasing errors of the investigated sensors. The presented camera-model-agnostic depth compensation methods successfully improve the trueness and the precision of the depth measurements. The introduced auto-calibration method shows an significant improvement of the calibration accuracy compared to the state of the art. In first experiments, the extrinsic camera pose calibration recovers the camera pose relative to a reference system.

Kurzzusammenfassung

In den letzten Jahren erweiterten neuartige Farb- und Tiefensensoren (RGB-D) die Grenzen der Roboterwahrnehmung. Hochauflösende Tiefendaten, kombiniert mit RGB Information, ermöglichen es detaillierte Sensorsignale dem dreidimensionalen Raum zuzuordnen. Allerdings ist nur wenig über die exakten Eigenschaften dieser Sensoren bekannt.

Kenntnisse über die exakten Fehlermodalitäten sind für die Entwicklung eines Sensorsimulationsmodells essentiell. Dieses ist wiederum notwendig um Algorithmen auf Basis synthetischer Daten gegenüber Tiefenfehlern zu desensibilisieren. Oft ist eine Desensibilisierung des Algorithmus nicht möglich, während die Sensoren zusätzlich unzuverlässige Daten liefern. Dies motiviert Verfahren, die Tiefenfehler nachträglich korrigieren können, jedoch ist eine Korrektur der Tiefendaten gelegentlich unzureichend. Beispielsweise wenn eine pixelgenaue Assoziation zwischen RGB- und Tiefenwerten benötigt wird, was eine exakte Kalibrierung erfordert. Die autonome Kalibrierung von RGB-D Sensoren ist dabei ein noch ungelöstes Problem. Zusätzlich sollte eine holistische Kalibrierung im Stande sein, die rigide Transformation zwischen dem Sensor und einem roboterfesten Koordinatensystem zu bestimmen.

Diese Arbeit befasst sich mit den erwähnten Herausforderungen und präsentiert fünf Lösungsvorschläge. Erstens werden zehn aktuelle Sensoren evaluiert, die die drei häufigsten RGB-D Sensorfamilien abdecken. Zweitens wird ein generics Tiefenfehlersimulationsmodell untersucht. Drittens werden zwei Kameramodell-unabhängige Methoden präsentiert, die es erlauben Tiefenfehler nachträglich zu korrigieren. Viertens wird die Assoziation von RGB- und Tiefendaten durch einen robusten autonomen Kalibrierungsalgorithmus verbessert, der eine Structure from Motion (SfM) Rekonstruktion und Annahmen über Ebenen im Raum mit einem Optimierungsproblem kombiniert. Fünftens erweitert diese Arbeit den Stand der Technik um eine Methode zur automatischen Kalibrierung der Sensorpose relativ zu einem roboterfesten Koordinatensystem.

Die Ergebnisse der Sensorstudie zeigen charakteristische Fehlermodalitäten abhängig von der verwendeten Sensortechnologie. Das vorgestellte Simulationsmodell repliziert die linearen, nicht-linearen und lokalen Fehler der untersuchten Sensoren. Die Kameramodell-unabhängigen Kompensationsmethoden verbessern erfolgreich die Richtigkeit und Präzision der Messungen. Die automatische Kalibrierungsmethode zeigt gegenüber dem Stand der Technik eine signifikante Verbesserung der Assoziation von Tiefen- und RGB Daten. Der Algorithmus zur Kalibrierung der Sensorpose extrahiert in ersten Experimenten erfolgreich die rigide Transformation zwischen dem Sensor und einem roboterfesten Referenzsystem.

Contents

1 Introduction **1**
1.1 Motivation 2
1.2 Research Questions and Contributions 6
1.3 Related Work 10
1.3.1 Sensor Calibration and Depth Error Models 11
1.3.2 Camera Calibration 12
1.3.3 Rigid Transformation Calibration Between Sensors 13
1.4 Related Publications 14

2 Depth Sensor Error Modalities **15**
2.1 Sensing Technologies for Indoor Robotics 16
2.2 Evaluation of Ten Different Depth Sensors 16
2.2.1 Performance Characteristics 17
2.2.2 Experimental Setup 19
Experiment I/II - Trueness, Precision & Lateral Noise 20
Experiment III - Lighting & Materials 21
Experiment IV - Multiple Sensors 21
2.2.3 Results 21
Trueness 22
Precision 22
Lateral Noise 22
Materials 23
Additional Sensors 24
Use Case 25
2.3 Discussion 25

3 Simulation of Systematic Depth Errors **29**
3.1 Error Model 30
3.2 Evaluation 33
3.2.1 Simulated Error Modalities 34
3.2.2 Synthetic Data Augmentation 36
3.3 Discussion 38

4 Camera-Model-Free Compensation of Systematic Depth Errors **41**
4.1 Methods for Camera Model Free Depth Correction 41
4.1.1 Levenberg-Marquardt Algorithm 41

4.1.2 Bicubic Interpolation 43
4.2 Discrete and Continuous Offset Compensation Models 46
4.2.1 Depth-Discrete Model 46
Jacobians 48
4.2.2 Depth-Continuous Model 49
Jacobians 51
4.3 Evaluation 51
4.3.1 Datasets 51
4.3.2 Depth-Discrete Error Compensation Model 52
4.3.3 Depth-Continuous Error Compensation Model 53
4.3.4 Limitations 56
4.3.5 Performance Analysis 57
4.4 Discussion 58

5 Autonomous Calibration of RGB-D Setups 61
5.1 Auto Calibration 62
5.1.1 Residuals 63
Reprojection Error 64
Depth Map Alignment Between Views 64
Inverse Point-Point Distance 65
Point-Plane Distance 65
5.1.2 Weighting Functions 66
Equalized Plane Weighting (WIPD,WPLD) 66
Residual Balancing (rB) 67
5.1.3 Optimization 67
5.1.4 Implementation 68
5.2 Evaluation 68
5.2.1 Baseline Validation 69
5.2.2 Datasets 69
5.2.3 Qualitative and Quantitative Evaluation 70
5.2.4 Results 71
5.2.5 Use Case Scenario 72
5.3 Discussion 73

6 Towards Holistic Autonomous Calibration 75
6.1 Camera Pose Calibration 75
6.1.1 Robot Trajectory Estimation 76
6.1.2 Camera Trajectory Estimation 77
6.1.3 Extrinsic Pose Calibration 77
6.1.4 Motion Blur Detection 78
6.2 Evaluation 78
6.2.1 Datasets 79
Synthetic Dataset 79
Real-Life Dataset 79
Evaluation Dataset 79

6.2.2 Implementation and Evaluation Framework 79
6.2.3 Result 80
Motion Blur Detection 80
Extrinsic Pose Calibration 81
6.3 Discussion 82

7 Conclusion 85
7.1 Summary 85
7.2 Outlook 87

A 3D Reconstructions of the ICL-NUIM Dataset 93

B Bicubic Interpolation 99

C Autonomous Calibration of RGB-D Setups 100

List of Figures

1.1 "En L'An 2000". Drawings from late 19th and early 20th century showing robotic devices. Re-published by Asimov [1]. 1
1.2 Field robot FRANC and floor washing robot FLOBOT. 2
1.3 Number of publications per year retrieved from a google scholar search with the keywords: "RGB-D"+"robotic". 3
1.4 Left: Robot used for sensor integration. Middle/right: Depth data gathered with different sensors (middle: Orbbec 3D [21], right: KinectV2 [22]). 4
1.5 Alignment of RGB and depth images using the inverse point-point distance [32] (left) compared to the method presented in this work that adds the point-plane distance (right). It shows improved convergence behavior and a more accurate alignment of RGB and depth images, even on datasets that offer motion blur and low image quality. 5
1.6 The trueness and the precision of ten different depth sensors is evaluated under various conditions. This work has been published in [37]. 6
1.7 Overview of the depth error model that uses a virtual stereo camera pair. It takes synthetic input data and augments them with the error model to get realistic depth information including realistic errors. 7
1.8 Overview of the camera model free depth offset compensation methods. 8
1.9 Overview of the autonomous auto calibration method that incorporates plane priors for more accurate alignment of depth and RGB images. The autonomous auto-calibration method has been published in [5]. 9
1.10 Overview of the hand-eye calibration method that estimates an optimal rigid transformation between a laser ranger that is used for navigation and a camera that is mounted on the robot. 10

2.1 Function principle of continuous wave ToF cameras [45]. 17
2.2 Relation between the error types, the performance characteristics and the quantitative expressions [68]. 18
2.3 Robotic s other wordsystem. 20
2.4 Setup for multi-sensor and material tests. 21
2.5 Trueness results for near and far range devices. Lower is better. 22
2.6 Precision results for near and far-range devices. Lower is better. 23
2.7 Precision and nan ratio (lower is better) in multi sensor setup. Indices represent distance to the target: 07= $0.7m$, 10= $1m$,15= $1.5m$. 24

2.8 Influence of additional sensors. SR300. Number of sensors observing the scene, from left to right: 1,2,3. 25

2.9 Reconstruction method from [64]. The two RGB images show the same location observed from different distances. The bottom images show the reconstruction results of the highlighted region (red circle), without (left) and with (right) the introduced error model. 25

2.10 Precision for different materials and lighting conditions. Lower is better. A precision of zero indicates that the sensor is not able to gather any depth information. 27

3.1 Depth data of a flat wall gathered with a factory calibrated R200 (green). The fitted plane highlights the radial error due to an imperfect calibration. With courtesy of DAQRI. 29

3.2 Total error of a R200 depth camera after factory calibration. The images show a systematic radial shaped depth error that increases over the depth and additional random noise. 31

3.3 Stereo error model overview. From left to right: First, the depth image is unprojected into the 3D space using the ground truth camera parameters. Second, the 3D points are back projected onto the left and right image plane, using the modified camera model, including lens distortion. Third, the depth image is recomputed based on the disparity values, the altered focal length, and modified baseline. 32

3.4 RMSE for focal length error, baseline error, principal point error and disparity quantization error. 35

3.5 RMSE for interpolation error and radial distortion error. 36

3.6 Radial distortion error. 36

3.7 Tangential distortion error. 37

3.8 Reconstruction generated with perfect depth and camera pose information. The generated 3D scene and the error histogram of the cloud-cloud distance between the CAD data and the reconstructed scene represent the baseline, and the absolute limits of the used reconstruction algorithm. 37

3.9 Impact of the interpolation error (orange), radial distortion error (blue), and tangential distortion error (turquoise) on the image quality. 38

3.10 Reconstruction error histograms, showing the point-point distance between the reconstructed scene and the ground truth CAD data. (a) Ground truth reconstruction; (b) quantization error; (c) interpolation error; (d) radial distortion error; (e) tangential distortion error; (f) principal point error; (g) focal length error; and (h) baseline error. 39

3.11 Left: ground truth reconstruction, right: reconstructed scene with applied baseline error. While the surface quality shows only slight flaws, the absolute accuracy shows a significant impact of the baseline error. . . . 39

3.12 Real data, left. Synthetically generated data, right. 40

4.1 Three different interpolation methods: The Nearest Neighbor interpolation is the simplest method and takes the value from the closest point without any interpolation between the points. The bilinear interpolation considers the value of the four corner points and interpolates between them. The bicubic interpolation considers the four corner points and their derivatives. 44
4.2 Bicubic interpolation scheme. 44
4.3 Depth compensation with depth-discrete compensation models. 46
4.4 Depth compensation with one compensation map and a depth-continuous exponential function. 47
4.5 Input images recording setup. 47
4.6 Sampling of several separate model and the resulting connection between the offset compensation models. 49
4.7 Plot of $f(z) = \exp(a_0 - a_1 \cdot z^{-1})$. 50
4.8 Trueness and precision for the synthetic dataset before (blue) and after (orange) applying the depth-discrete error compensation model. 52
4.9 Depth images for $d = 6.9$m of the synthetic dataset before and after applying the discrete compensation model. Blue represents small errors yellow represents high errors. 52
4.10 Trueness and precision for the real-life dataset before (blue) and after (orange) applying the depth-discrete error compensation model. 53
4.11 Depth at $d = 1.2$m before (left) and after (center) applying the discrete compensation model (right). Blue represents small errors, yellow represent high errors. 53
4.12 Surfaces before (left) and after (right) applying the depth-discrete error compensation model. 54
4.13 Trueness and precision for the synthetic dataset before (blue) and after (orange) applying the depth-continuous error compensation model. . . 54
4.14 Trueness and precision for the real-life dataset before (blue) and after (orange) applying the depth-continuous error model. 55
4.15 Depth at $d = 1.2$m before (left) and after (center) applying the continuous compensation model (right). 55
4.16 Compensation with Full Model. 56
4.17 Reconstruction using a corrected/uncorrected synthetic dataset of an indoor scene. The dataset is augmented with the depth simulation models presented in Chapter 3. 56
4.18 Cloud-Cloud distance error histogram, between reconstruction and ground truth CAD model. Absolute cloud-cloud distance to a least-squares fitted plane, k=6. 57
4.19 Limitations due to sampling in border regions. 57
4.20 Step 1 and Step 2, performance of analytic differentiation automatic differentiation. 58
4.21 Performance of the optimization using automatic differentiation (blue) and handcrafted analytic Jacobians (orange). 59

5.1 Camera Calibration to get the intrinsic parameters of the used R200 Active Stereo camera. 61

5.2 Overview of the calibration problem. (a) Depicts the reprojection error $\mathbf{r}_{\mathbf{vi}}^{\mathbf{B}}$ that is used for the bundle adjustment. (b) Shows the projection of an SfM map point into the depth camera frame, used for the inverse point-point distance r_{vi}^{I} and the point-plane distance r_{vi}^{P}. (c) Visualizes the projection of the depth value from one depth camera into a second view, used for the residual that models the alignment between a view pair j,k with r_{jk}^{V}. 64

5.3 Alignment of two point sets, point-point distance (left) vs. point-plane distance (right). While the point-point distance prevents movements of the points within the plane, the point-plane distance, allows movements at zero costs. 66

5.4 Results for the ICL-NUIM dataset using the method from [32]. The circles highlight the improved alignment of color and depth data. 69

5.5 QMSfM results for the reference method [32] and the presented method (only PLD enabled) on the office+blur dataset. 71

5.6 Qualitative results of the calibration. From left to right: scene, initial parameterization, reference method (IPD, [32]), the introduced method that gives the best qualitative result, using PLD, IPD, and WIPD. Figure best viewed in color. 73

5.7 Annotation results for a scene (a) of the dataset presented in [9]. (b) Shows the manual annotation in the RGB domain. The last two columns show the depth annotation results for a manual calibration (c) and with the presented calibration (d). 74

6.1 Sensor configuration of the experimental platform. Orange: RGB-D sensors, blue: Laser scanner. 76

6.2 Transformations for camera pose calibration. 78

6.3 Two different datasets that are used to train the motion blur detection algorithm. 79

6.4 Accuracy of algorithms trained on all data. 80

6.5 Accuracy of algorithms trained on v4rMB scenes. 81

6.6 Decision stump on different features. 81

6.7 (a) Histogram of the distance between the RGB-D depth sensor estimations and the laser-scan. (b) A pseudo-laser scan is extracted from the point cloud which is compared with the laser scan data of the robot. The two scans are well aligned and indicate a correct rigid transformation between the two systems. 82

6.8 Results for the distance between the RGB-D pseudo laser scan and the real laser scan. 82

7.1 Coverage statistic for one of the used datasets for auto calibration. The color information encodes the number of SfM point re-projections for the corresponding depth pixel. The histogram gives additional information regarding the covered distances. 88

A.1 Impact of the baseline error on the reconstruction. 94
A.2 Impact of the principal point error on the reconstruction. 94
A.3 Impact of the focal length error on the reconstruction. 95
A.4 Impact of the interpolation error on the reconstruction. 95
A.5 Impact of the quantization error on the reconstruction. 96
A.6 Impact of the tangential distortion error on the reconstruction 96
A.7 Impact of the radial distortion error on the reconstruction. 97
A.8 Impact of the all errors in combination on the reconstruction. 97
A.9 Impact of the camera pose error on the reconstruction. 98
A.10 Ground truth reconstruction. 98

List of Tables

2.1 Analyzed depth sensors and their specifications given by the manufacturers. 15
2.2 Lateral noise of the different sensors. 23

3.1 Virtual Camera Parameters. 34

5.1 Dataset Overview 70
5.2 Qualitative and quantitative results of the calibration method and combinations outperforming the state of the art. Example: Method that gives the overall best result (see column "$\sum$"): PLD enabled. Method that gives the best qualitative result (see column "man"): PLD, IPD, and WIPD enabled. A full table with all results can be found in the appendix. 72

C.1 Qualitative and quantitative results of the presented calibration method and combinations outperforming the state of the art. Example: Method that gives the overall best result (see column "$\sum$"): PLD enabled. Method that gives the best qualitative result (see column "man"): PLD, IPD, and WIPD enabled. 101

Chapter 1

Introduction

In the year 1986, Isaac Asimov, writer and professor of biochemistry, published a collection of several images called “En L’An 2000” [1]. The series had been drawn around the turn of the 19th to the 20th century and depicts scientific advances imagined to be achieved in the year 2000. Among various still seemingly futuristic drawings are images of autonomous and remote controlled machines that carry out everyday tasks as: floor cleaning, constructing a building, farming and giving a costumer a haircut (cf. Figure 1.1).

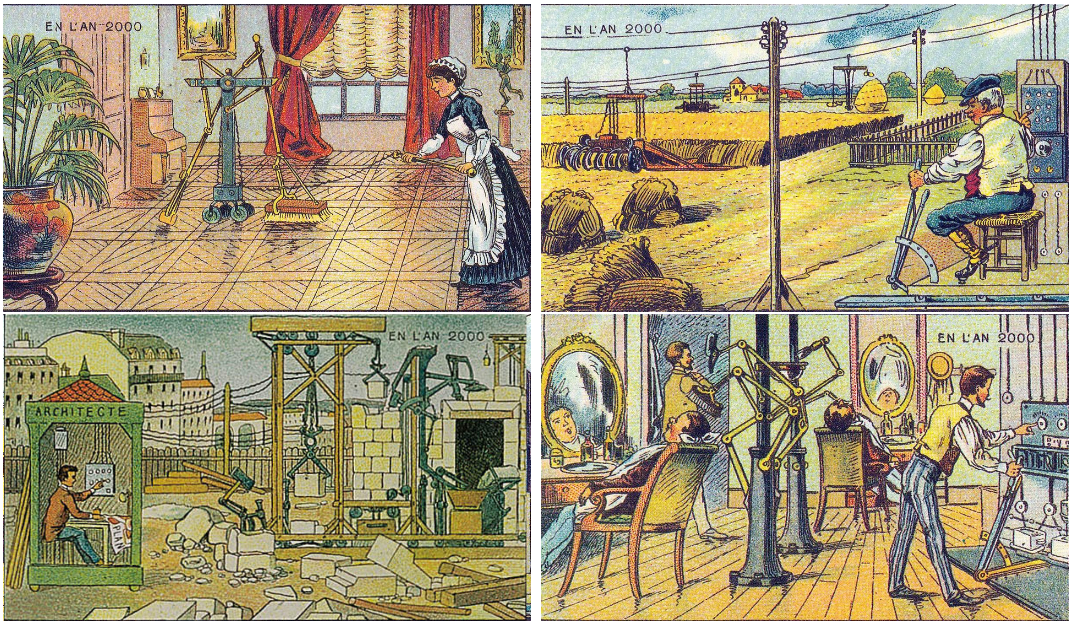


Figure 1.1: “En L’An 2000”. Drawings from late 19th and early 20th century showing robotic devices. Re-published by Asimov [1].

Twenty years after “En L’An 2000” had been drawn, in his theater play “R.U.R. - Rossumovi Univerzální Roboti” [2], Carl Čapek gave the concept of such autonomous machines a name: Robot.

Now, nearly 100 years later, we are at the technological edge to realize those dreams.

Industrial Robots have become ubiquitous in modern factories, increase the productivity, and take over repetitive, hard, challenging, or even dangerous work.

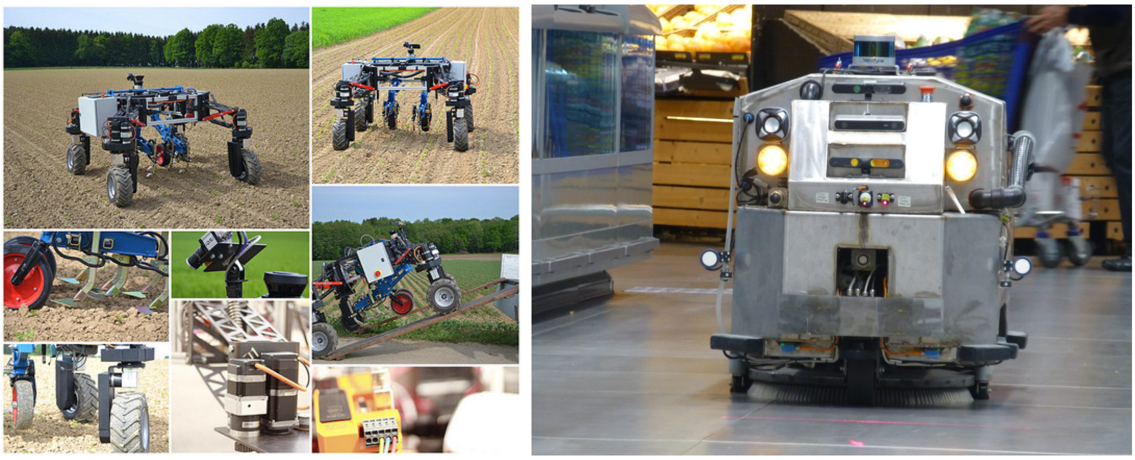


Figure 1.2: Field robot FRANC and floor washing robot FLOBOT.

During the last decade the first service robots made their way into our homes. Among the first robots were automated floor cleaning (or vacuuming) systems. The task of floor cleaning (vacuuming) provided many preconditions that paved the way for early robot-based automation. Flat surfaces, a well-defined work space, and a simple task made it possible to narrow down the robot's functional requirements to the essentials: wander around with an activated cleaning system and drive back to the docking station as soon as the energy drops below a predefined level. These types of tasks are sufficiently handled with low-level sensors and behavior-based control algorithms.

Over the years the technology advanced and educed more sophisticated robot systems. The robots are targeted to solve more complex tasks and demand more detailed information of the environment: The field robot FRANC [3], the floor washing robot FLOBOT [4]–[6], and the personal assistant HOBBIT [7] are just a small subset of first prototypes representing the bleeding edge of the current state of the art.

All these robots combine the information of various sensors to get the most accurate and rich representation of the environment possible. Often, a color (RGB) camera is combined with a depth sensor to a so-called RGB-D sensor. Recently, RGB-D sensors became affordable and extend the list of state-of-the-art sensors in robotic research. Although these sensors are widely used and become more and more popular, comparably little is known about their exact properties.

1.1 Motivation

Since the introduction of Microsoft's affordable depth sensor, Kinect [8] in the year 2010, RGB-D sensors have become an essential component for many methods and applications using machine vision, especially in the field of robotics. A simple Google Scholar[1] search for the combination of the keywords "robotic" and "RGB-D" reveals a steady growth of the field that started together with the release of the Kinect sensor (cf. Figure 1.3).

---

[1] `https://scholar.google.at/`

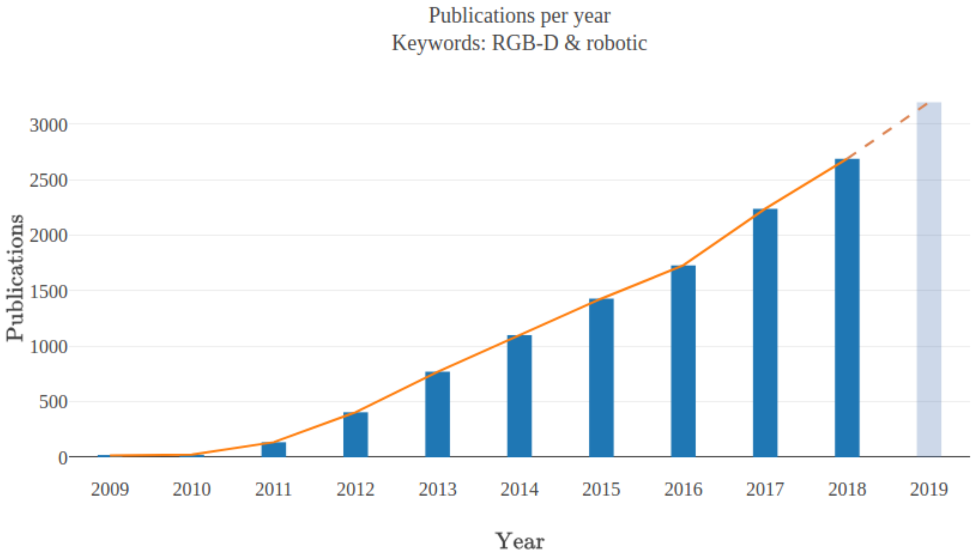


Figure 1.3: Number of publications per year retrieved from a google scholar search with the keywords: "RGB-D"+"robotic".

Nowadays RGB-D sensors are used in robotics applications such as object recognition and tracking, 3D SLAM and navigation, and reconstruction [9]–[13]. The Kinect and similar sensors provide not only 2D color data, but also depth measurements for each pixel. The transition from 2D to high resolution and low cost “2.5D” opened up new opportunities for researchers to develop more sophisticated algorithms.

A clear majority of leading research articles in the field of RGB-D data processing is developed in close relation to the Microsoft Kinect or its direct succesors: the Asus Xtion [14] or Primesense [15] RGB-D sensor series.

Now, seven years after their release, those sensors reached the end of the product life cycle. Recent developments in the field of machine learning try to replace depth sensors with algorithms that rely only on 2D images to estimate the corresponding 3D information. I.e. Eigen et al. [16] show that depth maps can be generated from single images with a multi scale deep network, and very recent research is pushing further into the direction of 3D point and 6D pose estimation using Deep Networks [17]–[19]. However, until today they are far from reaching the accuracy of state-of-the-art depth sensors.

Fortunately, the popularity of depth sensors has led to several successor systems from different manufactures that use similar or different technologies to provide RGB-D data. New sensors using Time of Flight (ToF) or Active Stereo promise to offer similar or even better depth-sensing results and different characteristics for robotics applications (cf. Figure 1.4). However, the sensor manufactures provide no, or only limited, information regarding the sensors’ noise behaviors [14], [20]–[28]. To our knowledge a comprehensive overview that quantitatively evaluates the new sensor systems, including their noise characteristics in the robotics context, is still missing.

Many of the algorithms operating on RGB-D data incorporate the specific noise characteristics of these sensors. For instance, SLAM algorithms include a model of

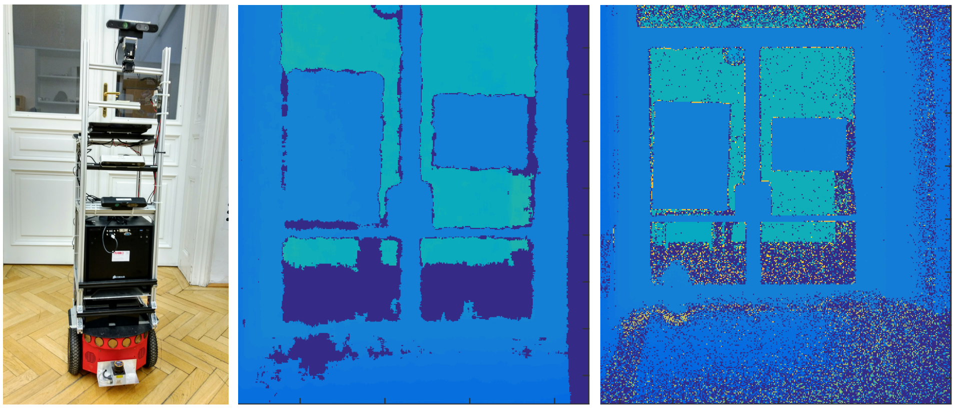


Figure 1.4: Left: Robot used for sensor integration. Middle/right: Depth data gathered with different sensors (middle: Orbbec 3D [21], right: KinectV2 [22]).

the decreasing accuracy (precision and trueness, ISO-15725 [29]) of the measurements to determine the reliability of the measured data, which directly results in better performance [8]. The same accounts for, but is not limited to, object recognition, segmentation, 3D reconstruction, and camera tracking [10]–[13]. Hence, knowing the correct noise models and sensor behavior lead already to better and more reliable results in various fields of robot vision. Also novel data driven methods demand for proper sensor models. Sünderhauf et al. highlighted in [30] the potential of deep networks in the field of robotics. They emphasized the use of existing physic-based models and their combination with deep networks, i.e. to use them for data augmentation and data generation in order to robustify and prepare the methods for robotic vision, where results will lead to real actions in the environment.

Apart from understanding the error modalities of depth sensors, one key step to reach the full potential of RGB-D sensor setups is to relate the depth information pixel-wise to the color information. Many algorithms are tailored to use the rich RGB information e.g. for object detection [31], or as presented in [4] the detection of small particles that would not be visible in the depth data. However, often it is not sufficient to detect something in the image domain. The spatial information about the detected objects or particles might be required, especially if a robot has to interact with the object or manipulate the object. One way to get the spatial information is to precisely assign depth measurements to pixels of the RGB image and using a perspective projection. The projection requires the extrinsic and intrinsic camera parameters of the color camera and the depth sensor. While the extrinsic parameters describe the spatial relation between the RGB and depth camera, or more general two sensors, the intrinsic parameters model the physical properties of the camera system itself, as the focal length, the lens distortion, and the principal point. These parameters are usually estimated in a calibration step with dedicated calibration methods. Most ready-to-use RGB-D sensors on the market are shipped with a factory calibration that provides reasonable registration results. However, the factory calibration is usually far from perfect. The predefined setup limits the possible combination of RGB cameras and depth sensors, and maybe an additional camera is added to the setup, resulting in a proprietary RGB-D

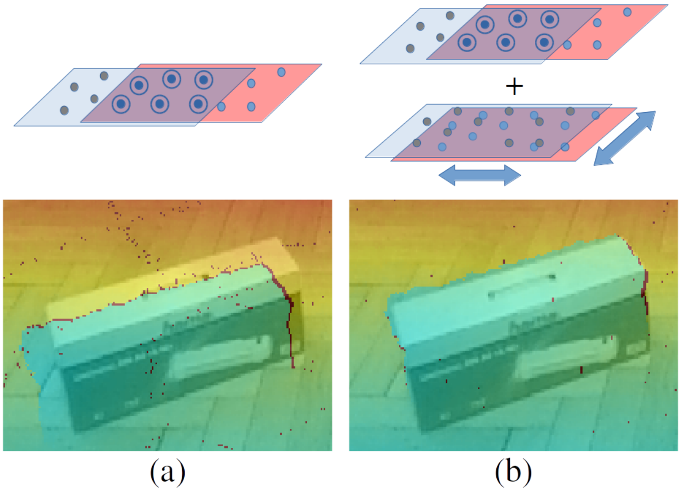


Figure 1.5: Alignment of RGB and depth images using the inverse point-point distance [32] (left) compared to the method presented in this work that adds the point-plane distance (right). It shows improved convergence behavior and a more accurate alignment of RGB and depth images, even on datasets that offer motion blur and low image quality.

system. If the sensor setup is a proprietary RGB-D system, there is simply no factory calibration available. Hence, to enable a precise perspective projection, it is necessary to determine the intrinsic and extrinsic camera parameters.

Usually, the calibration of the sensor setup involves the use of checkerboard patterns and has to be carried out by an expert. The property of the sensor setup might change during operation, what requires a recalibration of the whole setup. Classic methods are not suited for this scenario, and it might be impossible for a robot to autonomously gather the information needed for a recalibration in the field, where it faces uncontrolled environments. Hence, one of our ultimate goals is to develop an autonomous calibration method that enables a sensor (re)calibration on a robot in the field using its own motion without the need for a human expert or artificial calibration targets in order to precisely register RGB and depth images.

Auto calibration methods for RGB-D sensors take one step into that direction and have been addressed in the state of the art [32]–[34]. While those methods work well in controlled environments and with a decent initialization by an expert, they still struggle in real-life robotic indoor scenarios, where they face low image quality, motion blur, and an inaccurate initialization.

Moreover, these methods do not take the calibration of the sensor pose relative to the robot into account although this information is essential to transform the acquired information into the domain of the robot.

To wrap up the motivation this chapter continues with the research questions and gives a short overview of the contributions. After that it continues with a summary of the most important related work and ends with a list of related publications.

All figures of this work are best viewed in color.

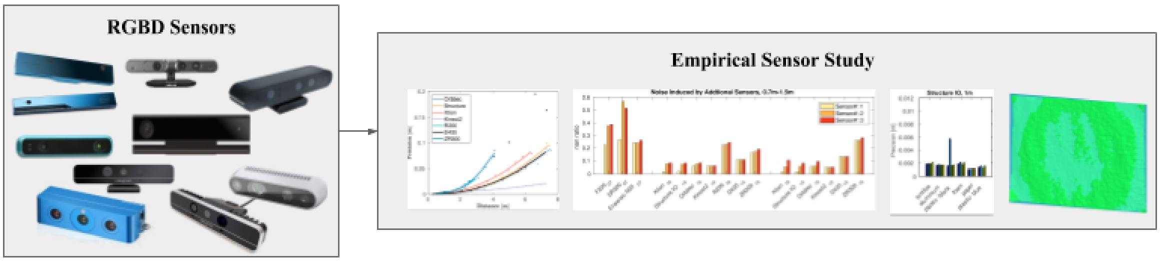


Figure 1.6: The trueness and the precision of ten different depth sensors is evaluated under various conditions. This work has been published in [37].

1.2 Research Questions and Contributions

The introduction of several new depth sensors and their steadily growing popularity triggers our first research question:

1. How do state-of-the-art depth sensors for robotic vision differ from each other? Are there any similarities in their accuracy? How do they impact other sensors in a multi sensor setup, and how do different materials impact the accuracy?

To answer this question, Chapter 2 contributes a comprehensive evaluation and comparison with respect to i) different depth sensors and ii) different metrics (cf. Figure 1.6). To be more precise, this work analyzes ten different sensors in terms of trueness and precision as defined in [8], [35], [36] under various conditions. The experiments are focused on indoor scenarios since sensors that rely on projected infrared patterns to obtain depth data are not designed to deal with incident sunlight. The experiments are tailored to extend the results of [8] and to give a comprehensive and general overview without focusing on certain applications. Hence, several experiments are designed to incorporate different distances, materials, and lighting conditions. This work also investigates interference induced by other sensors, which is of special interest in robotic and multi-robotic systems. The comparison includes seven wide-range sensors (Asus Xtion Pro Live [14], Orbbec Pro [21], Structure IO [20], KinectV2 [22], ReaslSense D435 [23], RealSense ZR300 [24], RealSense R200 [25]) and three near-range sensors (RealSense SR300 [27], RealSense F200 [26], Ensenso N35 [28]).

A comprehensive statistical analysis using 40 experiments, evaluating over 50000 images guarantees an impartial comparison of the different sensors. This work is focused on robotic related machine vision systems, hence all sensors are integrated on a robot (cf. Figure 1.4) and evaluated using their standard configurations. This analysis is of high interest for several robot perception tasks, i.e., where modeling the sensor noise increases the performance of the algorithms. This includes tasks for navigation, manipulation of objects that use RGB-D reconstruction, as well as object detection and recognition. The results have been published in

> G. Halmetschlager-Funek, M. Suchi, M. Kampel, *et al.*, “An Empirical Evaluation of Ten Depth Cameras: Bias, Precision, Lateral Noise, Different Lighting Conditions and Materials, and Multiple Sensor Setups in Indoor Environments,” *IEEE Robotics Automation Magazine*, pp. 1–1, 2018, ISSN: 1070-9932

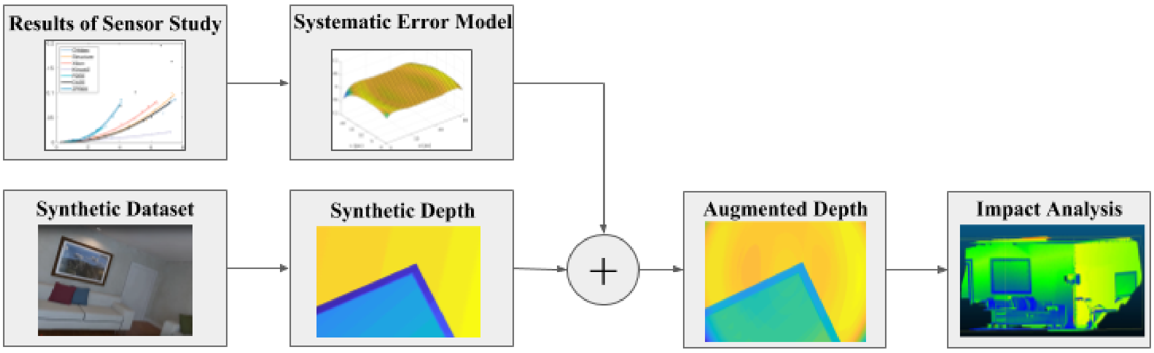


Figure 1.7: Overview of the depth error model that uses a virtual stereo camera pair. It takes synthetic input data and augments them with the error model to get realistic depth information including realistic errors.

The empirical sensor study compares different sensors and analyses their accuracy, but does not answer the research question:

2. How to simulate the systematic error behaviors of the investigated errors? Is it possible to define a generic model? How does the depth error impact other algorithms that rely on accurate depth data?

Hence, Chapter 3 picks up the results of the sensor tests and comes up with an intuitive generic method to simulate pixel-wise as well as image-wide systematic errors of state-of-the-art depth sensors (cf. Figure 1.7). The proposed method reproduces the characteristic sensor behaviors and is designed to be used together with synthetic datasets. The model is evaluated by comparing the synthetically generated images with real-life depth sensor data. The feasibility of the model is demonstrated by analyzing the impact of systematic depth errors on a state-of-the-art reconstruction method.

While some of the investigated depth cameras do not provide access to the intrinsic parameters, others allow only a calibration with manufacturer-specific methods which fail to achieve good results. This leads to the research questions:

3. How to compensate depth errors without modifying the camera intrinsic parameters that are used for the generation of depth images? Is it possible to use a generic model as a post-processing step, and how does it impact other algorithms?

To resolve this challenge, Chapter 4 presents two alternative camera-model-free depth compensation methods that optimize a depth compensation function and give a pixel-wise depth dependent compensation value. Both compensation methods are evaluated on synthetic and on real-life data. One compensation method is used together with a state-of-the-art reconstruction algorithm. The reconstructions with uncorrected and corrected depth data are qualitatively and quantitatively compared against a ground truth model. The chapter ends with a short performance analysis of two different implementations of the compensation method. One using an auto-differentiation technique and one, using analytic derivatives.

The introduced compensation methods can be used in controlled environments only

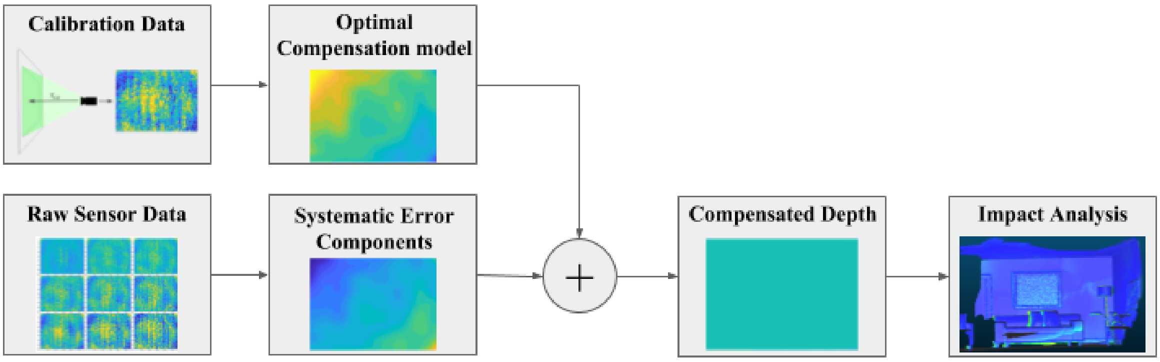


Figure 1.8: Overview of the camera model free depth offset compensation methods.

and demand manual ground truth measurements or special calibration setups. However, together with classic camera calibration methods the RGB-D setup can be calibrated and a precise alignment of RGB and depth data becomes possible. To conclude, rather simple but powerful RGB methods as

> A. Grünauer, G. Halmetschlager-Funek, J. Prankl, *et al.*, "The power of GMMs: unsupervised dirt spot detection for industrial floor cleaning robots," in *Annual Conference Towards Autonomous Robotic Systems*, Springer, 2017, pp. 436–449

can be easily combined with depth information, similar to our early work

> G. Halmetschlager, J. Prankl, and M. Vincze, "Probabilistic near infrared and depth based crop line identification," in *In Workshop Proceedings of IAS-13*, 2014, pp. 474–482

where we combine 3D and near infrared information to come up with a simple but reliable method to segment crop rows for a crop line detection.

However, the calibration of such setups is tedious and requires a trained expert. Since the calibration of the setup has a significant impact on the performance of the system and can change during operation due to mechanical stress or other effects, they might have to be recalibrated in the field by an expert, which might cause high costs and down times. Fortunately, there are also auto-calibration methods for RGB-D setups that do not require manual ground truth measurements or trained experts. Since these methods are still far from being perfect the question is raised

4. How to improve auto-calibration methods in order to enable an autonomous calibration of the robot in the field? Is it possible to incorporate priors to robustify these methods?

State-of-the-art autonomous calibration methods use the information of the environment as calibration target, but do not explicitly incorporate assumptions about the structure itself which impacts the quality of the results. Hence, Chapter 5 aims to extend the state-of-the-art in autonomous calibration methods by incorporating plane priors in two different ways into the calibration method, with the ultimate goal to enable an autonomous calibration of a robotic system in the field. This work introduces the

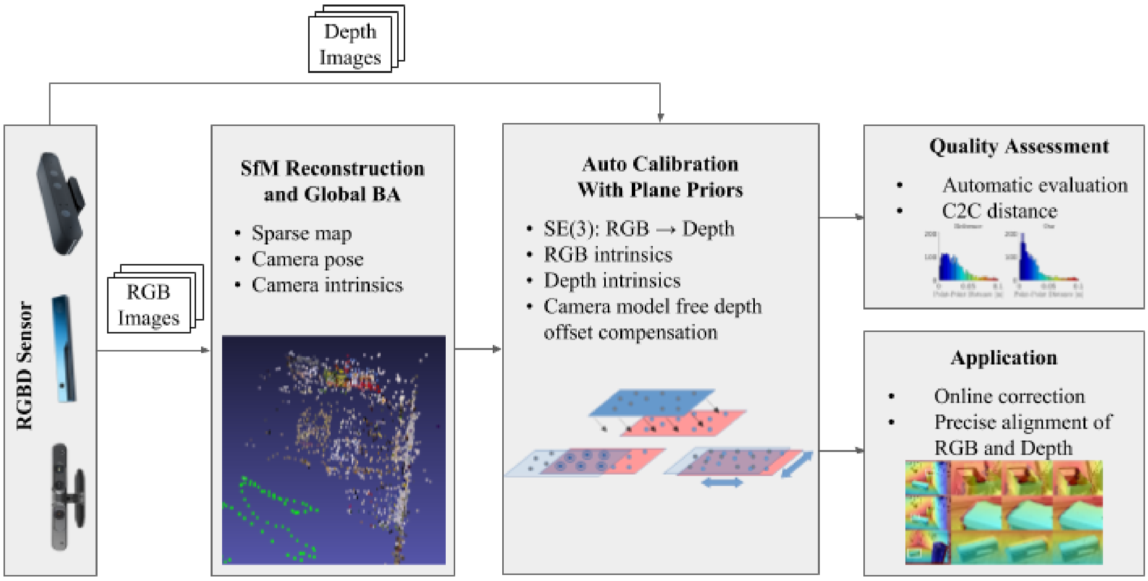


Figure 1.9: Overview of the autonomous auto calibration method that incorporates plane priors for more accurate alignment of depth and RGB images. The autonomous auto-calibration method has been published in [5].

point-plane distance as residual and adds a point-wise plane-based weighting function (cf. Figure 1.9).

It defines a novel quantitative evaluation metric and uses it together with a quantitative metric to evaluate the introduced algorithm in over 300 experiments on ten different datasets. The evaluation demonstrates the benefit of the improved alignment between RGB and depth images in a use case scenario where the output of the algorithm is used to semi-automatically annotate depth images [9]. This part of the thesis has been published in

> G. Halmetschlager-Funek, J. Prankl, and M. Vincze, “Towards Autonomous Auto Calibration of Unregistered RGB-D Setups: The Benefit of Plane Priors,” in *In Proceedings of 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 5547–5554

To relate the spatial information acquired with an RGB-D sensor to another sensor, it is necessary to extract the extrinsic transformation between the sensors. During the manufacturing of a robotic setup these transformations have to be obtained together with the intrinsic parameters of the camera. In practice they are often manually measured and fine-tuned. A holistic calibration of a RGB-D setup should include the calibration of the RGB-D camera, the depth correction, and the calibration of the camera’s pose relative to a reference system. That triggers the research question:

5. *What are potential next steps towards a holistic autonomous auto calibration of RGB-D cameras?*

Chapter 6 gives an extensive outlook how a hand-eye calibration method can be combined with the introduced auto calibration method to form a holistic calibration

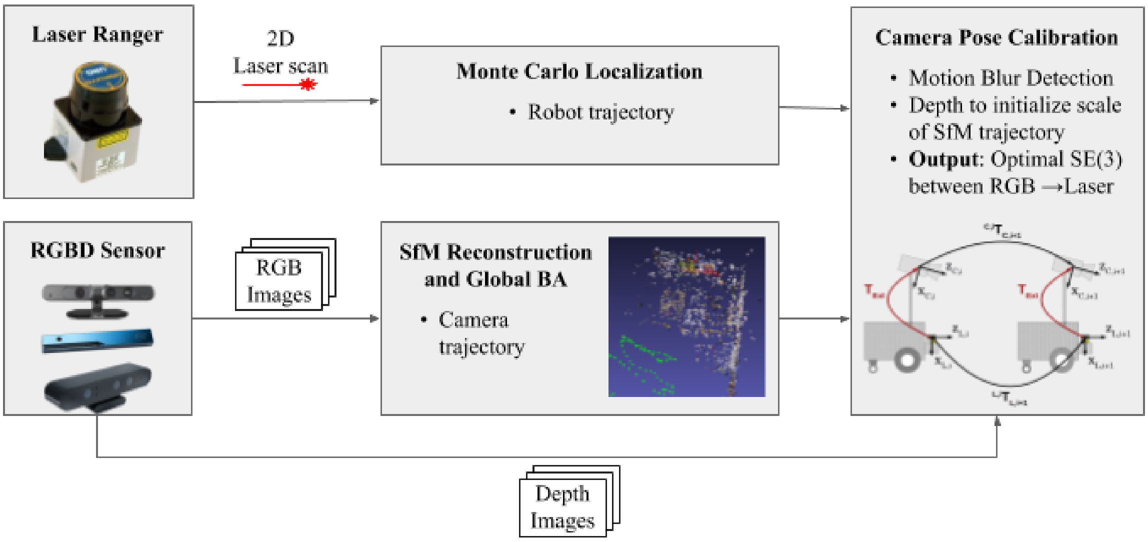


Figure 1.10: Overview of the hand-eye calibration method that estimates an optimal rigid transformation between a laser ranger that is used for navigation and a camera that is mounted on the robot.

method that includes the calibration of the camera pose relative to the robot. This work extends our method presented in

> F. Malekghasemi, G. Halmetschlager-Funek, and M. Vincze, "Autonomous Extrinsic Calibration of a Depth Sensing Camera on Mobile Robots," *Proceedings of the Austrian Robotics Workshop*, pp. 29–35, 2018

by adding first results of an optimal method to estimate the rigid transformation between the camera and a reference coordinate system on the robot, using only the motion of the robot together with the RGB and depth information (cf. Figure 1.10).

Three different classic machine learning methods (Random Forrest, KNN, and Decision Stump) are evaluated together with seven different features to develope a simple motion blur detection algorithm that is used to filter the input images before they are piped into the algorithm. The extrinsic camera pose calibration algorithm is tested on a synthetic dataset and on a real-life dataset. The calibration results are evaluated by using the estimated rigid transformation to align a laser scan with a point cloud from the RGB-D camera and compute the point-point distance between the two scans.

Finally, this work ends with a short summary and points out limits and potentials, as well as possible improvements for the presented algorithms.

1.3 Related Work

This section gives an overview of the state of the art regarding depth sensors, modeling of depth sensing errors, the correction of depth errors, and hand-eye calibration methods.

1.3.1 Sensor Calibration and Depth Error Models

Nguyen et al. [8] investigate different noise characteristics of a Kinect sensor and to our knowledge it is the most important and considered work regarding RGB-D sensor noise. They propose a 3D noise distribution for Kinect depth measurements in terms of axial and lateral noise. Their work describes in detail all experiments and metrics they use to quantize sensor noise in the context of reconstruction and tracking.

Han et al. [40] evaluate the potential of RGB-D sensors for enhanced computer vision tasks. They review the vision methods of data preprocessing, object tracking and recognition, human activity analysis, hand gesture analysis, and indoor 3D mapping. They consider the impact the Kinect RGB-D sensor has had in research and new technical challenges opened up by this sensor. It shows the importance of low cost depth sensing devices for the field of computer vision. Moreover it gives a short introduction to the technology used by the RGB-D sensors.

Andersen et al. provide a detailed analysis of the first Kinect sensor [41]. The conducted experiments use sequences of depth images, allowing a statistical evaluation of bias, precision, resolution, influence of other sensors, and lateral noise. This approach is also used by Smisek et al. in [42] and Pramerdorfer [43]. The main difference of [41] to this work is that the evaluation uses only an original Kinect sensor. Furthermore, they only consider one or at most three distances, depending on the experiment, and measure the influence of a single additional sensor. There is no evaluation of the sensor performance for different materials and lighting conditions.

An evaluation of sensor behavior with respect to multiple materials was performed by Berger et al. [44], including precision measurements on four materials. There are no other evaluation metrics presented in their work.

Foix et al. [45] compared six different ToF cameras. Different to our evaluation the work focuses solely on ToF cameras with respect to their limitations, advantages, and existing calibration methods.

Langmann et al. [46] compare two ToF cameras with a Microsoft Kinect camera. They use a ”Böhler Star” to compare the lateral resolution of the cameras. The depth resolution is evaluate with a sinusoidal surface and the accuracy of the cameras is measured.

The idea of including different materials was also used in the work of Pramerdorfer [43], which evaluates Asus Xtion Sensors regarding their usability in fall detection scenarios. The work includes a detailed description of the experiments conducted to evaluate resolution, lateral resolution, precision, sensor influence by adding one additional sensor, and bias. This work and [8] were the main input for designing the experiments in this thesis.

However, this work extends the experiments with nine new sensors, six different materials, and substitutes the metric for lateral resolution of [43] with a similar measurement for lateral noise as proposed in [8]. Overall, this results in 510 data points for five different metrics.

1.3.2 Camera Calibration

In [47], Yamazoe et al. present a method to model and correct the depth error of projector based consumer depth cameras by performing a correction in the disparity space. They use a distortion model for the projector and the infrared camera. Their calibration method requires a dedicated calibration target.

Aside from the classic calibration methods that utilize checkerboard patterns to achieve a camera calibration, several calibration methods have been developed that fuse color and depth information. Many of them deal with laser range finders or ToF cameras [48]–[50], but are not directly applicable to RGB-D sensors [51]. Fortunately there is also related work specialized on the auto calibration of RGB-D sensors.

Teichmann et al. [33] introduced a depth multiplier image to compensate for depth errors. Their method purely relies on depth data and exploits the property of the sensor that close depth values are more accurate than distant ones. They come up with a depth compensation function that requires 10^4 parameters. The algorithm needs a full Visual SLAM (VSLAM) reconstruction extracted from the depth data. Contrary to our method they only aim to find a way to compensate for depth errors and do not consider the RGB camera.

Similarly, Quenzel et al. [34] use a VSLAM algorithm to compensate depth errors and do not rely on planar geometry or a one-to-one pixel correspondence between color and depth cameras. They use visual features gathered with the RGB camera to triangulate points and compare the depth measurements with the expected distances. In contrast to [32], [33], they introduce an online method that iteratively updates the depth compensation. They compensate the depth values using thin plate splines, which is similar to the bicubic interpolation introduced by [32]. Quenzel et al. demonstrate improved reconstruction results and a sufficiently quick convergence of their calibration method.

Basso et al. [51] address the extrinsic calibration problem by using a calibrated color camera and a planar calibration target. While their method relies on fewer parameters than [33], they still rely on a specific calibration target, a planar checkerboard. Their algorithm shows improved results for the 3D structure estimation and the registration of RGB and depth images. Similarly to [32], [33] the correction method results in a radially symmetric error model for a Microsoft Kinect device and verifies the error observations of Smisek et al. [42].

Zeisl and Pollefeys [32] present a calibration algorithm for the automatic calibration of RGB-D sensors. The difference to other approaches [33], [34], [42], [51] is their use of an SfM reconstruction instead of a depth dependent VSLAM algorithm or a planar checkerboard. They favor the SfM reconstruction over VSLAM under the assumption that constraints in the image domain are more accurate compared to depth. They qualitatively evaluate their results against the factory calibration and a manual calibration of the camera intrinsics by superimposing the depth map on the color image. Our approach closely follows this method and adds plane priors to further improve convergence behavior and the alignment between the RGB and depth image. This work also adds a quantitative evaluation method that allows to make a quantitative assessment of the calibration results.

To conclude, several auto calibration methods for RGB-D sensors have been addressed in the state of the art. While those methods work well in controlled environments and with a decent initialization by an expert, they still struggle in real-life robotic indoor scenarios, where they face low image quality, motion blur, and inaccurate initializations. In contrast to other target-less auto calibration methods [32] utilize an SfM reconstruction that tends to be more accurate for the alignment of RGB and depth images than methods that purely rely on depth.

Our plane based weighting approach is inspired by the work of Rusinkiewicz and Levoy [52]. They focus on Iterative Closest Point (ICP) [53], [54] methods and observe an improved alignment error by introducing a normal space sampling. This work improves the method presented in [32] by adapting the idea of normal space sampling and point pair weighting to produce an adaptive residual weighting that favors a group of normal vectors representing planes over points that do not have a supporting plane. The weighting aims to reduce the number of observations while favoring more reliable and stable points over potential outliers.

1.3.3 Rigid Transformation Calibration Between Sensors

Sinha et al. [55] introduce a method to calibrate the extrinsic parameters of a multi camera setups from two silhouette sequences based on the constraints arising from the correspondence of frontier points and epipolar geometry.

Carrera et al. [56] calibrate a multi-camera rig by performing several independent SLAM reconstructions for every camera and match the resulting maps. The maps are fused together by computing corresponding invariant features and solving for the relative poses using bundle adjustment [57]. Different to Carrera et al. our method is not limited to cameras and can be used together with every sensor that allows to estimate the pose of the robot, i.e. the camera is calibrated relative to the coordinate system used for the localization of the robot.

Miller et al. [58] present a method to calibrate the relative pose of several depth cameras using the unstructured motion of objects in the scene to find potential correspondences between the sensor pair. The acquired transformation is refined with an energy minimization.

Li et al. [59] present an approach to calibrate the extrinsic parameters of multiple depth sensors by using a skeleton tracker instead of an artificial pattern to calibrate the setup. They use the joint positions to solve for the rigid transformation between the cameras.

Other related methods deal with the extrinsic calibration of laser based sensors. Levinson and Thrun [60] introduced a method to calibrate various sensor beams and their corresponding orientation and distance-response function as well as a probabilistic model. Moreover the method allows to recover the extrinsic pose of the sensor relative to the robot coordinate frame by using the 3D information and known poses.

Maddern et al. [61] introduced a method to calibrate 2D and 3D LIDARs. They optimize the extrinsic pose of the two sensors by optimizing a point cloud constructed with the two sensors. Similar to Maddern et al., the method presented in this work aims to calibrate a 2D LIDAR to another 3D sensor, but uses only the robot motion and not full point clouds.

Pandey et al. [62] calibrate the extrinsic parameters between a 3D LIDAR and a camera system. The two sensors are calibrated by maximizing the mutual information obtained between the sensor-measured surface intensities.

Similar to Schmidt et al. [63] the camera pose calibration method uses an SfM method to estimate the camera trajectory. Different to the state of the art, the presented method is tailored to be integrated into the auto calibration framework and allows to calibrate the rigid transformation of two sensors that do not have to contain information from the same domain. It directly uses the trajectories estimated with the sensors to find the rigid transformation between them.

1.4 Related Publications

G. Halmetschlager-Funek, M. Suchi, M. Kampel, *et al.*, "An Empirical Evaluation of Ten Depth Cameras: Bias, Precision, Lateral Noise, Different Lighting Conditions and Materials, and Multiple Sensor Setups in Indoor Environments," *IEEE Robotics Automation Magazine*, pp. 1–1, 2018, ISSN: 1070-9932

G. Halmetschlager-Funek, J. Prankl, and M. Vincze, "Towards Autonomous Auto Calibration of Unregistered RGB-D Setups: The Benefit of Plane Priors," in *In Proceedings of 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 5547–5554

A. Grünauer, G. Halmetschlager-Funek, J. Prankl, *et al.*, "The power of GMMs: unsupervised dirt spot detection for industrial floor cleaning robots," in *Annual Conference Towards Autonomous Robotic Systems*, Springer, 2017, pp. 436–449

G. Halmetschlager, J. Prankl, and M. Vincze, "Probabilistic near infrared and depth based crop line identification," in *In Workshop Proceedings of IAS-13*, 2014, pp. 474–482

F. Malekghasemi, G. Halmetschlager-Funek, and M. Vincze, "Autonomous Extrinsic Calibration of a Depth Sensing Camera on Mobile Robots," *Proceedings of the Austrian Robotics Workshop*, pp. 29–35, 2018

Chapter 2

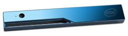
Depth Sensor Error Modalities

This chapter recaps three main state-of-the-art depth sensing technologies used for the application of machine vision algorithms in indoor robotics: Time of Flight (ToF) sensors, Active Stereo cameras, and sensors using a projected infrared pattern to acquire depth information.

Subsequently, the study evaluates ten different near- and far-range depth sensing devices, covering the three introduced sensing technologies (cf. Table 2.1). Thereby, it facilitates thereby state-of-the-art methods [8], [29], [43] to benchmark the sensors with respect to their trueness, precision, lateral noise, under varying lighting conditions, and in multiple sensor setups. The study aims at refelcting the state-of-the-art of depth sensing devices and to quantify their error characteristics. The chapter concludes with a short summary and demonstrates the benefits of incorporating a simple noise model in a state-of-the-art reconstruction method [64].

Table 2.1: Analyzed depth sensors and their specifications given by the manufacturers.

Sensor	**Xtion Pro Live** [14]	**Structure IO** [20]	**Orbbec Pro** [21]	**KinectV2** [22]	**RS D435** [23]
Manufacturer	ASUS	Occipital, Inc.	Orbbec 3D	Microsoft	Intel
Sensor Type	RGB-D	D	RGB-D	RGB-D	RGB-D
Technology	IR pattern	IR pattern	IR pattern	ToF	Act. Stereo
Depth Resolution	640x480	640x480	640x480	512x424	1280x720
Range [m]	0.8 to 3.5	0.4 to 3.5+	0.6 to 8.0	0.5 to 4.5	0.2 to 10
Interface	USB 2.0	USB 2.0	USB 2.0	USB 3.0	USB 3.0

					
Sensor	**RS ZR300** [24]	**RS R200** [25], [65]	**RS F200** [26]	**RS SR300** [27]	**Ensenso N35** [28]
Manufacturer	Intel	Intel	Intel	Intel	Ensenso
Sensor Type	RGB-D	RGB-D	RGB-D	RGB-D	D
Technology	Act. Stereo	Act. Stereo	IR pattern	IR pattern	Act. Stereo
Depth Resolution	628x468	640x480	640x480	640x480	1280 x 1024
Range [m]	0.55-2.8	0.51 to 4.0	0.2 to 1.2	0.2 to 2.0	0.47 to 1.1
Interface	USB 3.0	USB 3.0	USB 3.0	USB 3.0	Ethernet

2.1 Sensing Technologies for Indoor Robotics

This work targets depth sensors that use an active light source together with an image sensor to acquire depth information. One of the biggest advantages of those sensors compared to other depth sensing approaches as LIDARs or laser range finders is that they acquire high resolution depth information in an organized row and column structure. Organized depth data contains an inherent pixel-wise neighborhood information of adjacent points that allows for a significant speed-up of many algorithms. This work focuses on Active Stereo systems, structured-light-like cameras, and ToF sensors.

Stereo cameras are used for several years to acquire 3D information for machine vision applications. They consist of a pair of monocular cameras and estimate the depth using the intrinsic camera parameters, the extrinsic camera parameters, and the pixel offset between an observation in the left and the right camera (disparity). One crucial and not completely solved problem is to find observations in one camera and precisely locate exactly the same observation in the other camera. The detection of the observations relies on texture, which might not be available in many scenarios. Active Stereo cameras solve this problem by projecting a (infrared) pattern on the target that provides the necessary texture to perform a stereo matching algorithm.

Structured Light cameras, as the Kinect, consist of an infrared projector and an infrared-sensitive camera. The projector illuminates the target with a known (dot) pattern that is detected with the infrared camera. Since the extrinsic parameters between the projector and the camera, as well as the camera intrinsics are known, the distance to the illuminated surface can be estimated using a standard triangulation method.

Different to Active Stereo cameras and the very similar Structured Light cameras, ToF cameras measure the distance to an object by measuring the time difference between emitting the light and receiving the reflections from the surface. ToF cameras can be split into two groups: pulsed ToF cameras and continuous-wave ToF cameras. Pulsed ToF cameras illuminate the target for a very short period and accumulate the reflected energy for every pixel. The accumulated electric charges can be used to compute the distance. Pulsed ToF cameras are rather rare and require fast electronics to achieve high accuracy. For instance, reaching an accuracy of 1mm requires a pulse of $6.6 \cdot 10^{-12}$s. In contrast, continuous-wave ToF sensors use the phase angle between the illumination and the reflection to calculate the distance (cf. Figure 2.1). However, continuous wave ToF sensors have the disadvantage that the scene has to be scanned several times with different wave modulations in order to get a wider measurement range. For more details on ToF sensors it is referred to [45], [66], [67].

2.2 Evaluation of Ten Different Depth Sensors

Depth sensors are known to have a growing depth inaccuracy with a growing distance to the target. Moreover, this error is not constant throughout the image, which is hard to quantize and strongly depends on the calibration of the sensor. These local differences and calibration dependent inaccuracies make it difficult to benchmark the

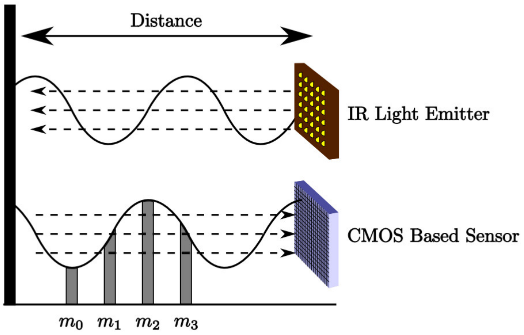


Figure 2.1: Function principle of continuous wave ToF cameras [45].

different sensors. This issue is circumvented by using only the sweet spot in the center of the image to benchmark the different sensors.

2.2.1 Performance Characteristics

This section describes the performance characteristics used to evaluate the different sensors. Following the review of related work and keeping in mind robotics applications such as navigation, object detection, and human machine interaction, this work proposes five different characteristics. These characteristics cover different aspects of accuracy, reflection properties, the response to ambient illumination, and sensor interference. Figure 2.2 depicts the relation between different error types, their characteristics and the corresponding quantitative expressions. The performance characteristics are introduced together with an outline how they capture the application requirements in an objective and measurable way.

Performance Characteristics I and II - Trueness & Precision – Trueness (ISO 5725-1) describes the deviation between the mean distance estimated by the sensor and the ground truth distance. Precision quantifies the standard deviation of the depth measurements. The definition of trueness and precision follows the official definitions of trueness and precision according to ISO 5725-1 [29]. These two discrete values are used to cover the full statistics of the measurement:

$$trueness = \mid d_l - d_o - \mu_d \mid, \tag{2.1}$$

where d_l is the measurement of the laser device, and d_o is the fixed distance offset between the mounted laser device and the tested sensor. μ_d is the average depth defined as:

$$\mu_d = \frac{1}{N \cdot n^2} \sum_{i=1}^{N} \sum_{u,v}^{n} I_i(u,v), \tag{2.2}$$

where N is the number of measurements, n is the size of the measured region, (u,v) is the corresponding coordinate in the 2D depth image I_i. The precision is defined as follows:

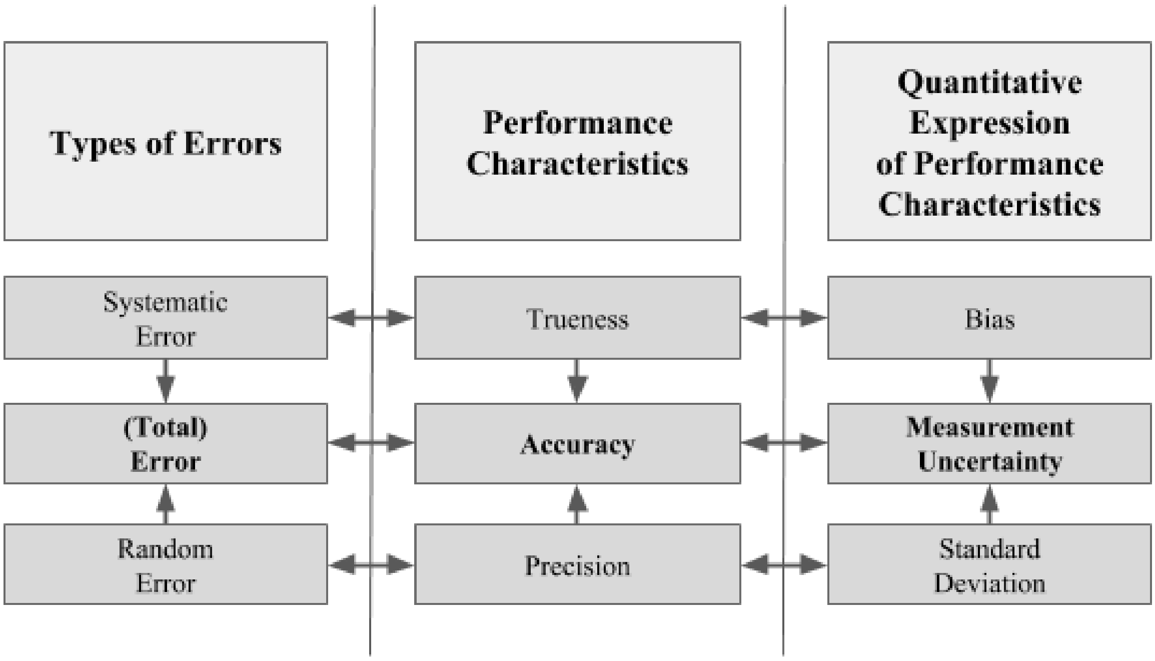


Figure 2.2: Relation between the error types, the performance characteristics and the quantitative expressions [68].

$$precision = \sqrt{\left(\frac{1}{N \cdot n^2} \sum_{i=1}^{N} \sum_{u,v}^{n} \tilde{I}_i(u,v)^2\right) - \tilde{\mu}_d^2}, \tag{2.3}$$

where $\tilde{I}_i$ is the corrected fronto-parallel depth image, and $\tilde{\mu}_d$ is the average depth of $\tilde{I}_i$ using (2.2).

Performance Characteristic III - Lateral Noise – This performance characteristic quantifies the lateral noise around a vertical depth edge as a function of depth. The maximum distance of the image pixels around a depth edge is used to quantify the noise in pixels:

$$latnoise(d) = \arg\max_{p \in P}(\mid \Delta(p,l) \mid), \tag{2.4}$$

where p is an instance in the set of detected edge pixels P within a selected region (using Canny edges [69]), l is the least mean squares fitted line representing the edge, and $\Delta(.)$ is the pixel-line distance function.

The lateral noise may be transformed into a lateral resolution by using the depth, the lateral noise in pixels, and the calibration parameters of a sensor together with its projective geometry. In other words, this performance characteristic evaluates the precision of the sensor to quantize spatial expansions of objects and scenes in the image space (while the pixel value gives the depth measurement).

Performance Characteristic IV - Lighting & Materials – This performance characteristic evaluates the precision depending on the reflectivity and absorption behavior of different materials in combination with the influence of ambient light. This performance characteristic indicates the performance of the sensor for different materials and under different lighting conditions.

Performance Characteristic V - Multiple Sensors – This performance characteristic quantifies the precision of a sensor in a multiple sensor setup and the number of invalid values in relation to the full sensor resolution (nan ratio). This is motivated by the fact that sensors using the same measurement technology tend to interfere with each other [27]. In other words, this performance characteristic measures the capability of the sensor to deal with multi-sensor setups occurring in the field of robotics on a regular basis.

Why do the performance characteristics capture the requirements of e.g. SLAM or reconstruction? SLAM and reconstruction algorithms include a model of the decreasing precision and trueness of the measurements to determine the reliability of the measured data and to incorporate the noise characteristics. It has been shown that incorporating a noise model results in better performance [8]. The same accounts for, but is not limited to, object recognition, segmentation, 3D reconstruction, and camera tracking [10]–[13]. Although the main contribution of this thesis is an extensive evaluation of ten different sensors, the relevance of the results are highlighted by showing preliminary qualitative results of the reconstruction algorithm introduced in [64] that incorporates the parametric error model.

2.2.2 Experimental Setup

This section describes the setup of the experiments and the method used to produce results for the different performance characteristics introduced in the previous section.

There exist various publications regarding sensor calibration and depth offset compensation methods [32], [33]. This work directly benchmarks the sensors and not the underlying calibration methods, therefore it uses the raw sensor data together with the factory calibration. However, it should be noted that this chapter is highly related to sensor calibration. Most calibration methods rely on information regarding the noise characteristics of raw sensor data to adapt the underlying noise and/or error model. This chapter provides the necessary information.

The data is gathered using a mobile platform equipped with the Robot Operating System (ROS) [70] and the publicly available ROS wrappers for the used sensors (cf. Figure 2.3). The ground truth measurements are obtained using a laser range measurement device. In addition, a luxmeter and a strong construction light are used to evaluate the sensors' capabilities under different lighting conditions.

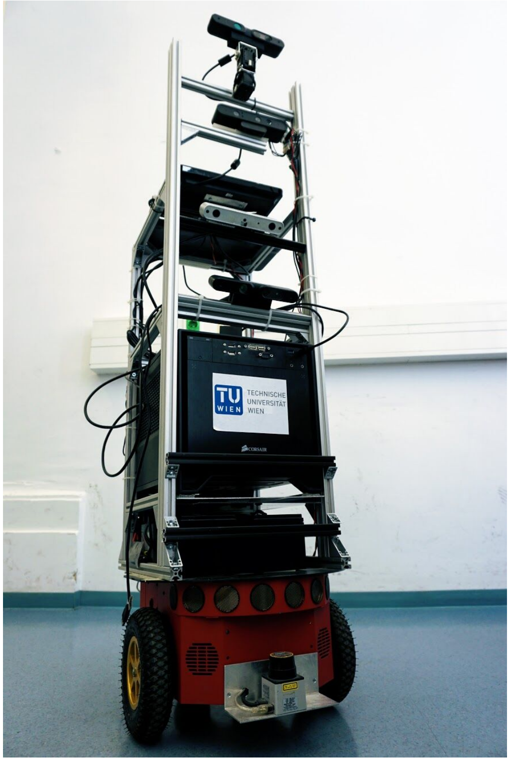


Figure 2.3: Robotic s other wordsystem.

Experiment I/II - Trueness, Precision & Lateral Noise

Gathering data for far-range sensors is done starting from the shortest distance at which the sensor is able to gather depth information up to the furthest distance ($\sim 7.0m$) using a step size of $0.5m$. For near-range sensors, measurements are conducted from approximately $0.3m$ to $2.0m$ with a step size of $0.1m$. The distances are validated using a laser measurement device. The depth offset between the sensor and the laser measurement device is determined manually and taken into account for the experiments. The sensor is positioned parallel to a planar surface. For each measurement, a region of 20×20 pixels on the target is recorded for 100 frames to make sure that temporal noise is included in our evaluation. The ground truth laser measurement is subtracted from the mean distance value obtained from the 100 frames to calculate the trueness.

For the precision, a plane is fitted to the target area to compensate for the non-exact parallel alignment of the sensor to the target area. This achieves a fronto-parallel sensor image. The obtained standard deviation gives a new data point for the precision of the sensor at the current distance.

For both the trueness and the precision, a parametric error model is fitted:

$$f(d) = p_0 + p_1 \cdot d + p_2 \cdot d^2, \tag{2.5}$$

where d represents the depth and p_0,p_1,p_2 are the coefficients of the quadratic error model.

The determined error models for every sensor and the collected numerical data are

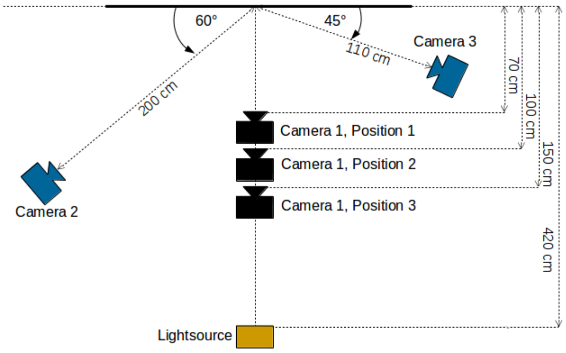


Figure 2.4: Setup for multi-sensor and material tests.

publicly available on our web page[1].

Similar to [8] the lateral noise is determined using the sharp vertical edge of the target. First, a region of the depth map that contains the vertical edge of the target is manually selected . Second, that edge is detected using Canny edges [69] and a line model is fitted to the obtained pixels using least mean squares. This enables the determination of the distance of each edge pixel to the fitted edge.

Experiment III - Lighting & Materials

In this setup, six different materials under four different lighting conditions ($4lux$, $36lux$, $277lux$, $535lux$), at distances $0.7m$ (near-range), $1.0m$, and $1.5m$ (far-range) are tested. The different lighting conditions are achieved by adding three light sources, one after the other, consisting of two ambient office lights and one strong spotlight.

The materials are chosen to cover a wide variety of reflective characteristics. This includes aluminum, black plastic, blue shiny plastic, foam, paper, and textile. The sensor is placed parallel to the objects. For each distance, object, and lighting condition, a region of 20×20 pixels is measured on the objects for 100 frames. The schematic of the experimental setup is depicted in Figure 2.4.

Experiment IV - Multiple Sensors

Simulation of interference of additional sensors is achieved by placing one additional sensor at a distance of $2m$ and an angle of 60°, and another at a distance of $1.1m$ and an angle of 45° to the object (cf. Figure 2.4). The interference measurements are conducted by adding one sensor after the other. Each measurement, consisting of 100 frames, is taken from a planar surface parallel to the sensor.

2.2.3 Results

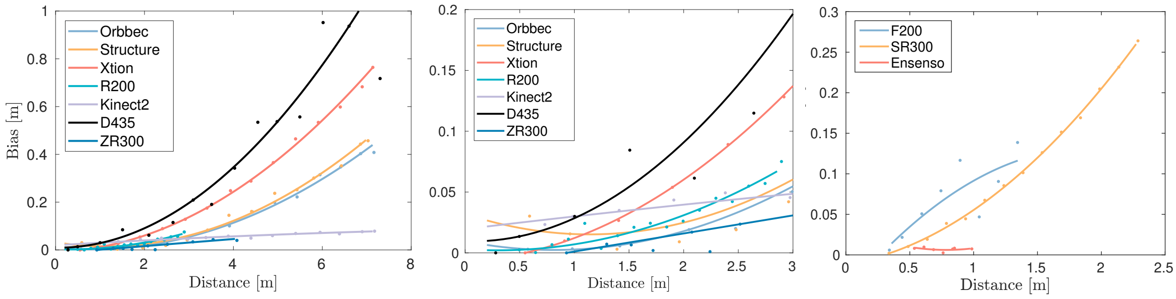
This section gives a comprehensive overview of the results achieved by the experiments including interpretations and explanations.

---

[1] `https://www.acin.tuwien.ac.at/RGB-D-sensor-tests/`

Trueness

While the KinectV2 offers a trueness over the whole range, a significant deterioration of the trueness is observed for sensors using Structured Light starting from $d > 3m$. While all three Structured Light sensors and the two Active Stereo cameras ZR300 and D435 offer a better trueness than the KinectV2 for distances $d < 1m$, three sensors (ZR300, Orbbec, Structure IO) offer an even better trueness for depth values $d < 2.5m$. The results show a quadratic deterioration of the trueness for all sensors (Full range: $d = 0 - 8m$, Figure 2.5a; Zoom-in: $d = 0m - 3m$, Figure 2.5b). The near-range sensors F200 and SR300 (cf. Figure 2.5c) show a slightly worse trueness than their far-range counterparts, while Ensenso N35 provides a good trueness over the whole measurement range.



(a) Far-range sensors.

(b) Far-range sensors, zoom-in.

(c) Near-range sensors.

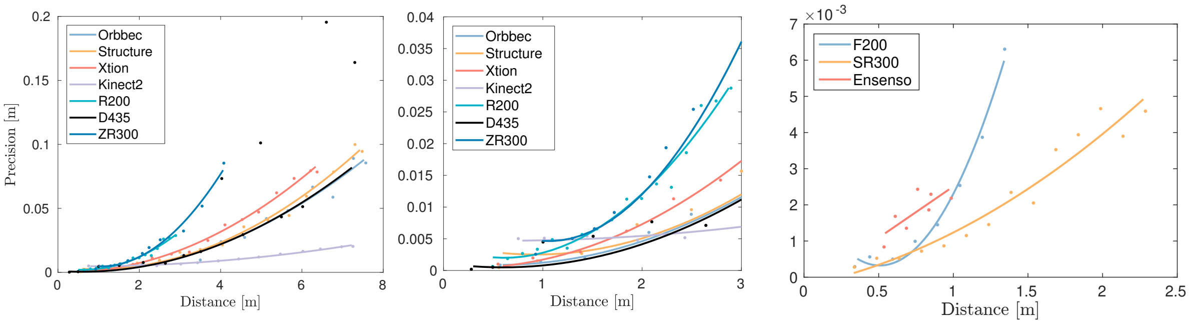
Figure 2.5: Trueness results for near and far range devices. Lower is better.

Precision

A quadratic decrease of precision is found in all far-range sensors (Full range: $d = 0 - 8m$, Figure 2.6a; Zoom-in: $d = 0m - 3m$, Figure 2.6b), but the Structured Light sensors differ in scale compared to KinectV2. Overall the R200 and the ZR300 sensors have the worst performance, while the Structure IO and Orbbec sensors perform very similar. For distances at $d < 2.0m$ all Structured Light sensors generate less noisy measurements than the KinectV2. Moreover, the D435 is able to gather more precise results than the KinectV2 at distances $d < 1m$. The precision results for the D435 are more scattered than for the other sensors. The near-range sensors (cf. Figure 2.6c) experience noise levels up to $0.007m$. In the ranges specified by the manufacturers the experiments obtain precision values under $0.004m$.

Lateral Noise

The analysis of lateral noise shows similar results for the three far-range Structured Light sensors and distances. For $d < 3m$ the noise level is independent of the distance, with 3 pixels for the Structured Light sensors and 1 pixel for the KinectV2 (cf. Table 2.2). Two Active Stereo sensors (D435 and ZR300) offer a similar low lateral noise level as



(a) Far-range sensors. (b) Far-range sensors, zoom-in. (c) Near-range sensors.

Figure 2.6: Precision results for near and far-range devices. Lower is better.

the KinectV2. The R200 achieves a lower lateral noise of 2 pixels for distances closer than $2m$. In the near-range the Ensenso N35 achieves the highest lateral noise value.

Materials

A total of 384 data points are gathered to determine how the precision of the sensors is influenced by the reflection and absorption properties of six different materials in combination with four different lighting conditions from $4.2lux$ to $535.75lux$ (cf. Figure 2.10).

The tests reveal that the Structure IO sensor handles the varying object reflectances and lighting conditions the best. Although it has a lower precision compared to the other sensors for distances $d > 1.5m$, it is able to gather information for high reflective surfaces, such as aluminum, and under bright lighting conditions. While the Structure IO sensor gives a dense depth estimation, the Xtion is not able to determine a depth value. It is also notable that the Orbbec completely fails to gather depth information for four out of six surfaces under bright lighting conditions. The KinectV2 fails to gather reliable depth data for aluminum at distances of $d = 1m$ and $d = 1.5m$ under bright lighting conditions. The F200 and SR300 have a significantly lower precision for bright lighting conditions. During the setup of the experiments we expected the

Table 2.2: Lateral noise of the different sensors.

	Lateral Noise [pixel]		
Sensor	**d=0..0.7m**	**d=0.7..3m**	**d=3..5m**
Asus Xtion	-	3	2
Structure IO	-	3	2
Orbbec	-	3	2
KinectV2	-	1	1
D435	1	1	2
ZR300	-	0.5-1.2	-
R200	-	2-3	-
F200	1.5	-	-
SR300	2	-	-
Ensenso	3	-	-

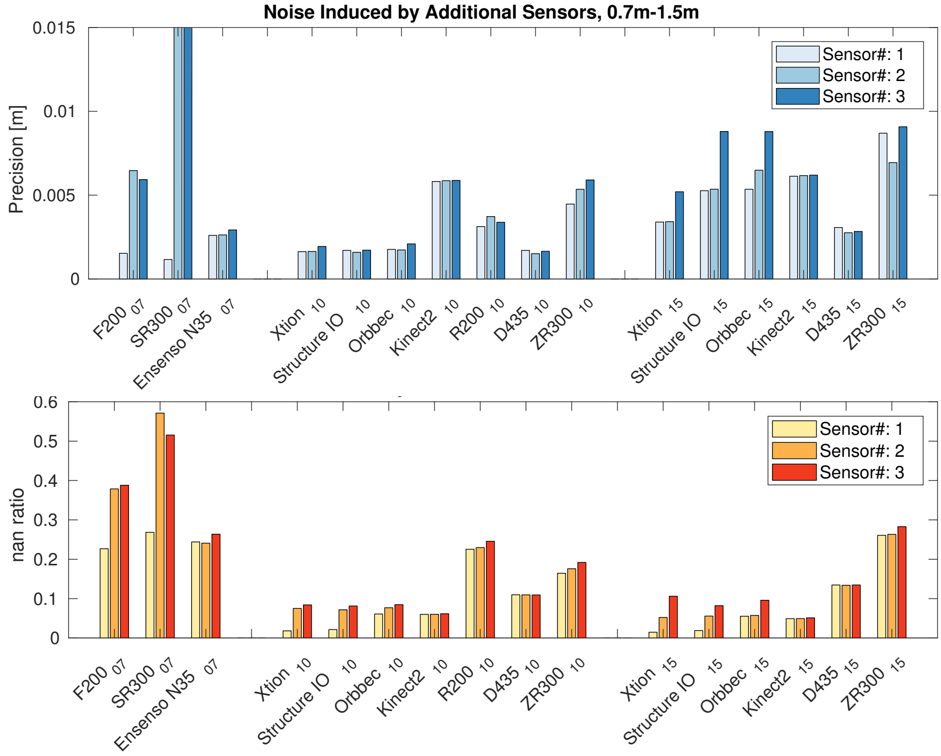


Figure 2.7: Precision and nan ratio (lower is better) in multi sensor setup. Indices represent distance to the target: 07= $0.7m$, 10= $1m$,15= $1.5m$.

Active Stereo cameras (Ensenso, R200) to be able to handle different lighting conditions better than the Structured Light sensors due to the nature of their technology. This expectation is partially fulfilled.

Additional Sensors

The results (cf. Figure 2.7) reveal that the far-range Structured Light sensors can handle noise induced by one and two additional sensors. One exception is when the distance to the target is $d = 1.5m$ and two additional sensors are introduced to the scene. A similar effect is not observed for the KinectV2. The sensor gives stable results for the precision independent of one or two additional sensors. The near-range sensors F200 and SR300 are significantly less precise with an additional sensor. The Ensenso N35 is only slightly affected by a third observing sensor. At this point it has to be noted that the high nan ratio for the close range devices can partially be derived from the setup. Half of the scene is out of the sensor's range (cf. Figure 2.8).

In summary, the first experiment with one sensor provides a baseline for the measurements with two and three sensors observing the scene. The first differences are already visible if only one sensor is added. In particular, the SR300 and F200 show a significant increase in the nan ratio if another RealSense device is added to the scene. For a closer analysis the corresponding depth images are depicted. In Figure 2.8 it is clear that the depth extraction is heavily influenced by an additional sensor. The Ensenso and KinectV2 are nearly unaffected by the additional sensors.

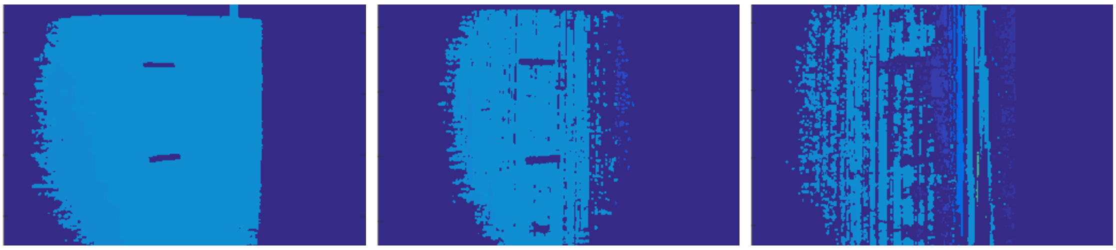


Figure 2.8: Influence of additional sensors. SR300. Number of sensors observing the scene, from left to right: 1,2,3.

Use Case

Schreiberhuber et al. [64] develop a scalable reconstruction method that uses a mesh to represent surfaces. In their work they incorporate the presented error model for the precision of the RGB-D sensor. As an outcome, they show a significant quality improvement of the reconstruction (cf. Figure 2.9).

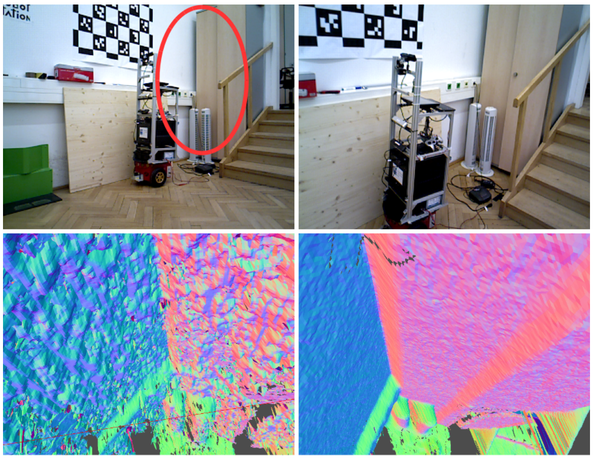


Figure 2.9: Reconstruction method from [64]. The two RGB images show the same location observed from different distances. The bottom images show the reconstruction results of the highlighted region (red circle), without (left) and with (right) the introduced error model.

2.3 Discussion

The three far-range sensors using Structured Light show similar results for trueness, precision, lateral noise, and noise induced by additional sensors. Their precision differs for different object properties and under varying lighting conditions. While the Structure IO sensor gathers valid depth data under all lighting conditions for all materials, it shows a slightly lower precision than the other sensors. The Orbbec sensor fails to gather data under bright lighting conditions for four out of six materials at a distance of $1m$. The difference in performance under bright lighting conditions may be related to the built-in IR cameras, their dynamic range, and the performance of the auto exposure.

The RealSense R200 achieves a similar trueness than the Structured Light sensors, the ZR300 shows a smaller trueness than the Structured Light sensors for $d < 2m$. However, the ZR300 appears to be less precise than the Structured Light sensors, independent of the target material. Moreover, they fail to gather depth data under bright lighting conditions.

The Microsoft KinectV2 behaves significantly different compared to the other sensors. The KinectV2 outperforms all sensors regarding trueness, lateral noise, and precision for $d > 2m$. For the range of $0.7m < d < 2m$ the KinectV2 is less precise than the Structured Light sensors. To conclude, the data provided by the KinectV2 is less smooth and generates worse surface representations for mid-range depths.

The D435 gives scattered results for the precision and the trueness. For some distances, the sensor achieves a similar trueness and precision as the Xtion. However, other distances show large outliers. Due to the nature of the used Active Stereo technology the D435 is not influenced by additional sensors.

For the near-range devices the experiments reveal that the F200 and SR300 sensors are not able to handle noise induced by additional sensors. Their precision and nan ratio are significantly influenced if a second sensor is added to the scene. In terms of precision, the F200 and SR300 are superior compared to the Ensenso N35 Active Stereo system. For all other performance characteristics the Ensenso N35 outperforms the two sensors.

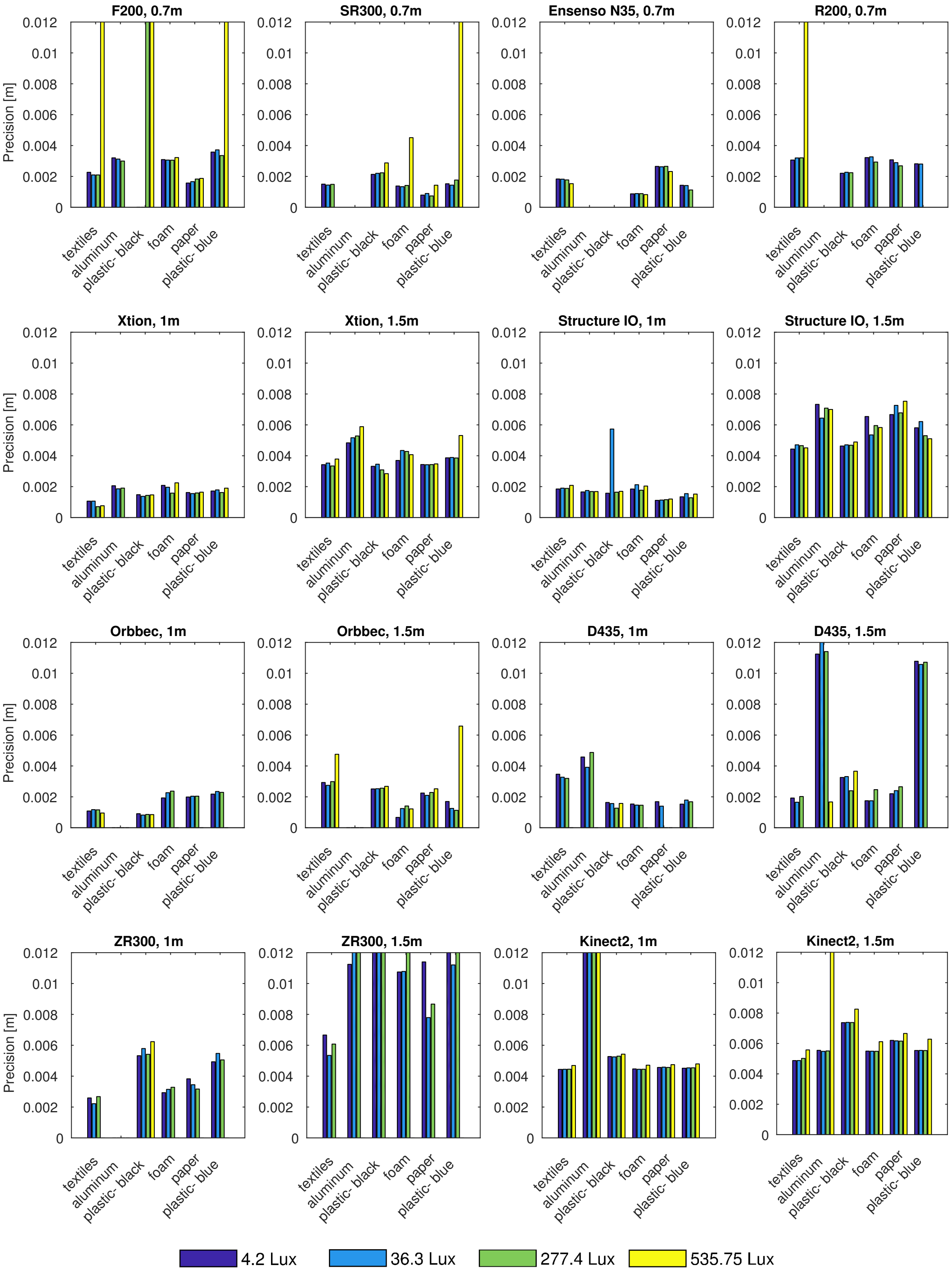


Figure 2.10: Precision for different materials and lighting conditions. Lower is better. A precision of zero indicates that the sensor is not able to gather any depth information.

Chapter 3

Simulation of Systematic Depth Errors

The sensor study presented in Chapter 2 revealed that several depth sensors show an exponential increase of the trueness and precision across different manufacturers. Additionally, the sensors show pixel-dependent (local) variations of the depth map accuracy (cf. Figure 3.1).

In theory, a well parameterized calibration model should be sufficient to compensate these errors. Such a perfect calibration requires dedicated setups in a controlled environment and might not be possible in practice. Especially if the intrinsic parameters change during operation due to mechanical stress, a recalibration in the field might not be possible. Moreover, it is not trivial to determine if a formally decent calibration is still valid or not.

To conclude, it is very likely that algorithms have to deal with depth inaccuracies in the field. One crucial step to bulletproof and robustify new algorithms is to test them with large, real-life datasets. These datasets have to include realistic depth estimates as well as ground truth information to enable an impartial analysis of algorithms. Especially, novel data driven methods would require such realistic large scale datasets to learn robust reliable algorithms that make reliable decisions, which directly result in actions in the real world [30]. Unfortunately, large, real-life, ground-truth datasets are very rare and hardly available. This problem is often circumvented by utilizing

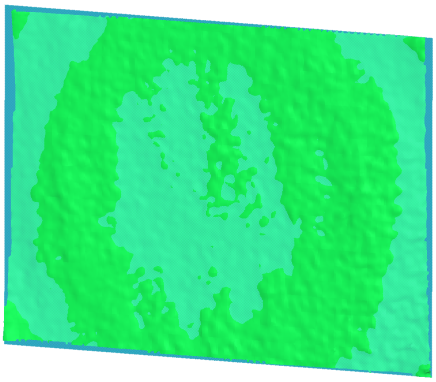


Figure 3.1: Depth data of a flat wall gathered with a factory calibrated R200 (green). The fitted plane highlights the radial error due to an imperfect calibration. With courtesy of DAQRI.

large synthetic datasets that provide the necessary ground truth information, but lack realistic depth data and will give results that are far from the algorithm's performance under real-life conditions. One way to bridge the gap between synthetic and real-life data is to make the synthetic data as realistic as possible, e.g. by incorporating sensor models. We believe that modeling the exact systematic error behavior is a crucial step towards the development of new robust algorithms. To our knowledge, there are hardly any generic depth error models available that replicate all error behaviors that have been observed in the empirical sensor study.

This chapter contributes a generic camera oriented model to simulate simple and complex systematic depth errors. The method is inspired and derived from the principals of stereo imaging and replicates the results of the sensor study.

The next sections present and evaluate a new comprehensive simulation method.

3.1 Error Model

Chapter 2 revealed that state-of-the-art depth sensors show a linear and/or exponential increase of the precision and trueness over the measured distance, which can be modeled with a second order polynomial. This quadratic error model is used to model errors of a small sub region $(u,v) \in S$ of the image, such that the depth error $e \sim N(\mu,\sigma^2)$ becomes independent of the pixel location:

$$\{S \subset U \quad | \quad e = f(u,v,z) \approx f(z)\}, \tag{3.1}$$

where S is a small subset of the set of all pixels U, e is the error and z is the depth. This simplified depth measurement error only gives a rough estimate of the sensor's accuracy. That is sufficient to benchmark the sensors, but is deficient if the error has to be simulated in detail.

Why is it deficient? In addition to the linear and non-linear increase of the trueness and precision, the sensor study presented in Chapter 2 revealed, systematic, radial shaped local variations of the trueness for a tested Active Stereo Camera (cf. Figure 3.2). Similarly, Teichmann et al. [33] and Smisek et al. [42] document a variation of the trueness across the image for a Structured Light sensor. Teichmann et al. locate the source of this error at the projected pattern, which cannot be influenced by the user and causes the error to persist in the raw depth data. It is simply impossible to cover this error with (3.1) because it does not cover more complex systematic errors behaviors that e.g. lead to locally variable errors across the depth image. Hence, it is necessary to come up with a new model that is able to simulate all of the observed properties:

- Linear increasing trueness and precision
- Non-linear increasing trueness and precision
- Local variations of the trueness to simulate a radial shaped, oscillating error.

Different to (3.1) and [8], [37], the depth error is formally modeled as

$$\{\forall(u,v) \in U \quad | \quad e \leftarrow f(u,v,z)\}, \tag{3.2}$$

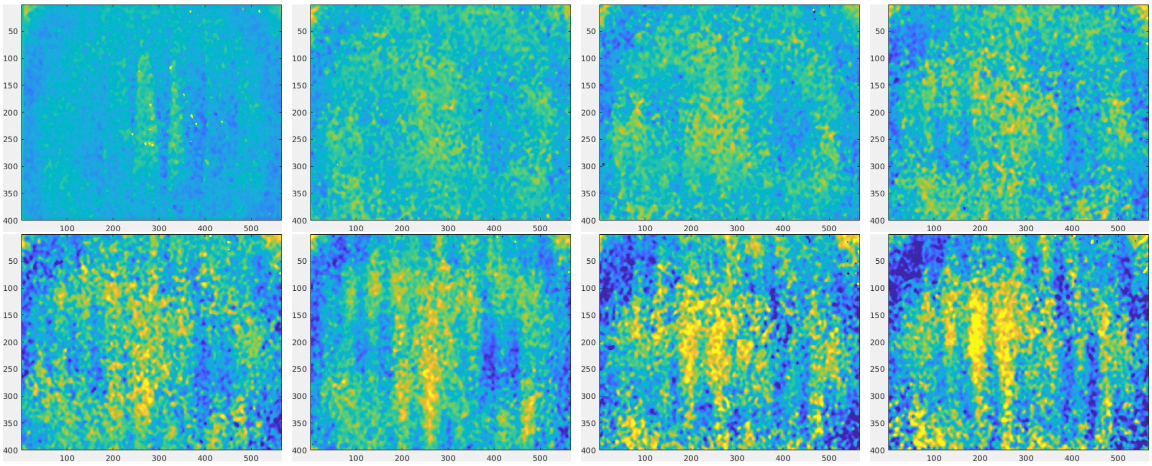


Figure 3.2: Total error of a R200 depth camera after factory calibration. The images show a systematic radial shaped depth error that increases over the depth and additional random noise.

where U is the set of all pixel coordinates, and e is the depth error that depends on the pixel coordinates ($\{u,v\}$) and distance (z).

One of the evaluated Active Stereo cameras showed a combination of all observed error modalities. That suggests the assumption that a virtual stereo camera with bad calibration parameters can be used to model these errors. Figure 3.3 gives an overview of the proposed method.

The method takes synthetic depth images of a virtual depth camera with known intrinsic parameters as input data.

First, the depth image is unprojected to a 3D point in the coordinate system of the left camera image by inverting the standard pinhole camera model

$$\begin{gathered}\mathbf{u} = \pi_{GT}\left(.\right) = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \end{bmatrix} \begin{bmatrix} \mathbf{x_n} \\ 1 \end{bmatrix} \\ \mathbf{x_n} = \left[x \cdot z^{-1}, y \cdot z^{-1}\right]^T, \end{gathered} \tag{3.3}$$

with f_x,f_y as the focal length, c_x,c_y as the principle point, and x,y,z as the 3D coordinates, such that

$$\mathbf{x} = \pi_{GT}^{-1}\left(.\right), \tag{3.4}$$

with $\mathbf{x} = [x,y,z]^T$ as a point in the coordinate system (COS) of the left camera.

Second, the 3D points are back projected onto the left and right image plane using a standard pinhole model. The pinhole model is modified during the projection with the error coefficients ϵ_n and includes a radial and tangential lens distortion model that will introduce different disparity errors across the image. These disparity error will directly result in depth errors. However, before the 3D points $\mathbf{x}_l$ are reprojected into the right camera, they have to be transformed into the right camera COS with

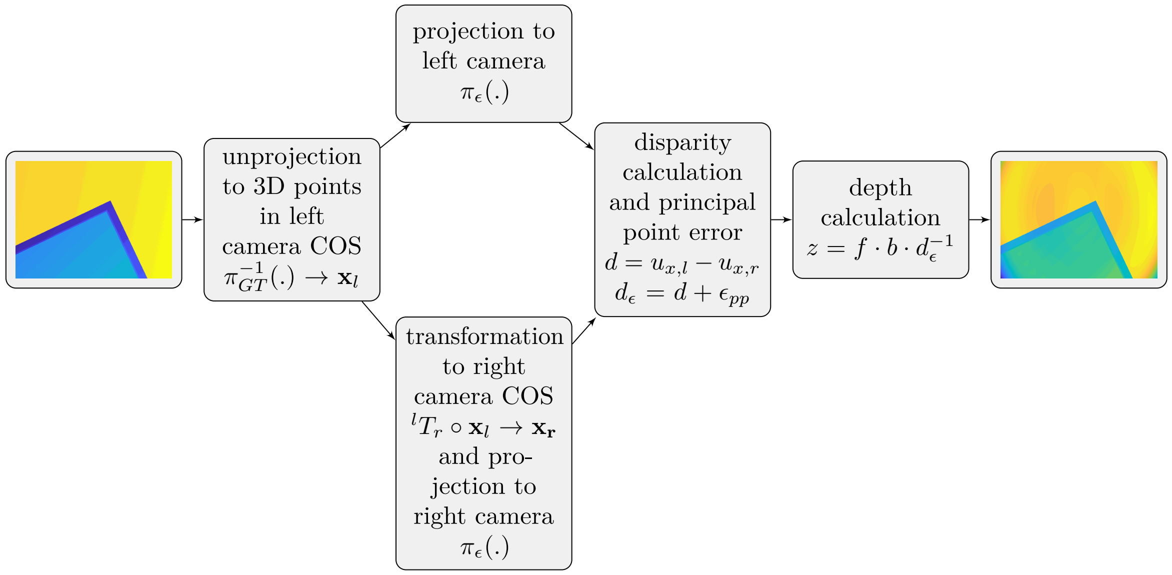

unprojection to 3D points in left camera COS $\pi_{GT}^{-1}(.) \rightarrow \mathbf{x}_l$
projection to left camera $\pi_\epsilon(.)$
transformation to right camera COS ${}^lT_r \circ \mathbf{x}_l \rightarrow \mathbf{x_r}$ and projection to right camera $\pi_\epsilon(.)$
disparity calculation and principal point error $d = u_{x,l} - u_{x,r}$ $d_\epsilon = d + \epsilon_{pp}$
depth calculation $z = f \cdot b \cdot d_\epsilon^{-1}$

Figure 3.3: Stereo error model overview. From left to right: First, the depth image is unprojected into the 3D space using the ground truth camera parameters. Second, the 3D points are back projected onto the left and right image plane, using the modified camera model, including lens distortion. Third, the depth image is recomputed based on the disparity values, the altered focal length, and modified baseline.

$$\begin{aligned} \mathbf{x}_r &= {}^l\mathbf{T}_r \circ \mathbf{x}_l \\ \text{s.t.} \quad {}^l\mathbf{T}_r &= \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \quad \mathbf{R} \in \mathrm{SO}(3) \\ \mathbf{R} &= \mathbf{E}, \quad \mathbf{t} = \begin{bmatrix} -b & 0 & 0 \end{bmatrix}^T \end{aligned} \tag{3.5}$$

where $\mathbf{x}_r$ are the points in the right camera COS, $\mathbf{x}_l$ are the 3D points in the left camera COS, $\mathbf{R} \in \mathrm{SO}(3)$ is the rotational part of the transformation, $\mathbf{t}$ is the translation part of the transformation, and b is the baseline of the camera.

Since the same camera model is used for the left and right camera, the principal point is not altered at this point, because it would not influence the resulting disparity.

$$\begin{aligned} \mathbf{u} = \pi(\mathbf{x}) &= \begin{bmatrix} f_x \epsilon_{fx} & 0 & c_x \\ 0 & f_y & c_y \end{bmatrix} \begin{bmatrix} \check{\mathbf{x}} \\ 1 \end{bmatrix} \\ \text{s.t.} \quad \check{\mathbf{x}} &= L(r^2)\mathbf{x_n} + \mathbf{t}(\mathbf{x_n}) \\ \mathbf{x_n} &= \begin{bmatrix} x \cdot z^{-1}, y \cdot z^{-1} \end{bmatrix}^T \qquad r^2 = \|\mathbf{x_n}\|_2^2 \\ L(r^2) &= 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 \\ \mathbf{t}(\mathbf{x}) &= \begin{bmatrix} 2xy & r^2 + 2x^2 \\ r^2 + 2y^2 & 2xy \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}. \end{aligned} \tag{3.6}$$

After the projection the disparity values are calculated for each 3D point, using the u_x coordinate of its observation in the left image $u_{x,l}$, respectively right image $u_{x,r}$ with

$$d = u_{x,l} - u_{x,r}, \tag{3.7}$$

where d is the calculated disparity. To simulate a misplaced principal point, a constant pixel offset ϵ_{pp} is added. The biased disparity value is calculated with

$$d_\epsilon = d + \epsilon_{pp} \tag{3.8}$$

Finally, the resulting disparity image is transformed to a depth image with

$$z = \frac{f_x \cdot b \cdot \epsilon_b}{d_\epsilon} \tag{3.9}$$

where z represents the depth value, f_x the ground truth focal length, b the ground truth baseline, and d_ϵ the erroneous disparity value.

So far the model does not take the finite resolution of the disparity values into account. Classic stereo vision algorithms derive the location of the correspondences from n discrete pixels coordinates. With that, the resolution of a feature location is limited to $\frac{1}{n}[px]$. To simulate this quantization error the pixel locations $u_{x,r}$ and $u_{x,l}$ are discretized with

$$\hat{u}_x = \frac{\text{integer}(u_x \cdot n)}{n} \tag{3.10}$$

where $\hat{u}_x$ is the discretized pixel value.

To simulate a potential smoothing of the depth images, the synthetic depth maps are downscaled to half of their size and subsequently upscaled back to their original size using a bilinear interpolation.

To simulate the non-overlapping field of view of the stereo pair, the very left border pixels of the images are set to zero. The size of the zero-pixel region is selected to correspond to a target at a distance of $z = 0.5m$, which is approximately the lower bound of the operational range of a real-life far-range depth sensor.

With that the error model is defined and covers errors due to: disparity offset (wrong principle point), focal length error, baseline error, lens distortion, quantization, and depth interpolation.

3.2 Evaluation

To get realistic distortion and camera parameters, the intrinsic parameters of an Active Stereo camera are recovered using a standard calibration method.

The retrieved parameters are summarized in Table 3.1 and used for the evaluation of the error model.

The error model is applied to 66 synthetic depth maps from $0.5\,\text{m} \leq z \leq 7\,\text{m}$ with a step size of $0.1\,\text{m}$. To quantify the impact of the different errors, the RMSE and the

Table 3.1: Virtual Camera Parameters.

Tangential Distortion	t_1	-0.0043
	t_2	-0.0021
Radial Distortion	k_1	0.069
	k_2	-0.5463
	k_3	1.0740
Focal Length	$[f_x,f_y]$	[445.4344,444.9822]
Focal Length Error	ϵ_f	1.03
Baseline	b	0.07
Baseline Error	ϵ_b	1.03
Principle Point	$[c_x,c_y]$	[236.775,177.484]
Disparity Offset	ϵ_p	0.32
Quantization Step	$\frac{1}{n}$	$\frac{1}{25}$
Resolution		$[480 \times 360]$
Interpolation		$[480 \times 360] \rightarrow [640 \times 480]$

precision (cf. (2.3)) are calculated. Although the precision quantifies the variation of the error across the image, it is not suitable to reflect the local influence of non-Gaussian errors on a pixel neighborhood. Hence, to visualize non-Gaussian errors, a 3D surface representation is generated that depicts the radial distortion error and the tangential distortion error.

To demonstrate the feasibility of the error simulation model, it is tested together with a reconstruction algorithm.

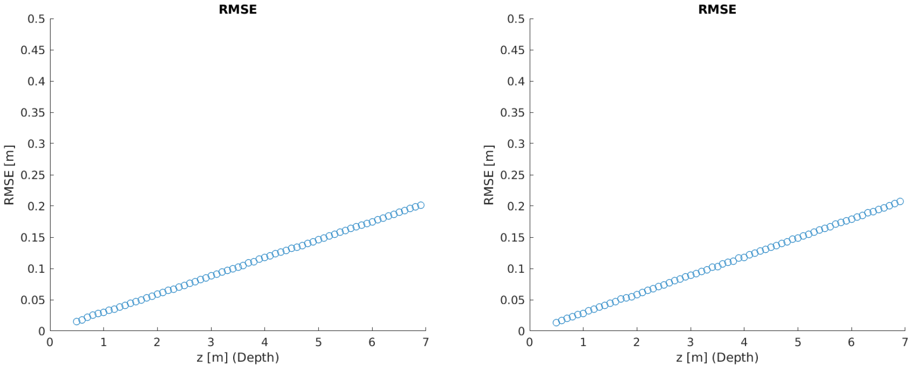
3.2.1 Simulated Error Modalities

The results reveal that the quantization error, baseline error, focal length error, disparity offset, and the interpolation error do not influence the precision of the depth sensor, but have strong influences on the RMSE (cf. Figure 3.4 and Figure 3.5).

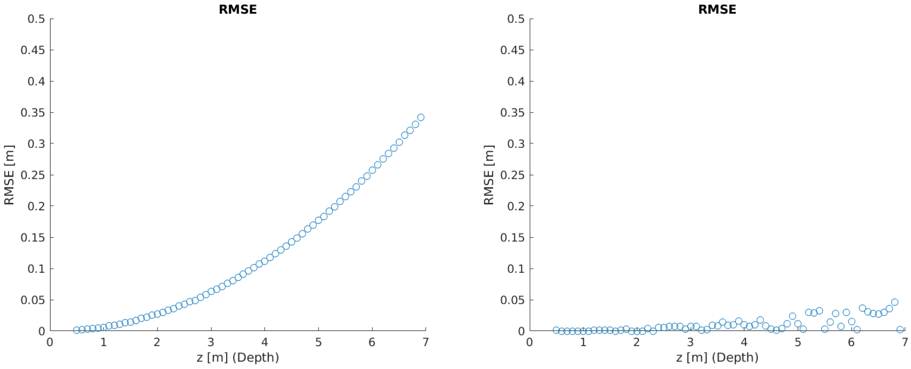
As expected, the focal length error and the baseline error result in a linearly increasing RMSE (cf. Figure 3.4a, Figure 3.4b).

The model simulates a miss-calibrated principle point by adding a constant disparity offset ϵ_d. This offset causes a non-linearly increasing RMSE of the form $\frac{1}{n+x}$ (cf. Figure 3.4c) and has a significant impact on the estimated depth values.

The disparity error reflects the finite precision of the stereo correspondences in the image space. Figure 3.4d depicts the RMSE caused by this quantization. Since the disparity quantization resolution is constant over the whole distance, and close depth values have a high disparity value, the influence is more significant for distances $> 3\,\mathrm{m}$.



(a) Focal length error. (b) Baseline error.



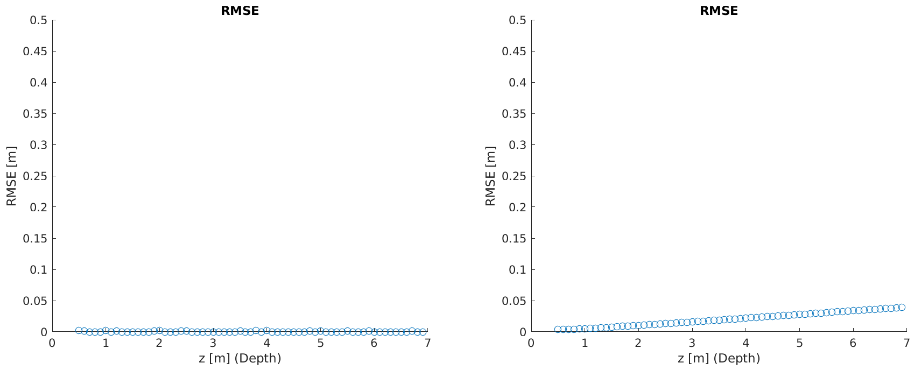
(c) Principal point error. (d) Disparity quantization error.

Figure 3.4: RMSE for focal length error, baseline error, principal point error and disparity quantization error.

Figure 3.5a shows the results for the interpolation error. As expected, the interpolation error has no influence on the RMSE for the given input data. Since the evaluation deals with synthetically generated data of fronto-parallel, perfectly flat surfaces, the interpolation value matches the ground truth value.

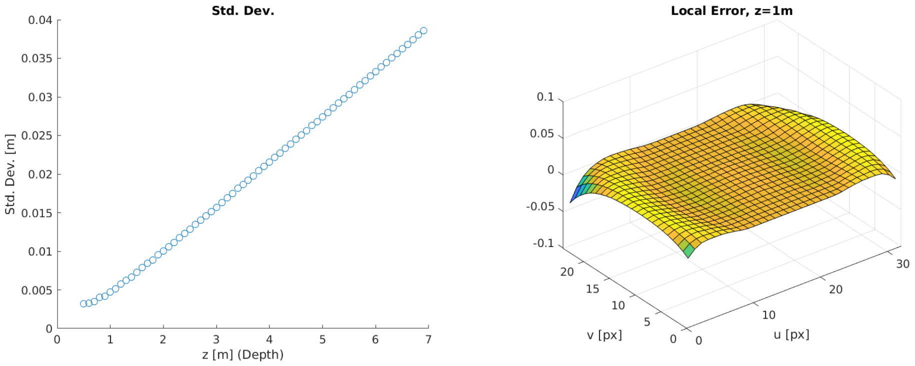
The model introduces radial distortion coefficients with the goal to simulate an alternating shift of the correspondences' coordinates, dependent on the position of a point in 3D space. Figure 3.6 shows exactly this behavior. The radial distortion error alters the pixel observation in the left and right image dependent on the observation's 3D location, which directly impacts the disparity value, and with that the estimated depth. The radial distortion error shows a linear impact on the RMSE (cf. Figure 3.5b), and on the precision (cf. Figure 3.6a), while it clearly causes a radial shaped error in the depth image (cf. Figure 3.6b).

Usually the tangential distortion is used to model a not perfectly parallel aligned lens and image sensor. The emulation model turns this around to simulate the effects of this misalignment. Figure 3.7b depicts the resulting precision, as well as the typically tilted depth map.



(a) Interpolation error.
(b) Radial distortion error.

Figure 3.5: RMSE for interpolation error and radial distortion error.



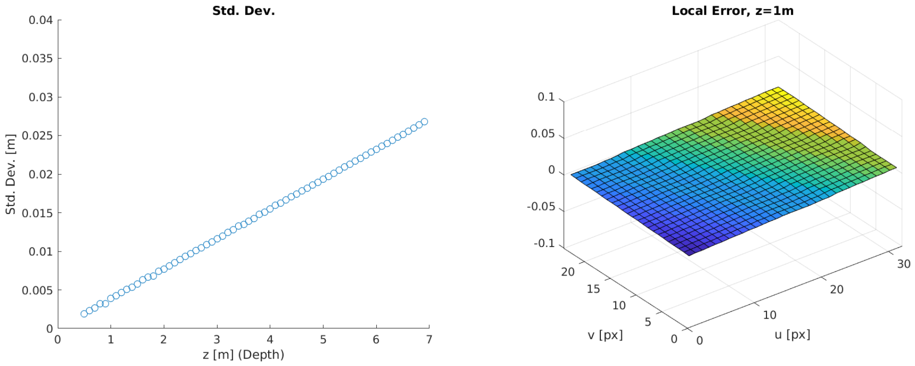
(a) Precision.
(b) Surface.

Figure 3.6: Radial distortion error.

3.2.2 Synthetic Data Augmentation

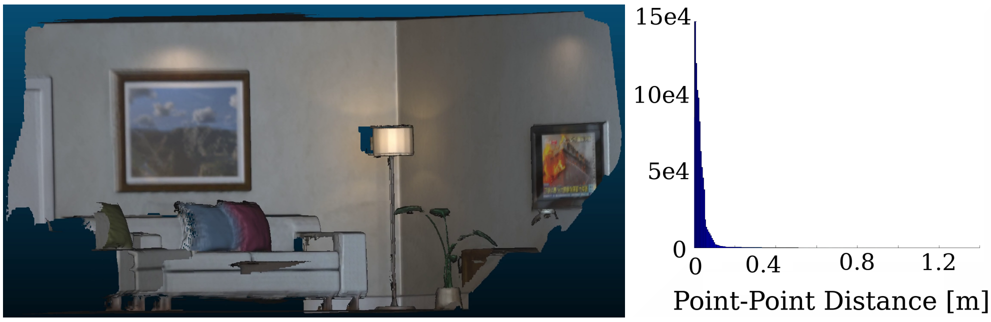
The introduced error model is meant to be applied to a synthetic dataset to simulate realistic depth errors. This section provides a feasibility study that applies the model to the synthetic ICL-NUIM dataset [71], which offers ground truth camera poses and depth maps of an indoor scene. The enriched dataset is used to investigate how depth errors impact a state-of-the-art reconstruction algorithm. The reconstruction algorithm is based on truncated signed distance functions (TSDF) [72] to acquire a detailed 3D surface representation. It relies on both, good depth data, and very accurate camera poses. The feasibility study focuses on the impact of poor depth data.

The results are qualitatively and quantitatively evaluated against a ground truth reconstruction, which is generated using the unmodified RGB-D data, the ground truth camera intrinsics, and the ground truth camera poses from the dataset. Similarly, the augmented depth images are also piped into the reconstruction algorithm. The reconstructions are analyzed and compared in order to get qualitative results. Each of the erroneous reconstructions is aligned with the ground truth reconstructions using an ICP [54] algorithm. The distance between the two aligned reconstructions is extracted



(a) Precision. (b) Surface.

Figure 3.7: Tangential distortion error.



(a) 3D reconstruction. (b) Histogram.

Figure 3.8: Reconstruction generated with perfect depth and camera pose information. The generated 3D scene and the error histogram of the cloud-cloud distance between the CAD data and the reconstructed scene represent the baseline, and the absolute limits of the used reconstruction algorithm.

with [73] and together with the corresponding error histograms gives a quantitative measure.

Figure 3.8a shows the acquired ground truth reconstruction. The walls are perfectly flat, without showing any reconstruction artifact. However, the error histogram between the CAD data of the scene and the reconstructed scene shows a slight deviation. These errors mostly originate from the used reconstruction method itself (TSDF with 1 cm grid size) and slightly from the used depth data representation which limits the depth resolution to 1 mm.

The experiments are repeated for each individual error modality and provide the reconstructions and error histograms. They reveal a direct relation between the error type (local or global) and the impact on the quality or the absolute accuracy of the reconstruction: The interpolation error, radial distortion error, and tangential distortion error are all local errors, hence they might be different for each individual pixel of the acquired depth map. Given that, they have the highest impact on the perceived surface quality, especially for flat surfaces (cf. Figure 3.9 and Figure 3.10), while they still achieve reasonable quantitative results for the absolute accuracy.

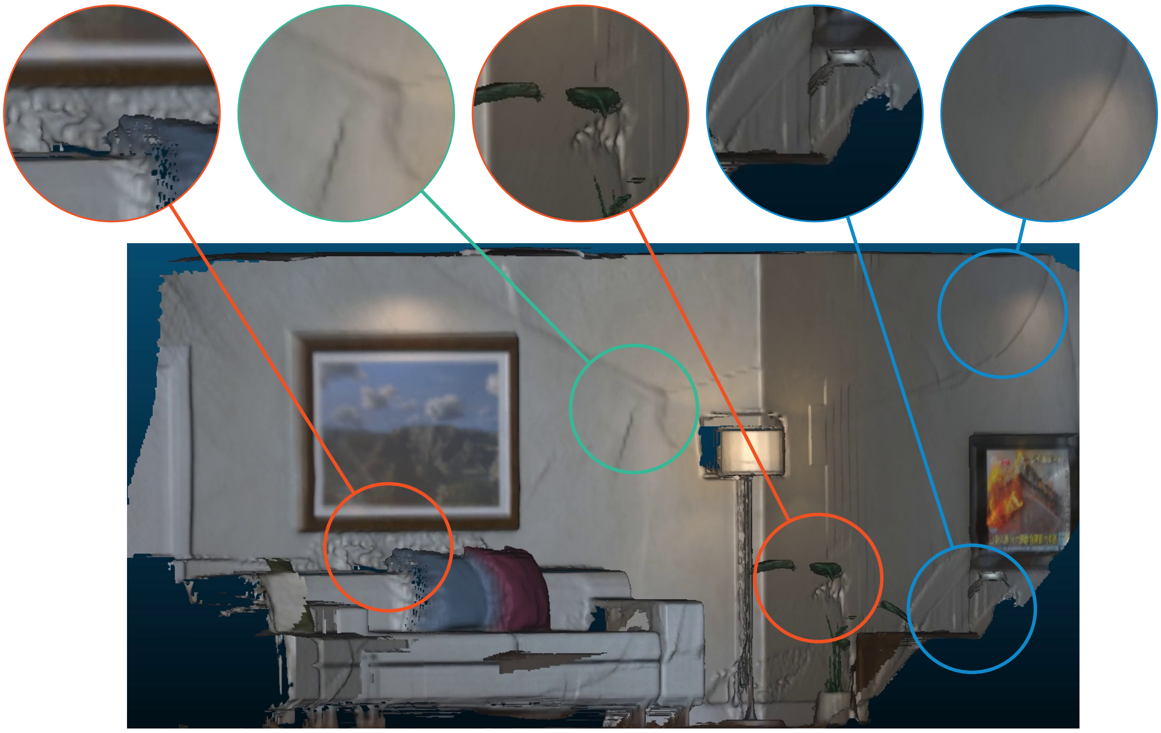


Figure 3.9: Impact of the interpolation error (orange), radial distortion error (blue), and tangential distortion error (turquoise) on the image quality.

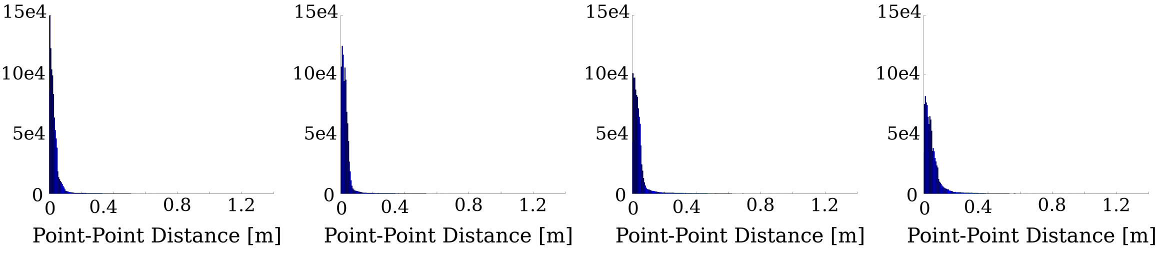
On the other hand, the focal length error, principal point error, and baseline error depend only on the depth and are independent of the 3D observation on the image plane. The experiments show that these errors tend to have less influence on the perceived quality of the reconstruction, but strongly impact the absolute accuracy of the reconstruction (cf. Figure 3.10). To conclude, these errors have less influence on neighboring pixels on a flat surface because the surface tends to have similar depth values or only gradually changing depth values. Figure 3.11 visualizes the significant impact of the baseline error on the absolute accuracy, while the surface of the reconstructions shows nearly no impact. High resolution reconstructions for the individual errors can be found in the appendix.

3.3 Discussion

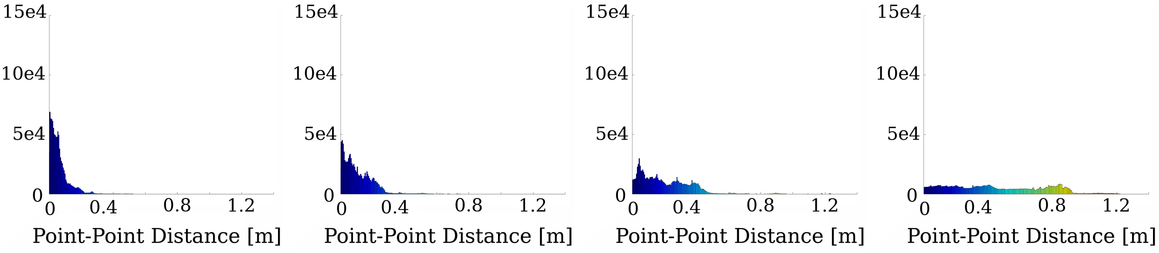
This chapter develops a generic, camera-model-oriented method to simulate systematic depth errors. It is demonstrated how the error model can be used to analyze the capability of a state-of-the-art algorithm to handle systematic linear, non-linear ($(x + n)^{-1}$), radial-shaped, and pixel-location-dependent errors (cf. Figure 3.12).

The systematic depth errors are simulated with a virtual stereo camera model that uses a slightly modified pinhole camera model and a radial and tangential distortion model.

The generic model replicates systematic depth errors of various sensor technologies



(a) Ground truth. (b) Quantization. (c) Interpolation. (d) Rad. distortion.



(e) Tan. distortion. (f) Principal point. (g) Focal length. (h) Baseline.

Figure 3.10: Reconstruction error histograms, showing the point-point distance between the reconstructed scene and the ground truth CAD data. (a) Ground truth reconstruction; (b) quantization error; (c) interpolation error; (d) radial distortion error; (e) tangential distortion error; (f) principal point error; (g) focal length error; and (h) baseline error.

and shows close-to-real-life results for the trueness and the precision of the sensor. I.e. it models non-linear and radial-shaped, local errors that are characteristic of Structured Light and Active Stereo cameras as well as linear increasing errors which are characteristic of ToF cameras.

The presented feasibility study applies the model to a synthetic dataset and investigates how imperfect depth data impacts a state-of-the-art reconstruction algorithm. It reveals that the accuracy of the reconstruction is not necessarily connected to the perceived surface quality. Hence, the results suggest that the often used pure qualitatively evaluation of reconstruction methods might be misleading and highlights the necessity of ground truth data together with realistic noise models for an impartial evaluation.

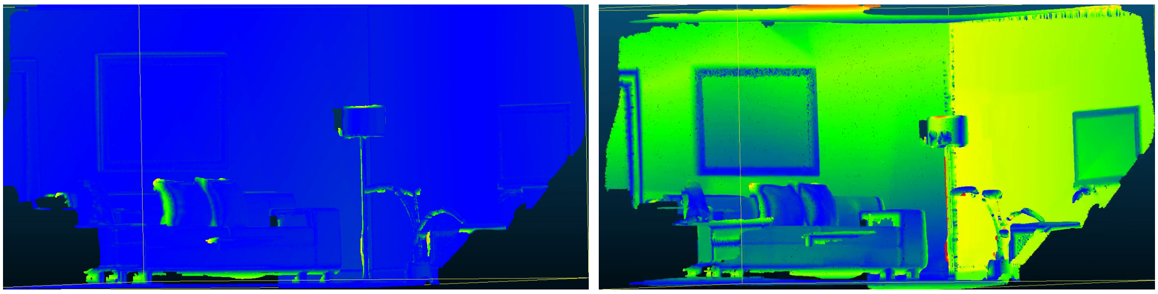
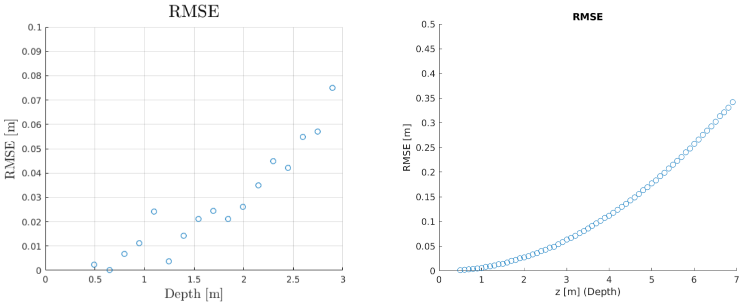
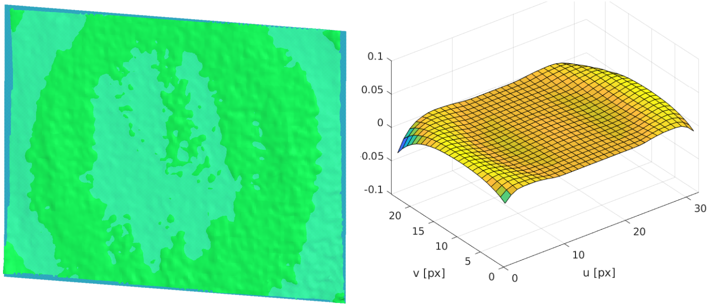


Figure 3.11: Left: ground truth reconstruction, right: reconstructed scene with applied baseline error. While the surface quality shows only slight flaws, the absolute accuracy shows a significant impact of the baseline error.



(a) R200 bias. (b) Modeled trueness using the principal point error.



(c) R200 radial local error. (d) Modeled local error using the radial distortion error.

Figure 3.12: Real data, left. Synthetically generated data, right.

The evaluation shows that synthetic datasets combined with the presented model are well suited for that purpose.

The introduced systematic error simulation model suggests that it should be possible to avoid most of the systematic errors by applying a decent calibration. For some reasons it might not be possible to directly access and correct the intrinsic parameters of a sensors, which cause the erroneous depth measurements. That motivates advanced methods, which enable a depth error compensation without modifying the intrinsic sensor parameters. The next chapter presents such an advanced method.

Chapter 4

Camera-Model-Free Compensation of Systematic Depth Errors

The sensor study presented in Chapter 2 revealed that state-of-the-art depth sensors show significant depth sensing inaccuracies. Chapter 3 shows that these errors have a negative impact on state-of-the-art algorithms. In theory a perfect calibration should correct these errors, as long as the camera's physical properties follow the calibration model. Often it is not possible to achieve perfect calibration results due to practical limitations, especially if the intrinsic camera parameters are not exposed to the user or if the required calibration models are not available. That motivates the use of alternative, camera-model-free depth compensation methods. This chapter contributes two such methods, one using a discrete compensation model, and the other a continuous compensation model. The discrete model consists of a set of compensation maps for several discrete distances, and the continuous model combines a single compensation map with an exponential function to continuously model the depth dependency.

The next section continues with a short introduction to the applied optimization and interpolation methods. After that the two error models are described in detail and evaluated on synthetic and real-life data.

4.1 Methods for Camera Model Free Depth Correction

This approach optimizes a camera-model-free depth error compensation model. It corrects depth sensing errors independent of the underlying camera model or depth sensing principals. The following two sections give a short introduction to the used optimization method and the function that is used to interpolate the compensation lattice.

4.1.1 Levenberg-Marquardt Algorithm

The Levenberg-Marquardt algorithm is an iterative, multidimensional, second-order optimization algorithm.

Iterative optimization algorithms aim to continuously update the parameter vector $\mathbf{x}^k$ to minimize a given multidimensional objective function $f(\mathbf{x}^k)$. The algorithms thereby focus on determining a search direction $\mathbf{s}^k$ and a step width $\boldsymbol{\alpha}^k$ to get the updated parameter vector

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \boldsymbol{\alpha}^k \cdot \mathbf{s}^k. \tag{4.1}$$

Different to zero- and first-order methods, second-order methods incorporate the gradient as well as the second-order derivatives (Hessian matrix, $\mathbf{H}(\mathbf{x})$) to find the correct search direction $\mathbf{s}^k$. Zero-order methods determine the search direction solely based on $f(\mathbf{x})$, while first-order methods incorporate the gradient $\nabla f(\mathbf{x})$ as well.

In [74] Treiber deduces the Levenberg-Marquardt algorithm as follows. Levenberg [75] combines the Gauss-Newton method with Gradient Descent in order to find the search direction and step width at once. Therefore, and for the sake of simplicity the combination of the search direction and the step width is further denoted as $\mathbf{s}^k$. While the Gauss-Newton method adds a quicker convergence if the initial solution $\mathbf{x}^0$ is already close to the optimal solution $\mathbf{x}^*$, the Gradient Descent method adds robustness, if $\mathbf{x}^0$ is far from the optimal solution $\mathbf{x}^*$. To understand how the Levenberg-Marquardt algorithm works, the Newton Method, and the Gauss-Newton method are summarized first.

The Newton method approximates the objective function with a second-order Taylor expansion,

$$f(\mathbf{x}) \cong T(\Delta\mathbf{x}) = f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k) \cdot \Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T \cdot \mathbf{H}(\mathbf{x}^k) \cdot \Delta\mathbf{x} \tag{4.2}$$

which holds for small $\Delta\mathbf{x}$.

Setting the first derivative to zero $\nabla T(\Delta\mathbf{x}) = 0$, and solving the linear equation system for $\mathbf{x}$ leads to an extremum of this function. If the solution represents a local minimum, the first derivative is going to be zero, and the Hessian $\mathbf{H}(\mathbf{x}^k)$ is positive definite. With that, the update rule derived from the Newton method is

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{s}^k = \mathbf{x}^k - \mathbf{H}(\mathbf{x}^k)^{-1} \cdot \nabla f(\mathbf{x}^k) \tag{4.3}$$

The Gauss-Newton method is a specialization of the Newton method and deals with objective functions $f(x)$, which are composed of a sum of squared values,

$$f(\mathbf{x}) = \sum_{i=1}^{m} r_i(\mathbf{x})^2 \quad \text{with} \quad \mathbf{x} = [x_1 \quad \dots \quad x_n]^T \tag{4.4}$$

where r_i are residuals (deviations from a regression function).

If the objective function is composed as given in (4.4), $\nabla f(\mathbf{x})$ simplifies to

$$\nabla f(\mathbf{x}) = 2 \cdot \mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{r} \quad \text{with} \quad \mathbf{J_r}(\mathbf{x}) = \begin{bmatrix} \frac{\partial r_1}{\partial x_1} & \dots & \frac{\partial r_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial x_1} & \dots & \frac{\partial r_m}{\partial x_n} \end{bmatrix}. \tag{4.5}$$

Moreover, $\mathbf{H}(\mathbf{x})$ can be derived from $\nabla f(\mathbf{x})$ with

$$\begin{aligned} H_{ij} = \frac{\nabla f_j(\mathbf{x})}{\partial x_l} &= 2\sum_{i=1}^{m} \left(\frac{\partial r_i(\mathbf{x})}{\partial x_l} \cdot J_{ij}(\mathbf{x}) + r_i(\mathbf{x}) \cdot \frac{\partial J_{ij}(\mathbf{x})}{\partial x_l} \right) \\ &= 2\sum_{i=1}^{m} \left(J_{il}(\mathbf{x}) \cdot J_{ij}(\mathbf{x}) + r_i(\mathbf{x}) \cdot \frac{\partial^2 r_i(\mathbf{x})}{\partial x_j \cdot \partial x_l} \right) \end{aligned}. \tag{4.6}$$

With two assumptions, (4.6) can be further simplified. First, if the current solution $\mathbf{x}^k$ is already near the optimum, the residuals $r_i(\mathbf{x})$ will be very small and the second order derivatives will not have too much influence on $\mathbf{H}$. Second, near the current solution $\mathbf{x}^k$, $f(\mathbf{x}^k)$ can be approximated with a linear function such that the second-order terms are small. If one of these two assumption holds, the influence of the quadratic part can be ignored, and the Hessian can be approximated with

$$\mathbf{H}(\mathbf{x}) \approx 2 \cdot \mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{J_r}(\mathbf{x}) \tag{4.7}$$

To conclude, considering an objective function of the form (4.4), and given that the initial solution is already near the optimal solution, or if $f(\mathbf{x})$ can be approximated with a linear function respectively, Newton's update rule simplifies to

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{s}^k = \mathbf{x}^k - \left(\mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{J_r}(\mathbf{x})\right)^{-1} \cdot \mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{r} \tag{4.8}$$

However, most likely these two assumptions might not hold in practice, especially if $\mathbf{x}^0$ is far from $\mathbf{x}^*$. Levenberg [75] approached that by combining the Gauss-Newton method with Gradient Descent. Gradient Descent uses the negative gradient

$$\mathbf{s}^k = -\nabla f(\mathbf{x}^k) = -2 \cdot \mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{r} \tag{4.9}$$

to determine the search direction.

The influence of Gradient Descent can be controlled with an additional factor λ, such that

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{s}^k = \mathbf{x}^k - \left(\mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{J_r}(\mathbf{x}) + \lambda \cdot \mathbf{I}\right)^{-1} \cdot \mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{r}. \tag{4.10}$$

λ has a big impact on the convergence behavior. Situations with a high curvature and a small gradient lead to very small steps and a poor convergence rate.

Hence, Marquardt [76] extended the algorithm by scaling the gradient by the curvature, which is estimated from the approximated Hessian. This extension results in the well-known Levenberg-Marquardt algorithm, and the update rule

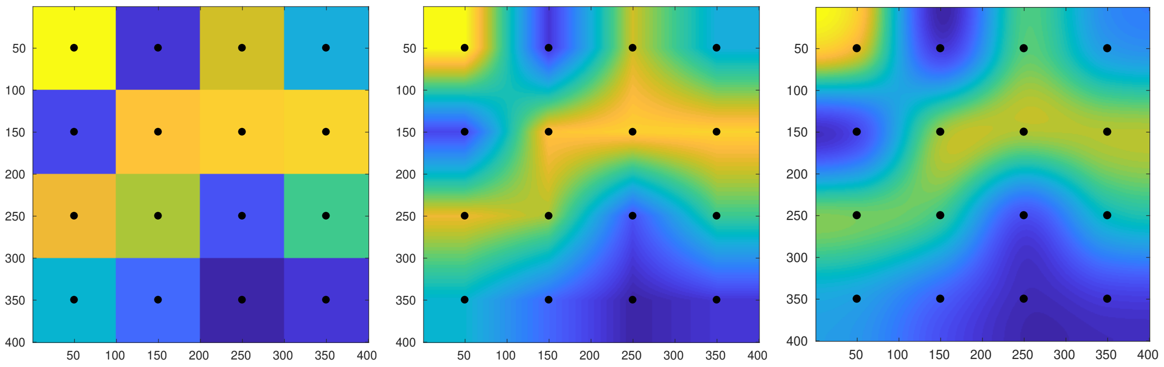
$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{x}^k + \mathbf{s}^k \\ &= \mathbf{x}^k - \left(\mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{J_r}(\mathbf{x}) + \lambda \cdot diag\left(\mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{J_r}(\mathbf{x})\right)\right)^{-1} \cdot \mathbf{J_r}(\mathbf{x})^T \cdot \mathbf{r} \end{aligned} \tag{4.11}$$

Summarized, the Levenberg-Marquardt algorithm is a robust, well-known optimization method that combines the Gauss-Newton method with Gradient Descent and scales the gradient according to the curvature to adjust the step size.

For more detailed information regarding the Levenberg-Marquardt algorithm, especially for more information on how to select the correct λ, it is referred to [74]–[77].

4.1.2 Bicubic Interpolation

There are several ways to interpolate a low-resolution lattice to achieve high resolution images. Three of the most commonly used methods are the Nearest-Neighbor Interpolation 4.1a, the Bilinear Interpolation 4.1b and the Bicubic interpolation 4.1c. Figure 4.1 depicts the different interpolation results of the three methods.



(a) Nearest Neighbor. (b) Bilinear. (c) Bicubic.

Figure 4.1: Three different interpolation methods: The Nearest Neighbor interpolation is the simplest method and takes the value from the closest point without any interpolation between the points. The bilinear interpolation considers the value of the four corner points and interpolates between them. The bicubic interpolation considers the four corner points and their derivatives.

The nearest neighbor interpolation sets the pixel that has to be interpolated to the closest interpolation coefficient. The method is easy to implement, offers the least computational costs but shows strong interpolation artifacts. The linear interpolation takes four corner points into account and interpolates linearly between them. Compared to nearest neighbor, the linear interpolation offers smooth transitions but results in non-continuous derivatives over the boundaries. The bicubic interpolation interpolates the current area based on the four neighboring points and their derivatives, extracted from the point neighborhood. The bicubic interpolation offers continuous derivatives over the image boundaries and is therefore well suited to be used in an optimization problem.

Together with Figure 4.2, the bicubic interpolation and the corresponding Jacobians are derived as follows: With the four corner points and the derivatives $\frac{\partial f}{\partial x} = f_x$, $\frac{\partial f}{\partial y} = f_y$,

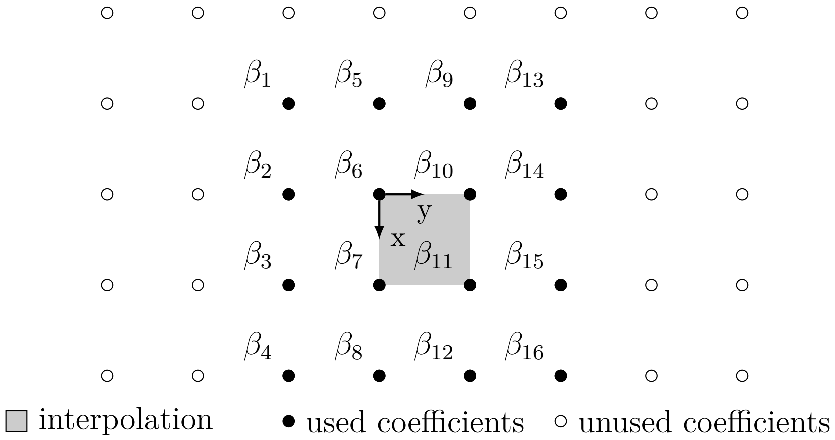


Figure 4.2: Bicubic interpolation scheme.

$\frac{\partial^2 f}{\partial x \partial y} = f_{x,y}$, the interpolation function $f(.)$ is constructed with

$$f(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{i,j} x^i y^j \quad \forall x,y \in [0,1] \tag{4.12}$$

under the conditions

$$\begin{array}{cc}
f(0,0) = \beta_6 & f(1,0) = \beta_7 \\
f(0,1) = \beta_{10} & f(1,1) = \beta_{11} \\
f_x(0,0) = \frac{\beta_7-\beta_5}{2} & f_x(1,0) = \frac{\beta_8-\beta_6}{2} \\
f_x(0,1) = \frac{\beta_{11}-\beta_9}{2} & f_x(1,1) = \frac{\beta_{12}-\beta_{10}}{2} \\
f_y(0,0) = \frac{\beta_{10}-\beta_2}{2} & f_y(1,0) = \frac{\beta_{11}-\beta_3}{2} \\
f_y(0,1) = \frac{\beta_{14}-\beta_6}{2} & f_y(1,1) = \frac{\beta_{15}-\beta_7}{2} \\
f_{xy}(0,0) = \frac{\beta_{11}-\beta_9-\beta_3+\beta_1}{4} & f_{xy}(1,0) = \frac{\beta_{12}-\beta_{10}-\beta_4+\beta_2}{4} \\
f_{xy}(0,1) = \frac{\beta_{15}-\beta_{13}-\beta_7+\beta_5}{4} & f_{xy}(1,1) = \frac{\beta_{16}-\beta_{14}-\beta_8+\beta_6}{4},
\end{array} \tag{4.13}$$

where β_i are the interpolation coefficients.

Solving the equation system and rewriting it in matrix form results in

$$\begin{aligned}
c(x,y,\boldsymbol{\beta}) =& \mathbf{g}^T(x,y) \cdot \mathbf{A} \cdot \boldsymbol{\beta} \\
\mathbf{g}^T(x,y) =& \begin{bmatrix} 1 & x & x^2 & x^3 & y & xy & x^2y & x^3y & y^2 & xy^2 & x^2y^2 & x^3y^2 & y^3 & xy^3 & x^2y^3 & x^3y^3 \end{bmatrix} \\
\boldsymbol{\beta}^T =& \begin{bmatrix} \beta_1 & \dots & \beta_{16} \end{bmatrix} \\
\mathbf{A} \in& \mathbb{R}^{16\times 16}
\end{aligned} \tag{4.14}$$

where $\mathbf{A}$ is constant (cf. Appendix), $\boldsymbol{\beta}$ is a vector containing the 16 necessary interpolation coefficients, and x,y are the coordinates of the point that has to be interpolated. Further simplifying (4.14) results in

$$c(x,y,\boldsymbol{\beta}) = \boldsymbol{\gamma}^T(x,y) \cdot \boldsymbol{\beta}, \qquad \boldsymbol{\gamma}^T \in \mathbb{R}^{16}. \tag{4.15}$$

With (4.15), it is trivial to calculate the corresponding Jacobian $\mathbf{J}_c$ to

$$\begin{aligned}
\mathbf{J}_c(x,y,\boldsymbol{\beta}) &= \left[\frac{\partial c}{\partial x}, \quad \frac{\partial c}{\partial y}, \quad \frac{\partial c}{\partial \boldsymbol{\beta}}\right] \\
&= \left[\frac{\partial \boldsymbol{g}^T}{\partial x} \cdot \mathbf{A} \cdot \boldsymbol{\beta}, \quad \frac{\partial \boldsymbol{g}^T}{\partial y} \cdot \mathbf{A} \cdot \boldsymbol{\beta}, \quad \boldsymbol{\gamma}^T(x,y)\right] \\
&= \left[\frac{\partial \boldsymbol{\gamma}^T}{\partial x} \cdot \boldsymbol{\beta}, \quad \frac{\partial \boldsymbol{\gamma}^T}{\partial y} \cdot \boldsymbol{\beta}, \quad \boldsymbol{\gamma}^T(x,y)\right] \qquad \mathbf{J}_c \in \mathbb{R}^{18}
\end{aligned} \tag{4.16}$$

With that the bicubic interpolation is defined and ready to be used together with a Levenberg-Marquardt optimizer.

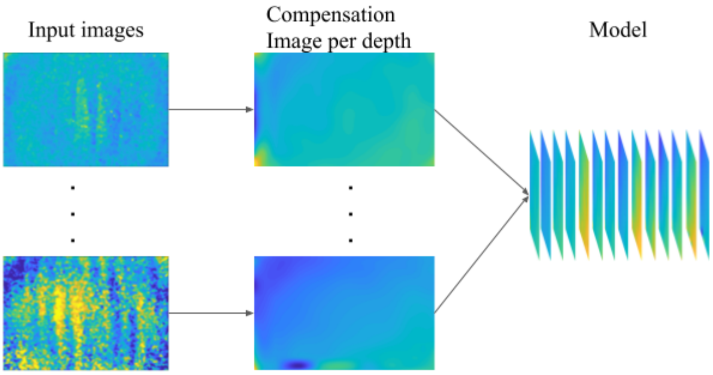


Figure 4.3: Depth compensation with depth-discrete compensation models.

4.2 Discrete and Continuous Offset Compensation Models

With (3.2), Chapter 3 defines a comprehensive model for the systematic depth sensing error e. It depends on the depth z as well as the pixel coordinates u,v. This section presents two approaches to compensate such depth-dependent and pixel-dependent errors without taking a classic camera model into account.

First, the error is modeled with a depth-discrete set of independent functions

$$e(u,v,z) \rightarrow C = \{f_{z_0}(u,v),...,f_{z_n}(u,v)\}, \tag{4.17}$$

where every f_{z_i} represents a bicubic interpolation of the form $c(u,v,\boldsymbol{\beta_i})$, at a given distance z_i. To conclude, the first method determines the parameter vectors $\boldsymbol{\beta}_i$ of several bicubic interpolations to compensate the depth image at specific distances (cf. Figure 4.3).

Second, the connection between the independent functions is modeled to come up with a depth-continuous model of the form

$$e(u,v,z) \rightarrow f(z) \cdot c(u,v,\boldsymbol{\beta}), \tag{4.18}$$

where $c(u,v,\boldsymbol{\beta})$ is a single bicubic interpolation to model the local variation of the systematic error and $f(z)$ is an exponential function to continuously model the depth dependency of the pixel-wise errors (cf. Figure 4.4).

4.2.1 Depth-Discrete Model

This section in detail describes a depth-discrete method to compensate depth errors with a set of correction images of the form

$$C = \{f_{z_0}(u,v),...,f_{z_n}(u,v)\}. \tag{4.19}$$

The model is optimized for an input set of erroneous depth images and ground truth depth maps. If the data includes only systematic error components, a single image of

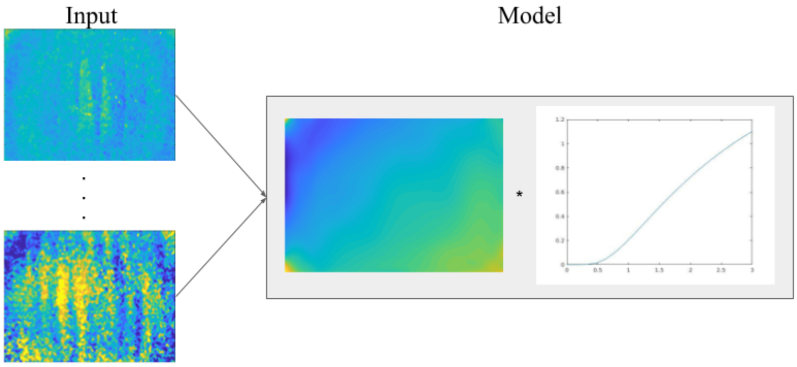


Figure 4.4: Depth compensation with one compensation map and a depth-continuous exponential function.

a perfectly flat surface for each distance is sufficient to determine the compensation parameters. However, in practice the input data also includes random error components which are normally distributed. Hence, several depth images are recorded for every distance, and the whole set $S_i = \{D_{ij}\}$ is used for the optimization. The optimization will result in an optimal parameterization as long as the errors are normal distributed. The necessary sets are usually generated by placing the camera at known distances z_{GT} as parallel as possible to a planar target that covers the full camera image (cf. Figure 4.5).

In practice, the sensor might be imperfectly placed relative to the flat surface. Hence, the orientation of the camera relative to the surface $\mathbf{R_i} \in SO(3)$ is initially optimized and added to the residual. With that, it is possible to compensate for the imperfect alignment of the camera.

The input tuples $U_i = (S_i,z_{GT,i})$ are used to construct an optimization problem with corresponding parameter set $\boldsymbol{\theta_i} = \{\boldsymbol{\beta_i}, \mathbf{R_i}\}$ for every distance i.

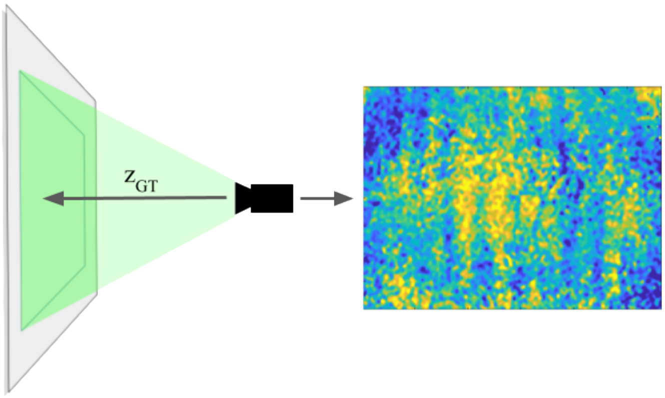


Figure 4.5: Input images recording setup.

The residual is defined as

$$\begin{aligned} r_{D,ijm} &= z_{GT,i} - z_{ijm,c} - c_i\left(u_{ijm}, v_{ijm}, \boldsymbol{\beta_i}\right) \\ \text{s.t.} \qquad \mathbf{x_c} &= \mathbf{R_i} \circ \mathbf{x_{ijm}} = \begin{bmatrix} u_{ijm,c} & v_{ijm,c} & z_{ijm,c} \end{bmatrix}^T \\ \mathbf{x_{ijm}} &= \begin{bmatrix} u_{ijm} & v_{ijm} & D_{ij}(u_{ijm}, v_{ijm}) \end{bmatrix}^T , \end{aligned} \tag{4.20}$$

with i as the index of the current input tuple U_i, j as the index of the depth map of the input image set S_i, and m as the index of the image coordinates for the current sampled depth map D_{ij}. $\mathbf{R_i}$ defines the rotation in the image space and with that avoids the use of the camera intrinsic parameters.

A qualitative analysis revealed that the depth data includes gross outliers that could negatively influence the optimization result. Hence, to increase the robustness of the optimization and the quality of the result, a Tukey loss function [78], [79] of the form

$$\rho(r_i^2) = \rho(s) = \begin{cases} \frac{a^2}{6} \cdot \left(1 - \left(1 - \frac{s}{a^2}\right)^3\right) & \text{for} \quad s \leq a^2 \\ \frac{a^2}{6} & \text{for} \quad s > a^2 \end{cases} \tag{4.21}$$

is introduced. It sets the Jacobians of gross outliers to zero and eliminates their impact on the optimization.

With that n independent optimization problems of the form

$$\min_{\theta_i} \sum_{j,m} \rho(\|r_{D,ijm}\|^2) \tag{4.22}$$

are constructed.

Jacobians

The optimization is two-fold. First, it optimizes only the orientation of the camera relative to the plane. It utilizes auto-differentiation techniques to derive the Jacobians for the $SO(3)$ operation.

$$\mathbf{J} = \left[\frac{\partial r_{C,ijm}}{\partial \boldsymbol{\theta}}\right] = \left[\frac{\partial r_{C,ijm}}{\partial z_{ijm,c}} \cdot \frac{\partial z_{ijm,c}}{\partial \mathbf{R_i}}\right] \tag{4.23}$$

Second, the camera orientation is kept constant and only the bicubic interpolations are optimized.

We exploit automatic differentiation as well as analytic differentiation to determine the Jacobians for the second step optimization problem.

Since the lookup coordinates (u,v) can be held constant during the optimization, the Jacobians of the compensation functions (4.16) simplify to.

$$\begin{aligned} \mathbf{J}_c(x,y,\boldsymbol{\beta}) &= \left[\frac{\partial c}{\partial x}, \quad \frac{\partial c}{\partial y}, \quad \frac{\partial c}{\partial \boldsymbol{\beta}}\right] \\ &= \left[0, \quad 0, \quad \boldsymbol{\gamma}^T(x,y)\right] \qquad \mathbf{J}_c \in \mathbb{R}^{18} \end{aligned} \tag{4.24}$$

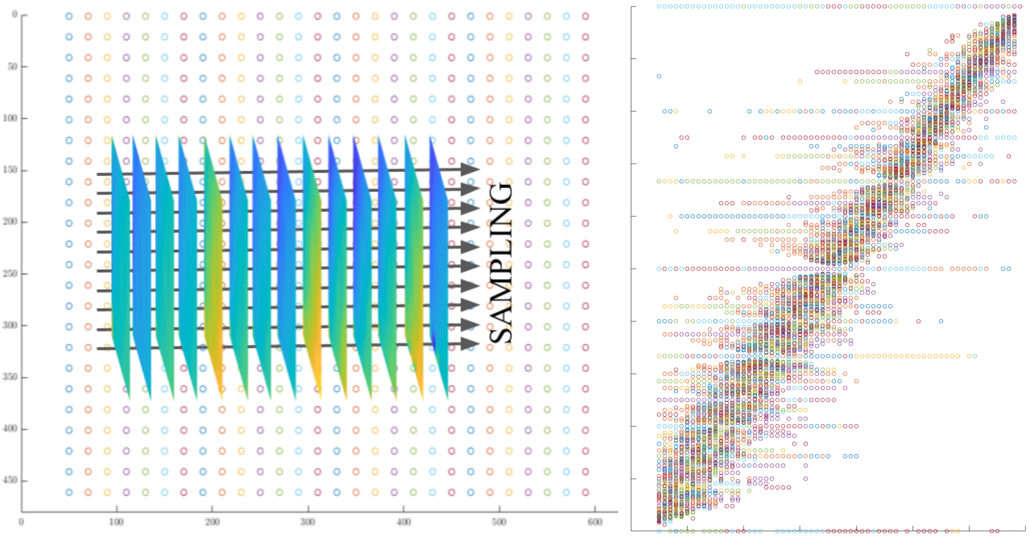


Figure 4.6: Sampling of several separate model and the resulting connection between the offset compensation models.

4.2.2 Depth-Continuous Model

The depth-discrete model introduced beforehand parameterizes one bicubic compensation function for each measured distance, which results in a set of n independent bicubic compensation functions. The data presented in Chapter 3 indicates that the shape of the systematic pixel-dependent error is constant but has a varying magnitude over the distances. To investigate if the magnitude variation can be modeled with a model of the form $f(z)$, the connection between the compensation functions has to be analyzed. Therefore, 24×28 (672) uniformly distributed pixel coordinates are analyzed and the magnitude of the corresponding correction values δ_i is extracted for all n distances, ending up with 672 sets $(C_1 \dots C_{672})$ of n correction values. To compare the shape of the resulting curves, the correction values are normalized with

$$
\begin{aligned}
C_1 &= \{\delta_{1,1}, \dots, \delta_{1,n}\} \cdot \arg\max_i(\delta_{1,i})^{-1} \\
&\vdots \\
C_{672} &= \{\delta_{672,1}, \dots, \delta_{672,n}\} \cdot \arg\max_i(\delta_{672,i})^{-1}
\end{aligned}
\tag{4.25}
$$

using the maximal correction value of the set.

Figure 4.6 depicts the sampling procedure and the resulting plot. It indicates that the change in the error magnitude can be modeled with an exponential function of the form

$$
f(z) = \exp(a_0 - a_1 \cdot z^{-1}). \tag{4.26}
$$

Figure 4.7 shows a plot of (4.26), where $[a_0, a_1] = [0.5, 1.5]$.

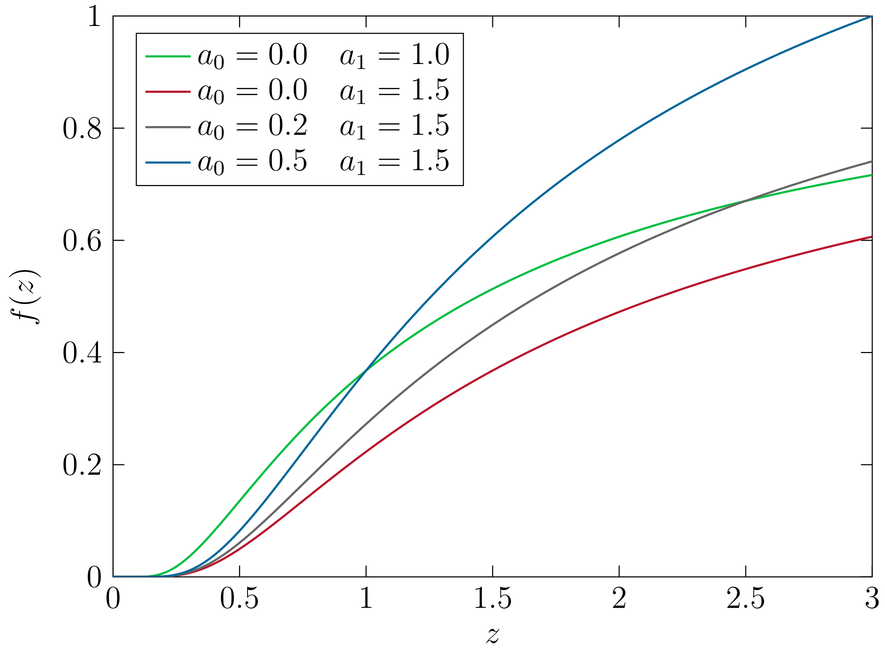


Figure 4.7: Plot of $f(z) = \exp(a_0 - a_1 \cdot z^{-1})$.

With that the continuous offset compensation model is defined as

$$\begin{aligned} f(u,v,z) &= f(z) \cdot c(u,v,\boldsymbol{\beta}), \\ &= \exp(a_0 - a_1 \cdot z^{-1}) \cdot c(u,v,\boldsymbol{\beta}) \end{aligned} \tag{4.27}$$

Different to the depth-discrete model, the depth-continuous model parameterizes a single bicubic interpolation function together with an exponential function that continuously models the z-dependency of the offset's magnitude. To conclude, the method constructs an optimization problem for all input tuples $U_i = (S_i, z_{GT,i})$ and the parameter set $\boldsymbol{\theta} = \{a_0, a_1, \boldsymbol{\beta}, \{\mathbf{R_i}\}\}$. That results in the new residual

$$\begin{aligned} r_{C,ijm} &= z_{GT,i} - z_{ijm,c} - \delta(u_{ijm}, v_{ijm}, \boldsymbol{\beta}, a_0, a_1) \\ &= z_{GT,i} - z_{ijm,c} - \exp(a_0 - a_1 \cdot z_{ijm,c}^{-1}) \cdot c\,(u_{ijm}, v_{ijm}, \boldsymbol{\beta}) \\ \text{s.t.} \qquad \mathbf{x_c} &= \mathbf{R_i} \circ \mathbf{x_{ijm}} = \begin{bmatrix} u_{ijm,c} & v_{ijm,c} & z_{ijm,c} \end{bmatrix}^T \\ \mathbf{x_{ijm}} &= \begin{bmatrix} u_{ijm} & v_{ijm} & D_{ij}(u_{ijm}, v_{ijm}) \end{bmatrix}^T, \end{aligned} \tag{4.28}$$

with i as the index of the input tuple U_i, j as the index of the depth map of the input image set S_i, and m as the index of the image coordinates for the current sampled depth map D_{ij}.

A Tukey loss function $\rho(.)$ is used to reject gross outliers. With that the optimization problem has the form

$$\min_{\theta} \sum_{j,m,i} \rho(\|r_{C,ijm}\|^2). \tag{4.29}$$

Jacobians

As for the depth-discrete error model described in Section 4.2.1, the optimization is two-fold. First, the method optimizes the orientation of the camera relative to the plane using the auto-differentiation capabilities of the used solver.

Second, the camera orientation is kept constant and only the bicubic interpolation is optimized together with the exponential function that models the z-dependency. The necessary Jacobian for the optimization is

$$\begin{aligned}\mathbf{J} = \left[\frac{\partial \mathbf{r_{C,ijm}}}{\partial \boldsymbol{\theta}}\right] &= \left[\frac{\partial \delta}{\partial a_0}, \quad \frac{\partial \delta}{\partial a_1}, \quad \frac{\partial \delta}{\partial \boldsymbol{\beta}}\right] \\ &= \left[\delta(.), \quad -z_{ijm,c}^{-1} \cdot \delta(.), \quad \boldsymbol{\gamma}^T(x,y) \cdot \exp(a_0 - a_1 \cdot z_{ijm,c}^{-1})\right].\end{aligned} \tag{4.30}$$

4.3 Evaluation

The evaluation utilizes the simulation model developed in Chapter 3 to generate a synthetic dataset. The synthetic dataset is used together with a real-life dataset to test the two newly introduced error compensation models. This section analyses the impact of the compensation models on the RMSE (trueness) and the standard deviation (precision) of the results. Since the precision might be dominated by the random errors, the results are qualitatively evaluated by comparing the depth maps before and after the correction was applied.

In addition, the depth-continuous model is used together with the augmented synthetic dataset to evaluate the impact of the correction on the results of a state-of-the-art reconstruction algorithm.

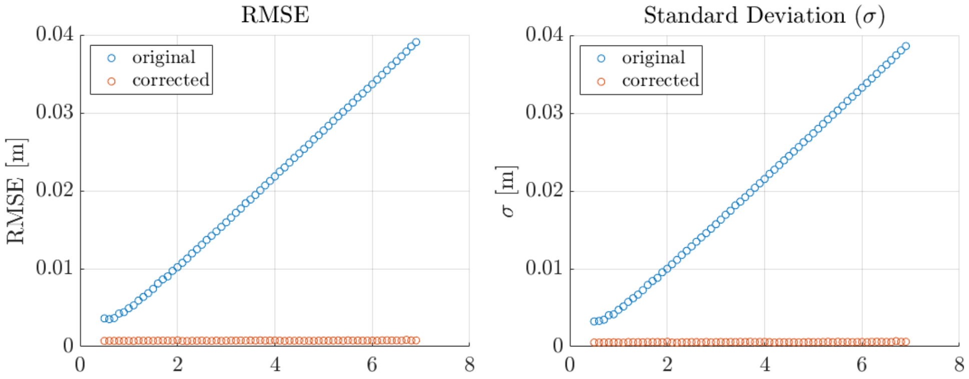
Finally, the performance of two different implementations is analyzed in terms of computational speed. The first implementation uses only the automatic differentiation capability of the solver, while the second makes use of handcrafted analytic Jacobians.

4.3.1 Datasets

The synthetic dataset consists of several synthetic depth images which are augmented with the error simulation model introduced in Chapter 3. The dataset contains depth maps for distances from 0.5m to 6.9m at an interval of 0.1m and does not include random noise components.

The real-life dataset is collected using an Active Stereo camera (Intel RS200) which is placed as parallel as possible in front of a flat target. The distance of the camera to the target was gradually increased from 0.4m to 2.1m using varying intervals. For every distance 40 consecutive depth frames are recorded and the ground truth distance is manually measured to construct the input tuples for the optimization.

To test the impact of the depth-continuous compensation method on a TSDF reconstruction algorithm, a third dataset is generated that contains a sequence from the ICL-NUIM living room dataset [71]. The publicly available dataset is augmented with the simulation model introduced in Chapter 3. The same error model is used that is also utilized to construct the first dataset.



(a) Trueness.

(b) Precision.

Figure 4.8: Trueness and precision for the synthetic dataset before (blue) and after (orange) applying the depth-discrete error compensation model.

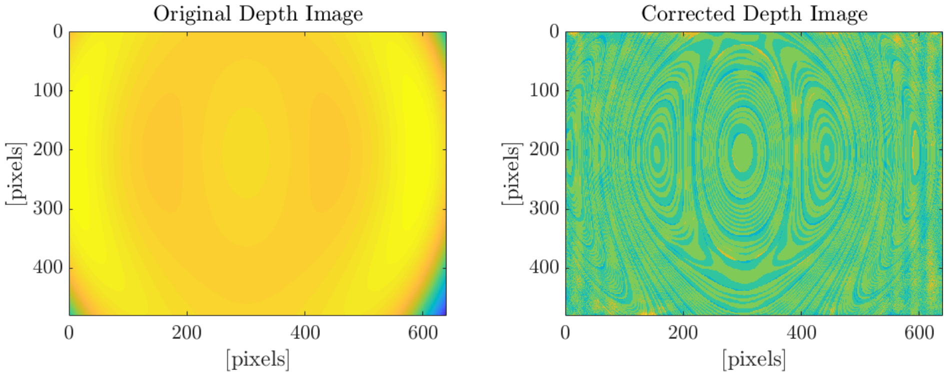


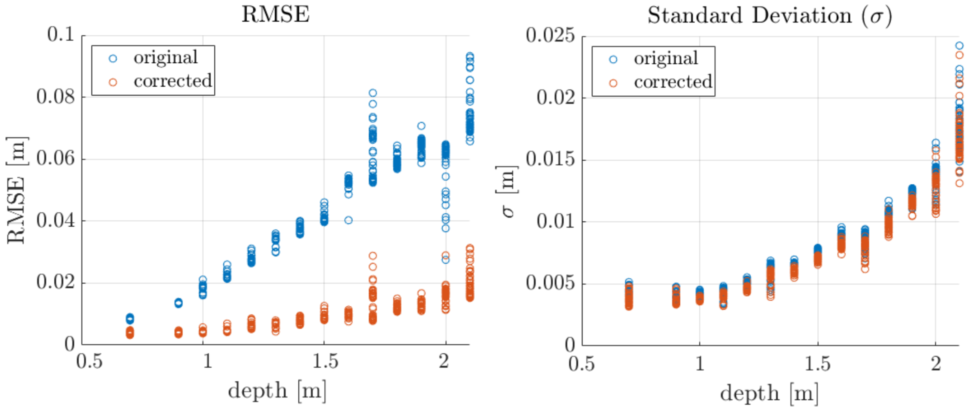
Figure 4.9: Depth images for $d = 6.9$m of the synthetic dataset before and after applying the discrete compensation model. Blue represents small errors yellow represents high errors.

4.3.2 Depth-Discrete Error Compensation Model

First, the depth-discrete error compensation model is evaluated on the synthetic dataset. Our discrete model optimizes an individual compensation model for every depth distance, ending up with an optimal compensation for every input tuple. Since the synthetic dataset contains only systematic noise, the model is able to unfold its full capabilities and drastically reduces the introduced errors (cf. Figure 4.8).

Figure 4.9 enables a qualitative analysis of the depth images before and after the correction. It shows that the corrected images contain only minor quantization artifacts with a magnitude around $1e^{-3}$m.

Second, the experiments are repeated on the real-life dataset. The experiments revealed significant improvements of the trueness, but show less impact on the precision (cf. Figure 4.10). As expected, the precision seems to be dominated by the random error components which mask the compensation of local errors. Hence, for the experiments with real-life data, the qualitative results are extended by an image of the applied correction (cf. Figure 4.11) and a mesh of the gathered surface measurements. To



(a) Trueness. (b) Precision.

Figure 4.10: Trueness and precision for the real-life dataset before (blue) and after (orange) applying the depth-discrete error compensation model.

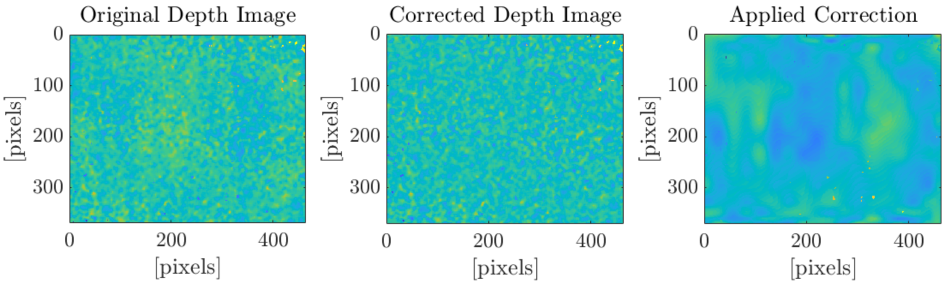


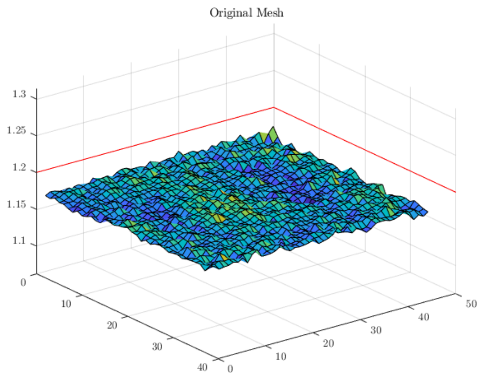
Figure 4.11: Depth at $d = 1.2$m before (left) and after (center) applying the discrete compensation model (right). Blue represents small errors, yellow represent high errors.

better illustrate the magnitude of the errors, the mesh and the depth images are colored according to the minimum and maximum value. Blue represents small values yellow represents high values.

Figure 4.12 and Figure 4.11 compare the depth images and meshes, respectively, of the original and the corrected images, showing measurement results of a flat target observed from 1.2m. The red line indicates the ground truth value. The left image shows the uncompensated data, including a measurement trueness for the whole image, systematic surface irregularities especially in the image center, and random noise across the whole image. The systematic noise is reflected in a slightly yellow and green region in the center of the original image (left), which is compensated in the center of the right image by the applied depth-discrete compensation model.

4.3.3 Depth-Continuous Error Compensation Model

Different to the discrete model, the continuous model consists of a single bicubic interpolation function and an exponential function that implicitly models the z dependency of the error. Hence, it is possible to extract an offset compensation value even for depth

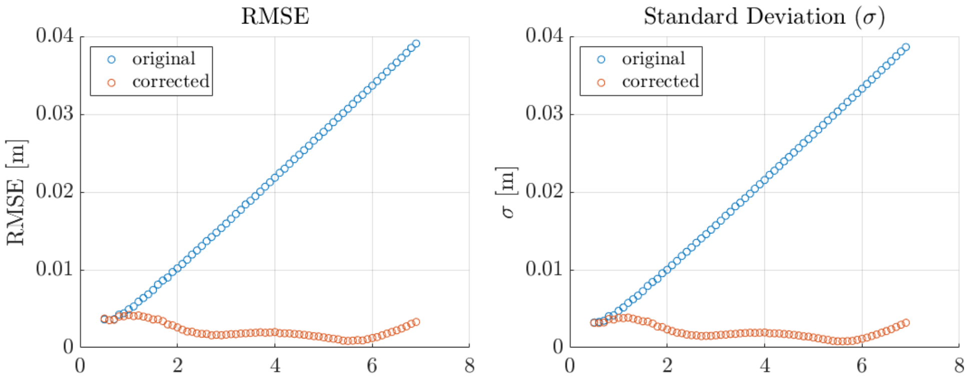


(a) Original. (b) Compensated.

Figure 4.12: Surfaces before (left) and after (right) applying the depth-discrete error compensation model.

values that are not represented in the input dataset, without the need to interpolate between two sets.

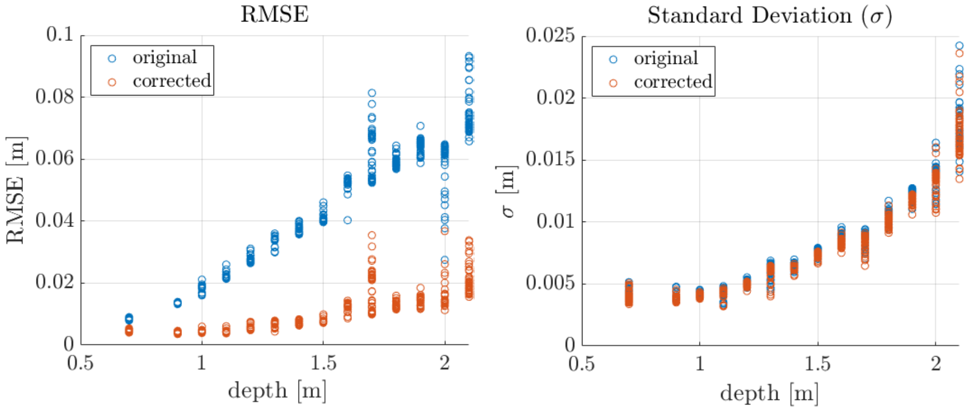
Similar to the evaluation of the depth-discrete model, the depth-continuous model is evaluated with the synthetically augmented dataset first. Figure 4.13 shows that the depth-continuous model improves the trueness and the precision, but shows worse results than the depth-discrete model, especially for shorter distances. These results are in line with the introduced model and the form of the exponential function to implicitly model the z dependency. It results in small compensation values for distances close to zero and models a stronger compensation for larger distances (cf. Figure 4.7), hence it has less influence on near depth measurements.



(a) Trueness. (b) Precision.

Figure 4.13: Trueness and precision for the synthetic dataset before (blue) and after (orange) applying the depth-continuous error compensation model.

Next, the compensation model is applied to the real-life dataset, which contains systematic depth errors and random errors. The experiments revealed that the depth-continuous model and the depth-discrete model achieve comparable results for correcting the trueness and the precision. Similar to the results for the discrete model, the continuous model has less influence on the precision since it is dominated by the random



(a) Trueness.

(b) Precision.

Figure 4.14: Trueness and precision for the real-life dataset before (blue) and after (orange) applying the depth-continuous error model.

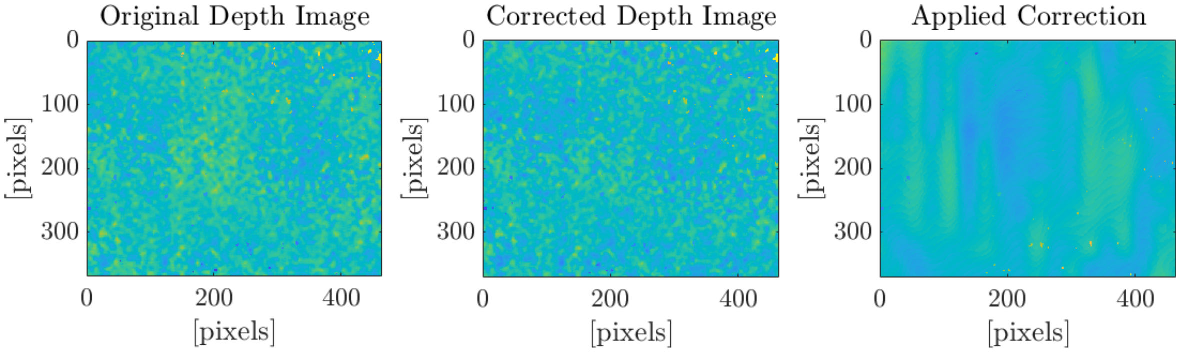


Figure 4.15: Depth at $d = 1.2$m before (left) and after (center) applying the continuous compensation model (right).

error (cf. Figure 4.14).

Figure 4.15 depicts the qualitative results of the compensation model. It shows the depth image before and after the compensation, as well as the applied compensation and shows similar improvements as the depth-discrete error model. The improvements are especially visible in the center of the image: The original image shows a yellow region that indicates higher depth errors, while the bright yellow region is not visible anymore in the compensated image. In addition, Figure 4.16 depicts a mesh of the surface before and after correcting the depth image. The red line indicates the ground truth value and the color the magnitude of the error.

In opposite to the discrete model, the continuous model allows to calculate compensation values for distances that are not represented in the input set. Therefore, it is easily possible to apply the model to the synthetic test sequence of an indoor scene that is augmented with the error simulation model developed in Chapter 3. The synthetic scenes are correct with the depth-continuous error model and piped into a TSDF based [72] reconstruction method.

Figure 4.17 shows significant improvements of the surface quality, especially for surfaces that have been observed from longer distances (i.e. ceiling, wall). Surfaces

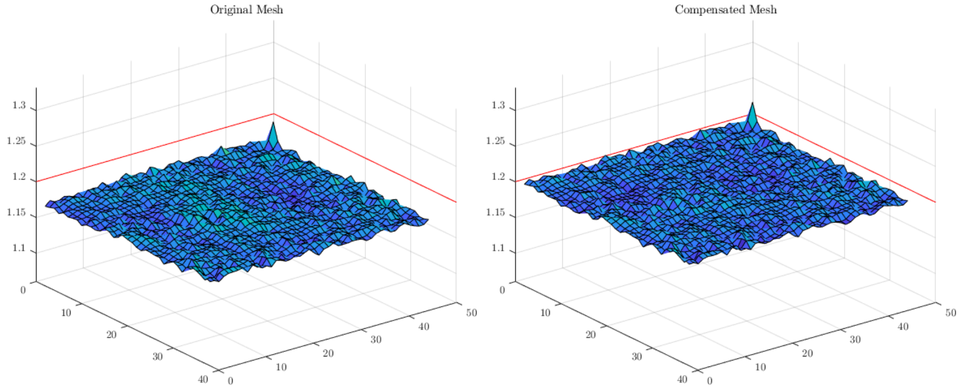
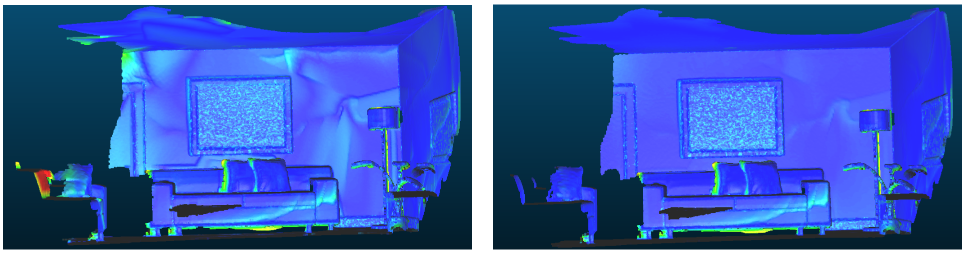


Figure 4.16: Compensation with Full Model.



(a) Without correction.

(b) With correction.

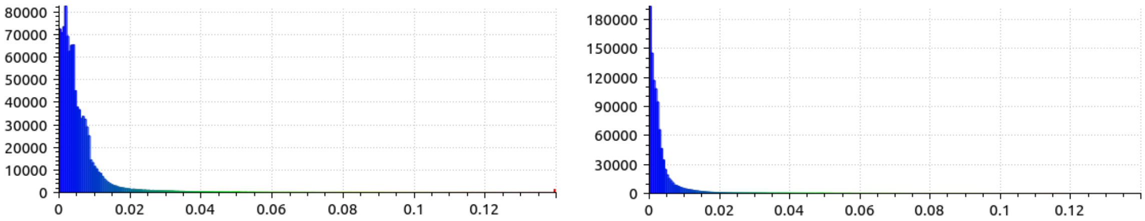
Figure 4.17: Reconstruction using a corrected/uncorrected synthetic dataset of an indoor scene. The dataset is augmented with the depth simulation models presented in Chapter 3.

that have been observed from shorter distances show less or even no improvements. These results are in line with Figure 4.13, which shows that the model has only little or no influence for small depth values. While the introduced exponential functions suppress gross corrections for close observations it allows more significant corrections for far distant observations. This makes perfect sense for real-life sensor data which has exactly these properties (cf. Chapter 2), but falls short together with the synthetic data.

For a quantitative evaluation the reconstructed scene is compared with a ground truth point cloud, extracted from CAD data of the scene. Figure 4.18 depicts the error histograms for the point-plane distance between a point of the reconstruction and the closest least-squares fitted plane (using 6 points) of the ground truth model. The correction significantly reduces the cloud-cloud distance.

4.3.4 Limitations

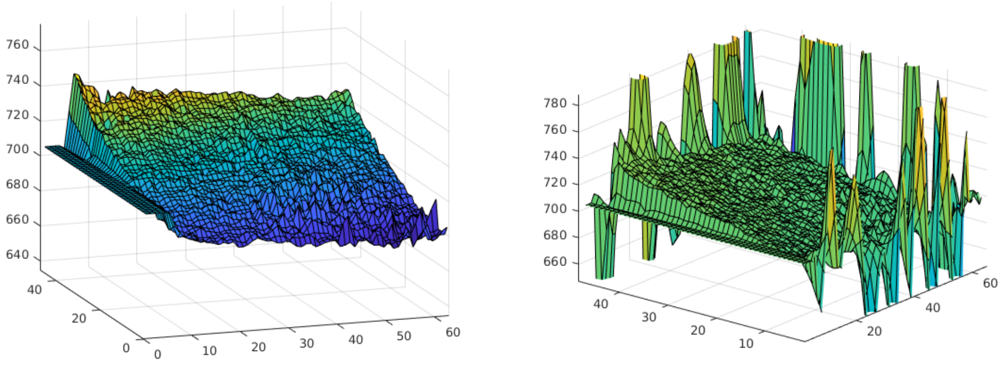
Figure 4.19 shows that the optimization of the interpolation lattice tends to be unstable close to the image borders. The instability originates from interpolation coefficients which are actually outside of the image and necessary for the bicubic interpolation, but are not directly influenced by the error observations. However, the corrected image can simply be clipped to remove these artifacts. Another possible solution might be



(a) Without compensation.

(b) With compensation.

Figure 4.18: Cloud-Cloud distance error histogram, between reconstruction and ground truth CAD model. Absolute cloud-cloud distance to a least-squares fitted plane, k=6.



(a) Original image.

(b) Compensated image with artifacts.

Figure 4.19: Limitations due to sampling in border regions.

to increase the sample density at the image borders. That would directly increase the number of residuals and with that the runtime of the optimization. Moreover the non-uniform sampling might lead to an overrepresentation of border pixels in the optimization problem.

4.3.5 Performance Analysis

The optimization of the compensation models requires the calculation of the corresponding Jacobians and the use of a linear solver. Today automatic differentiation techniques represent the state-of-the-art and are perfectly suited to get quick first results, but might have a poor runtime performance. Hence, the implementation utilizes automatic differentiation techniques that make use of dual numbers and their multidimensional representations (Jets) [80], as well as handcrafted analytic derivatives. In particular, the models include a lower dimensional $[m \times n]$ interpolation lattice that is used to correct the depth images. Section 4.1.2 shows that the Jacobians for this interpolation depend only on 16 entries of the lattice, while all other $((n \times m) - 16)$ entries will result in zero. Regardless of that, the underlying Jets will be of size $m \times n$, which will result in several unnecessary calls when the cost function is evaluated on the Jets. To

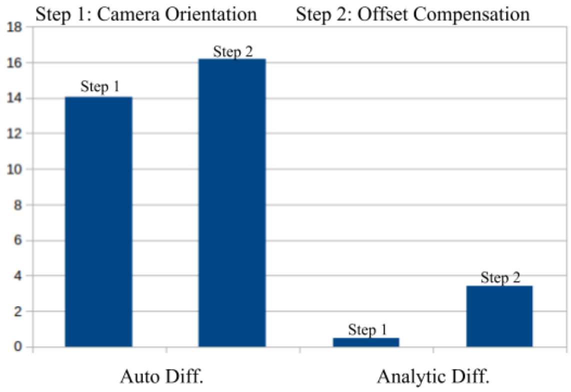


Figure 4.20: Step 1 and Step 2, performance of analytic differentiation automatic differentiation.

circumvent the unnecessary calls, the implementation makes use of analytic Jacobians, which only perform expensive calculations for non-negative entries. To evaluate the impact of analytic and automatic differentiation techniques on the performance, two implementations are tested and compared.

Figure 4.20 shows the total time needed for the two steps of the optimization. While the optimization of the camera orientation shows only a slight impact, the optimization of the compensation function was dramatically reduced.

Figure 4.21 depicts the CPU usage over time during eleven minimization steps. It compares the usage of automatic differentiation (blue) and analytic Jacobians (orange). The Jacobian evaluation is parallelized, while the linear solver is not. Therefore it is possible to distinguish the Jacobian evaluation from the linear solving by looking at the maximum CPU usage (1 core 100%, 16 cores: 1600%). The use of analytic Jacobians reduces the total time for the minimization from 1060s to 380s. While the time for the linear solver stayed the same, the Jacobian evaluation per iteration step is reduced by 95%, from 55s to 3s.

4.4 Discussion

This chapter presents two depth error compensation methods that use flat targets as input data.

Both methods optimize a comparably small set of interpolation parameters which result in high-resolution, pixel-wise, depth-dependent compensation maps. They do not rely on a dedicated sensor model and are directly applied to the depth image as a post-correction step.

The depth-discrete model uses several compensation images for discrete depth distances of the input data and does not implicitly model the depth dependency. In opposite, the depth-continuous model uses only a single compensation image and models the depth dependency with an exponential function, resulting in significantly less parameters than the depth-discrete model.

Both methods are evaluated on synthetic and real-life data. The continuous com-

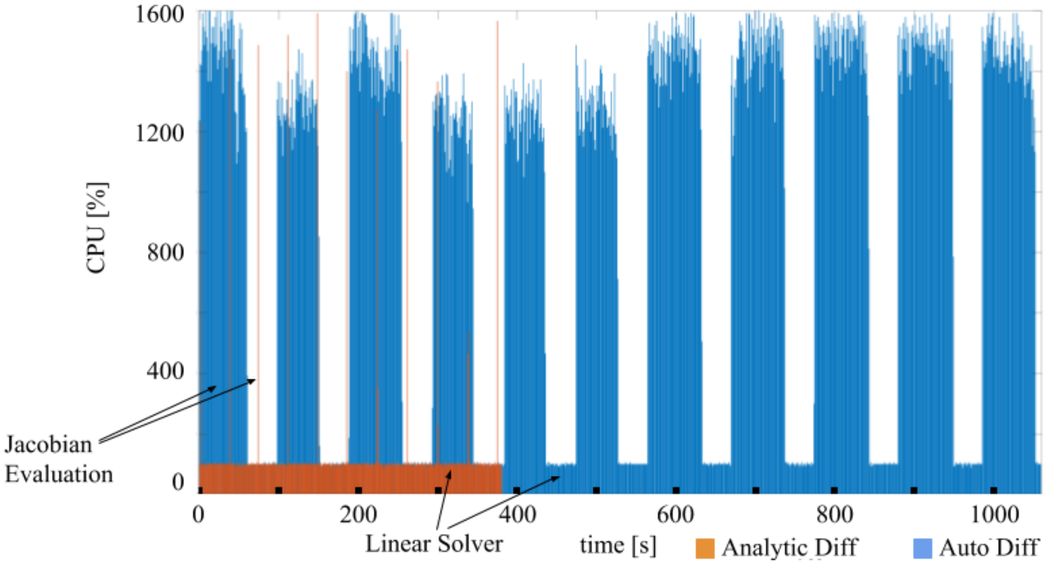


Figure 4.21: Performance of the optimization using automatic differentiation (blue) and handcrafted analytic Jacobians (orange).

pensation method is additionally evaluated together with a reconstruction algorithm and a synthetic dataset of an indoor scene. The synthetic datasets are combined with the depth error simulation method presented in Chapter 3 and provide close-to-real-life depth images together with ground truth information.

The depth-discrete and the depth-continuous model succeeded in improving the trueness and precision on synthetic and real-life data. While the discrete model performs significantly better on the synthetic dataset, the two methods perform equally on the real-life dataset.

The continuous model allows to directly extract continuous depth values even for distances that are not contained in the input dataset. Hence, it is directly applied to the synthetic dataset of an indoor scene. The model shows a clear impact on the surface quality and the map accuracy. It significantly improves the results of the state-of-the-art reconstruction algorithm and generates more precise and accurate depth estimation. This is especially useful for robotic tasks, e.g. grasp planning or the detection of tiny objects.

Both methods significantly improve the depth measurements using planar targets. However, the methods require an expert for the tedious input data collection and therefore might not be feasible under real-life conditions. The next chapter introduces plane-priors into an autonomous calibration method that avoids the complicated data collection step and does not need dedicated calibration targets.

Chapter 5

Autonomous Calibration and Depth Error Compensation of RGB-D Setups

Chapter 2, 3, and 4 focus on depth sensors, their characteristic error behavior, and present methods to simulate and compensate depth errors.

This chapter focuses on the autonomous calibration of RGB-D sensors, which combine a depth sensor with an RGB camera. Depth data alone alreday provides rich information, adding RGB information allows to combine the benefits of both worlds, RGB and depth. For instance, it is easily possible to run 2D object detection methods on the RGB data, and extract the necessary spatial information from the depth data [4]. One key step is to relate the depth information pixel-wise to the color information. Precisely assigning depth measurements to pixels of the color images, requires a precise perspective projection, a decent camera model, and reliable depth information. With these requirements a good calibration is de rigueur. Performing such a good calibration is usually tedious work and requires an expert in the loop. In addition, the calibration might be easily corrupted during operation, what would require a recalibration step in the field during operation. Classic calibration methods are not suited for that scenario: Usually the calibration of sensor setups involves the use of checkerboard patterns and has to be carried out by an expert (cf. Figure 5.1). The required checkerboards and the expert operator might not be available in the field.

Moreover, classic calibration methods do not explicitly compensate a depth offset. Although Chapter 4 contributes two methods for exactly that purpose, they suffer from the same limitations as classic checkerboard-based calibration methods: it might be not feasible or even impossible to autonomously collect the required input data of planar targets together with ground truth information during operation.

Luckily, auto-calibration methods for RGB-D sensors have been addressed in the

Figure 5.1: Camera Calibration to get the intrinsic parameters of the used R200 Active Stereo camera.

state of the art. These methods address some of the limitations of classic methods, but are still far from being perfect. While they work well under controlled environments and with a decent initialization by an expert, they still struggle in real-life robotic indoor scenarios, where they face low image quality, motion blur, and an inaccurate initialization. In [32], Zeisl and Pollefeys develop a promising approach that uses an SfM reconstruction as calibration target. Summarized, the calibration optimizes several parameters to i) optimize the SfM reconstruction, ii) align the depth maps with each other and iii) to align the depth maps with the SfM reconstruction. The method assumes that the SfM reconstruction in general is more precise than the corresponding depth information. Zeisl and Pollefeys do not incorporate any additional assumption about the environment into their method. Unfortunately, SfM reconstructions are not necessarily more precise than depth estimations, which in some situations causes a bad convergence behavior and unsatisfying calibration results. However, SfM reconstructions are usually very precise for feature-rich flat targets that allow close-to-perfect point triangulations. That suggests to directly favor flat areas (planes) during the calibration.

Chapter 4 shows that flat areas are well suited calibration targets to optimize an offset-compensation function, which is similar to the one used in [32].

Moreover, the state of the art indicates that the alignment of 3D point clouds is improved by incorporating plane priors [52].

Hence, this chapter extends the state of the art with an autonomous calibration method that incorporates plane priors in two different ways. First, they are used to favor SfM points which are supported by a plane, and second by using the point-plane distance to improve the alignment between the SfM map and the depth maps. The presented method enables a more robust RGB-D sensor (re)calibration on a robot in the field. The method neither requires a human expert for collecting the data, nor artificial calibration targets. It uses the robot's motion together with an SfM reconstruction to calibrate the sensor and perform a depth offset compensation with the goal to precisely align RGB and depth images.

The following sections present and evaluate the new approach.

5.1 SfM Auto Calibration With Plane Priors

Similar to [32] this work utilizes a sparse map generated by the SfM method presented in [81] as calibration target. SfM maps of texture-rich flat areas tend to be more accurate than reconstructions recovered with just the depth information of a structured-light-like sensor as the Microsoft Kinect. This is due to the fact that color images provide high mega-pixel resolution, which enables a fine-grained feature localization and well constrained 3D point triangulations on flat areas. Moreover, loop closures, even over geometricaly simple scenes, give additional constraints and allow more precise camera pose estimations [32].

First, it is assumed that the SfM reconstruction is given. It provides the camera poses $\mathbf{T_v}$ within the SfM coordinate space, a sparse map M consisting of the 3D reconstructed feature points $\{\mathbf{x}'_\mathbf{i}\}$, and 2D observations of the feature points for every view $\{\mathbf{u}_{\text{obs}}\}$ (cf. Figure 5.2). Moreover, it also provides initial camera parameters for the RGB camera: $[f_x,f_y,c_x,c_y]$ for the pinhole camera model, $[p_1,p_2]$ to model tangential distortion and $[k_1,k_2]$ to model the radial distortion. Given that, $\mathbf{T_v} \circ \mathbf{x_i'}$ transforms a point from the global reference frame into any other view.

The classic pinhole camera model in combination with a radial and tangential distortion term projects an arbitrary point $\mathbf{x}$ from the 3D space onto the image plane with

$$\begin{aligned} \mathbf{u} = \pi(\mathbf{x}) =& \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_x & c_y \end{bmatrix} \begin{bmatrix} \check{\mathbf{x}} \\ 1 \end{bmatrix} \\ \text{s.t.} \quad \check{\mathbf{x}} =& L(r^2)\mathbf{x_n} + \mathbf{t}(\mathbf{x_n}) \\ \mathbf{x_n} =& \left[x \cdot z^{-1}, y \cdot z^{-1}\right]^T \qquad r^2 = \|\mathbf{x_n}\|_2^2 \\ L(r^2) =& 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 \\ \mathbf{t}(\mathbf{x}) =& \begin{bmatrix} 2xy & r^2 + 2x^2 \\ r^2 + 2y^2 & 2xy \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}. \end{aligned} \tag{5.1}$$

Now, the model is extended to incorporate the presence of the depth sensor. $\mathbf{T_R}$ models the relative pose between the cameras and s as the constant scale factor that is necessary to get a to-scale SfM reconstruction. With that it is possible to transform a point $\mathbf{x_i'}$ from the sparse map into the current depth camera frame as $\mathbf{x_D} = \mathbf{T_R} \circ s \circ \mathbf{T_v} \circ \mathbf{x_i'}$.

The related work presents several depth compensation methods that strongly vary in their complexity. Due to its simplicity and optimization-friendly properties the depth offset compensation from [32] is used which is similar to the depth-continuous model introduced in Chapter 4. It interpolates a lower dimensional lattice β with a bicubic interpolation $\mathbf{c}(\mathbf{u})$ and combines it with an exponential function. The depth offset compensation δ is given as

$$\delta(\mathbf{u},d) = c(\mathbf{u}) \cdot exp(a_0 - a_1 d). \tag{5.2}$$

The exponential function is parameterized over the inverse depth d [82] and two coefficients a_0 and a_1. For more details about the bicubic interpolation, especially the Jacobians, see Chapter 4.

With that the calibration model is finalized and equivalent to the model used in [32]. The calibration parameters are denoted as $\theta = \{\mathbf{x_i'}, \mathbf{T_v}, s, \mathbf{T_R}, \pi_c, \pi_D, \beta, \mathbf{a}\}$, where $\pi_{c,D}$ represent the intrinsic parameters of the depth and color camera.

5.1.1 Residuals

The approach introduces four residuals. The first three residuals (reprojection error, inverse point-point distance, and the residuals that maintain the alignment between views) are already discussed in a comprehensive manner in related work [32], [34], [57]. Hence, the motivation is briefly repeated before a fourth residual that maintains the alignment to the map by utilizing the point-plane distance is added .

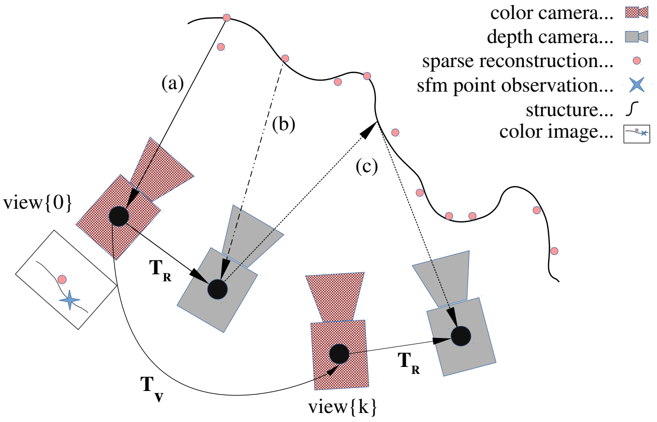


Figure 5.2: Overview of the calibration problem. (a) Depicts the reprojection error $\mathbf{r^B_{vi}}$ that is used for the bundle adjustment. (b) Shows the projection of an SfM map point into the depth camera frame, used for the inverse point-point distance r^I_{vi} and the point-plane distance r^P_{vi}. (c) Visualizes the projection of the depth value from one depth camera into a second view, used for the residual that models the alignment between a view pair j,k with r^V_{jk}.

Reprojection Error

The reprojection error models the deviation between the observation of the feature in the color image and the reprojection of the 3D feature into the image (cf. Figure 5.2 (a)). The residual prevents strong deformations of the map and camera poses. A Huber loss function ρ_B reduces the influence of outliers.

$$\begin{aligned} \mathbf{r^B_{vi}}(\theta_I) &= \mathbf{u}_\text{obs} - \mathbf{u} \\ \text{s.t.} \quad \mathbf{u} &= \pi_c(\mathbf{T_v} \circ \mathbf{x'_i}), \end{aligned} \tag{5.3}$$

with $\mathbf{u}_\text{obs}$ as the feature based 2D point observation of $\mathbf{x'_i}$ for the current view, $\pi_\mathbf{C}$ as the camera model for the color camera, $\mathbf{T_v}$ transforming a 3D point into the current view, and $\mathbf{x'_i}$ as the observed point from the sparse map.

Depth Map Alignment Between Views

This residual considers only the depth measurements of the depth cameras (cf. Figure 5.2 (c)). It wraps the alignment of the depth maps into the calibration problem.

$$\begin{aligned} r^V_{jk}(\theta) = (\mathbf{T_{jk}} \circ \mathbf{\tilde{y}_j})_z^{-1} &- D_k(\mathbf{u})^{-1} - \delta(\mathbf{u}, D_k(\mathbf{u})^{-1}) \\ \text{s.t.} \quad \mathbf{u} &= \pi_D(\mathbf{T_{jk}} \circ \mathbf{\tilde{y}_j}) \\ \mathbf{T_{jk}} &= \mathbf{T_R} \circ s \circ \mathbf{T_k} \circ \mathbf{T_j}^{-1} \circ \frac{1}{s} \circ \mathbf{T_R}^{-1} \\ \mathbf{\tilde{y}_j} &= \pi_\mathbf{D}{}^{-1}(\mathbf{v}, y_{j,z}) \\ y_{j,z}^{-1} &= D_j(\mathbf{v})^{-1} + \delta(\mathbf{v}, D_j(\mathbf{v})^{-1}), \end{aligned} \tag{5.4}$$

with $\mathbf{T_{jk}}$ transforming a 3D point from view j into view k, $\mathbf{\tilde{y}_j}$ as a point gathered from a depth lookup $D_j(\mathbf{v})$ in view j, δ as the offset compensation, and $\pi_\mathbf{D}$ as the camera model for the depth camera.

Inverse Point-Point Distance

The inverse point-point distance is similar to the ICP algorithm. It aims to align each depth map D_v to the sparse map M (cf. Figure 5.2 (b)). The inverse depth parameterization is used, to inherently model the close range accuracy. A Tukey loss function rejects outliers of the sparse map.

$$\begin{aligned} r^I_{vi}(\theta) &= \mathbf{x}^{-1}_{D,z} - D_v(\mathbf{u})^{-1} - \delta(\mathbf{u},D_v(\mathbf{u})^{-1}) \\ \text{s.t.} \quad \mathbf{x_D} &= \mathbf{T_R} \circ s \circ \mathbf{T_v} \circ \mathbf{x'_i} \\ \mathbf{u} &= \pi_D(\mathbf{x_D}) \end{aligned} \tag{5.5}$$

Point-Plane Distance

The motivation for the incorporation of plane priors into the optimization is two-fold. First, man-made indoor scenes are dominated by planes and points on a plane tend to be more stable than points that are on fine structures. Second, the first tests revealed that in some situations the inverse point-point distance fails to perfectly align the depth values to the SfM map, especially when the initial parameters for s and $\mathbf{T_R}$ are too far off. The sparsity of the SfM reconstruction in combination with the use of the aggressive Tukey loss function directs the problem into a local minimum, which restricts the solution to converge to the desired result (cf. Figure 5.3). One solution that may work in some situations is to simply adapt the Tukey loss function for every new problem. However, tweaking optimization parameters for each and every single scene is not an option for an unsupervised calibration that is re-run autonomously on a robot. As such, (5.6) introduces a residual that models the point-plane distance for an improved convergence behavior in indoor scenes that are often dominated by planes. During the first iterations it acts as the point-point distance, but as soon as the two planes are aligned they can move freely at zero cost, using the plane to guide the solution.

The residual that models the distance between a sparse point of the map and the corresponding plane of the depth map is given as

$$\begin{aligned} r^{\mathrm{P}}_{vi}(\theta) &= \mathbf{n}(\mathbf{u}) \bullet (\mathbf{x_D} - \mathbf{\tilde{x}}_\delta) \\ \text{s.t.} \quad \mathbf{x_D} &= \mathbf{T_R} \circ s \circ \mathbf{T_v} \circ \mathbf{x'_i} \\ \mathbf{u} &= \pi_\mathbf{D}(\mathbf{x_D}) \\ x^{-1}_{\delta,z} &= D(\mathbf{u})^{-1} + \delta(\mathbf{u},D(\mathbf{u})^{-1}) \\ \mathbf{\tilde{x}}_\delta &= \pi_\mathbf{D}{}^{-1}(\mathbf{u},x_{\delta,z}), \end{aligned} \tag{5.6}$$

with $\mathbf{n}(\mathbf{u})$ as the normal vector at a location $\mathbf{u}$ in the depth frame, and $\mathbf{x_D}$ as the 3D feature point transformed to the current view of the depth camera. $\mathbf{\tilde{x}}_\delta$ represents the

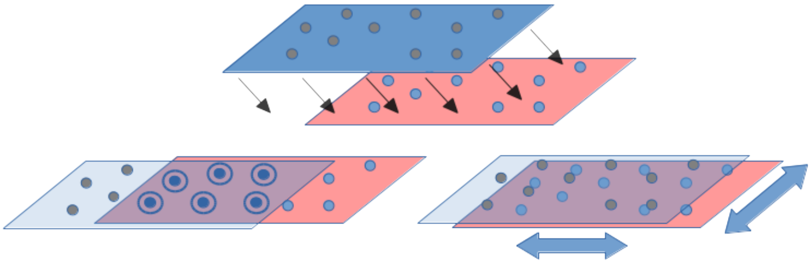


Figure 5.3: Alignment of two point sets, point-point distance (left) vs. point-plane distance (right). While the point-point distance prevents movements of the points within the plane, the point-plane distance, allows movements at zero costs.

back projected depth value observed at the location $\mathbf{u}$ corrected with the depth offset. With the normal vector $\mathbf{n}(\mathbf{u})$ and the two points $\mathbf{x_D}$ and $\tilde{\mathbf{x}}_\delta$ the point-plane distance is calculated using the inner product.

The calibration will directly affect the depth values which are used to calculate the normal vectors. However, since the corrections will be smooth and not radical it can be assumed that the calibration will not heavily influence the normal vectors of the dominant planes. It is therefore possible to pre-compute the normal vectors for every depth map to reduce computational costs during the optimization. In addition, the depth maps are smoothed with a Gaussian filter before the computation of the normal vectors, with the goal to achieve smooth normal vector transitions and derivatives.

5.1.2 Weighting Functions

To achieve proper convergence behavior, optimization problems demand loss functions that penalize outliers. Finding and defining the correct parameters for the loss function is often tedious work. The same applies to the weights, which have the potential to significantly influence the convergence behavior and the contribution of the different residuals. Hence, different weighting strategies are implemented and investigated that i) incorporate a plane prior and ii) balance the introduced residuals.

Equalized Plane Weighting (WIPD,WPLD)

The approach assumes that points on a plane tend to be more stable and accurate than single points that are not supported by a plane and may represent small structures. Based on this assumption, it introduces a weighting that favors points supported by a plane. Therefore, the method runs a simple plane detection on the sparse reconstruction. If the scene is dominated by large planes such as walls or the floor, those planes may dominate the calibration problem, suppressing the influence of tiny planes that would provide additional information. To avoid that, the sum of the weights for every plane that is considered within the optimization is limited to 1. That equalizes the influence of the planes, independent of their size, and favors points on tiny planes (e.g. on objects).

$$w_{\text{IPD,PLD}} = \frac{1}{m}, \tag{5.7}$$

with m as the number of plane inliers for the currently observed plane. This weighting strategy is incorporated into the experiments for both, the inverse point-point distance (IPD) and the point-plane distance (PLD) using the additional weights $w_{\text{IPD,vi}}$, $w_{\text{PLD,jk}}$ for each residual instance.

Residual Balancing (rB)

Since the number of residual instances per residual group strongly depends on the dataset, it is unclear how much each residual group will contribute to the optimization problem, and how it will influence the result. However, it is possible to control the influence of every residual with the weights λ. To regain more control and add dataset independence without manually adapting λ for every residual and calibration problem, the preset residual group weighting factors $\check{\lambda}$ are normalized with the number of corresponding residual instances n, if the equalized plane weighting is enabled with $\sum w_{\text{IPD,PLD}}$ respectively.

$$\lambda_{\text{B,V}} = \frac{\check{\lambda}_{B,V}}{n} \qquad \lambda_{\text{I,P}} = \frac{\check{\lambda}_{\text{I,P}}}{\sum w_{\text{IPD,PLD}}} \tag{5.8}$$

With the weighting strategies, the optimization problem is finally formalized as

$$\begin{aligned}\min_{\theta} \lambda_B \sum_{v,i} \rho_B \left(\|\mathbf{r}^{\mathbf{B}}_{\mathbf{vi}}(\theta)\|^2 \right) &+ \\ \lambda_V \sum_{j,k} \rho_V \left(\|r^V_{vi}(\theta)\|^2 \right) &+ \\ \lambda_I \sum_{v,i} w_{\text{IPD,vi}} \cdot \rho_I \left(\|r^I_{vi}(\theta)\|^2 \right) &+ \\ \lambda_P \sum_{v,i} w_{\text{PLD,jk}} \cdot \rho_P \left(\|r^P_{vi}(\theta)\|^2 \right) &,\end{aligned} \tag{5.9}$$

with ρ_B representing a Huber loss function and $\rho_{V,I,P}$ a Tukey loss function.

5.1.3 Optimization

For the optimization it is necessary to calculate the Jacobians with respect to the parameter vector θ. Chapter 4 discusses already the Jacobians of the depth correction method. To calculate the partial derivatives for the newly introduced point-plane distance, the Jacobians for the normal vectors $\mathbf{n}$ are needed. It is assumed that the depth correction will not radically influence the normal vector obtained from the depth map. With this assumption, it is trivial to compute the Jacobians for the normal vector. Moreover, they can be computed in advance to speed up the optimization problem.

$$\left[\frac{\partial \mathbf{n}}{\partial \mathbf{u}}\right] = \left[\frac{\partial n_x}{\partial \mathbf{u}}, \frac{\partial n_y}{\partial \mathbf{u}}, \frac{\partial n_z}{\partial \mathbf{u}}\right]^T \tag{5.10}$$

The unprojected point $\tilde{\mathbf{x}}_\delta$ depends on the inverse projection function ${\pi_\mathbf{D}}^{-1}(\cdot)$ which cannot be solved in closed form. Because of this, first tests were carried out with the numeric differentiation functionalities to obtain the Jacobians. However, the tests revealed no noticeable improvements in the results or for the convergence behavior, hence the corresponding entries in the Jacobian are set to zero to speed up the calibration.

For the rest of the optimization problem the implementation makes heavy use of the Ceres solver library [77] and its automatic differentiation capabilities.

The optimization itself contains five steps.

Step 0: Since the third party SfM reconstruction technique presented in [81] is used, the optimization reruns the Bundle Adjustment using the reprojection error residual, with fixed RGB camera intrinsics.

Step 1: Optimizes over the scale factor s, the relative pose between the RGB camera and the depth sensor $\mathbf{T_R}$, and the intrinsic parameters of the depth camera, but keeps the distortion coefficients constant.

Step 2 (v4): Optimizes solely for the exponential function, setting the lattice to 1.

Step 3: Exclusively optimizes the parameters for the depth offset compensation δ. Namely, the interpolation coefficients and the exponential function parameterized over the inverse depth $[\beta,\mathbf{a}]$.

Step 4: Optimizes over all parameters for a last refinement.

5.1.4 Implementation

The implementation heavily utilizes the Ceres solver library **-solver** for the optimization, together with OpenCV [83] and the PCL [84]. The SfM reconstruction is obtained using the openMVG framework [85]. The application of the calibration uses Streaming SIMD Extensions (SSE) and utilize openMP [86] for parallelization. With that it extracts a corrected and projected depth map in average within 12ms and a colored point cloud in average within 17ms.

5.2 Evaluation

All in all the approach introduces five different extensions of the state of the art that use the inverse point-point distance (IPD) to align the depth to the SfM reconstruction: The use of the point-plane distance (PLD); two weighting functions, incorporating plane priors (WPLD, WIPD); the residual Balancing (rB); and the additional optimization Step 2 (v4). The experiments are automatized to execute experiments with every possible combination of the newly introduced extensions and add additional noise to the SfM reconstruction in order to test the robustness of the algorithm on low quality data.

The approach is evaluated against the baseline algorithm presented in [32]. Therefore, the evaluation replicates the qualitative experiments presented to verify the alignment capabilities of the introduced algorithm. Additionally, it comes up with two quantitative evaluation methods that aim to measure the quality of the results. Finally, this section

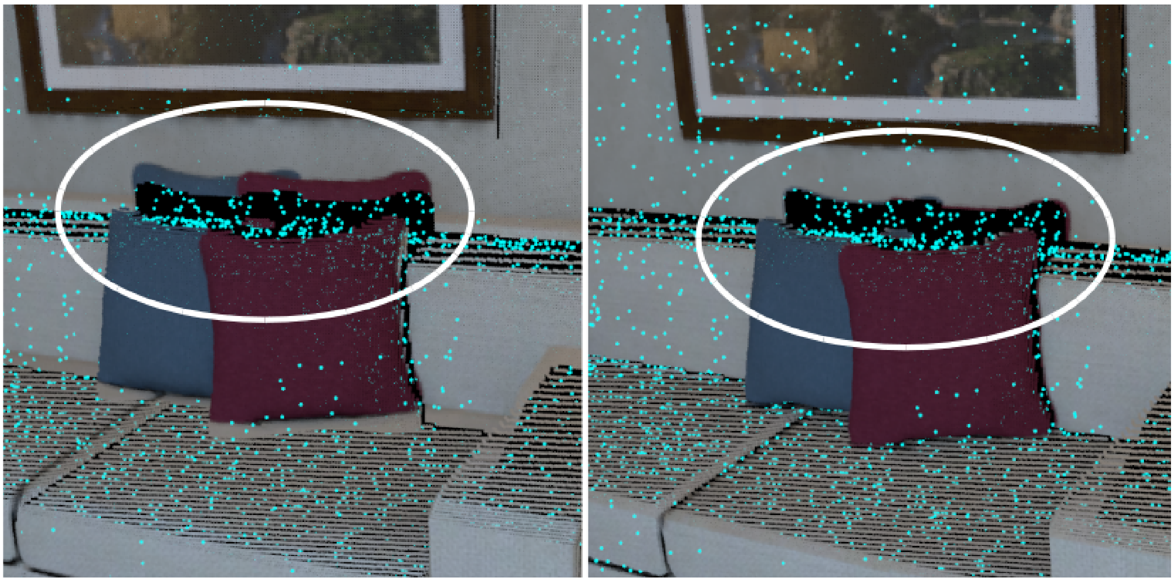


Figure 5.4: Results for the ICL-NUIM dataset using the method from [32]. The circles highlight the improved alignment of color and depth data.

shows the usability of the calibration in the use case of semi-automatically annotating an RGB-D dataset for deep learning.

5.2.1 Baseline Validation

Zeisl and Pollefeys [32] did not publish their datasets or algorithm, hence their algorithm is reimplemented to provide a baseline for the experiments. To verify that the implementation of [32] is working as expected, it is tested on the ICL-NUIM dataset [71]. The dataset provides artificial scenes and hard ground truth data for all parameters that are determined during the calibration. Noise is added to the parameters and the optimization is run to verify that the implementation of the related work converges to the ground truth parameters (cf. Figure 5.4). However, although the implementation conscientiously follows the steps presented in [32] and the results show that the method succeeded in determining the ground truth calibration parameters on the ICL-NUIM sequence, it still cannot be guaranteed that the reimplementation will give exactly the same results as the original work.

5.2.2 Datasets

After the first tests nine different datasets are collected showing indoor scenes, two sets containing heavily motion blurred images. Additionally, the publicly available TUM dataset sequence *"fr3/structure_texture_far"* [87] is added to the test set. Table 5.1 gives an overview of the used datasets.

Table 5.1: Dataset Overview

Dataset	#views	#features	#observations
TUM STF [87]	91	3177	36926
office	150	3726	34164
office+blur	194	4161	37343
home 1	218	6729	124024
home 2	110	3674	19892
home 2+blur	147	4187	23454
corner 1	128	3062	35906
corner 2	47	1586	12977
corner 3	65	1704	18795
workplace	124	2198	30246

5.2.3 Qualitative and Quantitative Evaluation

In [32] Zeisl and Pollefeys evaluate their method against the factory calibration and a manual calibration by superimposing the projected depth image and the corresponding RGB image. The evaluation replicates this qualitative method and verifies the results against the reference method and a handcrafted initial parameterization.

The experiments show that the manual evaluation tends to be ambiguous in some situations. Hence, the experiments are extended with two quantitative evaluation metrics:

They are designed to give a quantitative measurement that on one hand verifies the qualitative results, and on the other hand may be used in future as quality indicator for a completely autonomous calibration. However, simply using the final residuals of the optimization as quality criteria would not be sufficient to compare the different optimization problems, because they use different weighting functions and residuals. Hence, this work comes up with a more general but straight forward approach.

The optimization aims to achieve two goals: First, it tries to align the depth maps as precise as possible to the SfM reconstruction. Second, it aims at aligning depth maps from different views as precise as possible. The residuals introduced try to achieve exactly those two goals. Hence, a depth map corrected with a perfect calibration and aligned to a flawless SfM reconstruction will result in an alignment error of zero for all points. The same accounts for the alignment of two depth maps from different views. With that assumption two quantitative metrics QMSfM and QMDepth are defined.

First, QMSfM calculates the cloud-cloud distance between the scaled SfM reconstruction (SfM cloud) and a point cloud extracted from the depth information of the view that offers the most observations (depth cloud). The information needed is extracted using the software CloudCompare[1]. Due to the sparseness of the SfM cloud, the depth cloud is used as reference. With that the evaluation does not use any interpolation strategy

---

[1] CloudCompare (version 2.9) [GPL software]. (2017). Retrieved from http://www.cloudcompare.org/, PCL interfaces developed by Luca Penasa

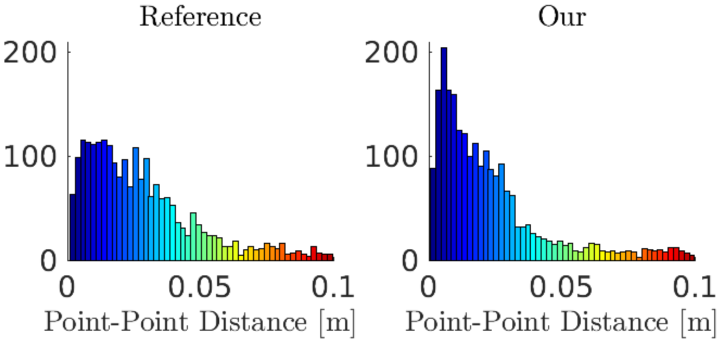


Figure 5.5: QMSfM results for the reference method [32] and the presented method (only PLD enabled) on the office+blur dataset.

that may influence the results. Figure 5.5 shows the histogram of the quantitative results using QMSfM. It depicts the results of the method and the reference method on the *"office+blur"* dataset. To quantitatively compare the histograms of all experiments, outliers are removed with a simple thresholding and the mean cloud-cloud distance is extracted, what finally provides QMSfM.

Second, QMDepth verifies the quality of the depth compensation with a similar method. It extracts the view with the highest number of observations and searches for the view that has an observation overlap ratio

$$\eta(D_{\max},D_j) = \frac{\#\text{observations}(D_j)}{\#\text{observations}(D_{\max})}, \tag{5.11}$$

of at least $t_o = 0.5$ and maximizes the difference of the means of the two corresponding depth maps.

$$\begin{aligned} & \max_j \left(\mu(D_{\max},D_j)\right) \\ \text{w.r.t.} \quad & \eta(D_{\max},D_j) > t_o \\ \text{s.t.} \quad & \mu = \left|\overline{\mu}_{\max}(D_{\max}) - \overline{\mu}_j(D_j)\right| \\ \overline{\mu}_{\max,j}(D_{\max,j}) & = \frac{1}{n+m}\sum_{x,y}^{n,m} D_{\max,j}(x,y). \end{aligned} \tag{5.12}$$

Then, the point clouds for the two views are generated and the mean distance between the two clouds is computed as QMDepth.

5.2.4 Results

The evaluation performs a total of over 300 experiments on 10 different datasets, without manually adapting the optimization or loss function parameters to the datasets. The experiments are summarized in Table 5.2.

The implementation automatically generates an RGB and depth overlay for all experiments and all images of the datasets. The overlay is used to manually rate the

Table 5.2: Qualitative and quantitative results of the calibration method and combinations outperforming the state of the art. Example: Method that gives the overall best result (see column "$\sum$"): PLD enabled. Method that gives the best qualitative result (see column "man"): PLD, IPD, and WIPD enabled. A full table with all results can be found in the appendix.

Experiments							Results			
#	PLD	IPD	WPLD	WIPD	v4	rB	QMDepth	QMSfM	man	$\sum$
1	1	0	0	0	0	0	**211**	**198**	**228**	***637***
2	0	1	0	0	1	1	***214***	**217**	157	**588**
3	1	0	1	0	0	1	**197**	**179**	**189**	**565**
4	0	1	0	1	0	0	**197**	144	**221**	**562**
5	1	1	1	0	0	1	**194**	***218***	149	**561**
6	1	1	0	1	0	0	167	151	***240***	**558**
7	0	1	0	0	0	1	189	**209**	150	548
*8	0	1	0	0	0	0	172	**185**	**190**	547

* Reference method introduce in [32]

results according to the qualitative method described in the previous section. Figure 5.6 shows an excerpt of the results for two different scenes.

After extracting both the qualitative and the quantitative results the different methods are ranked and points are assigned according to their score. The experiments achieving the best results got 32 points (=#experiments) while the weakest experiment, and experiments that were already removed during the qualitative evaluation received 0 points. The higher the score, the better the result. All scores are separately summed up over the different datasets for the qualitative and the quantitative measures. The six highest scores are printed bold, while the best result is printed bold and italic. The experiments in Table 5.2 were ranked based on the sum over all scores (cf. Table 5.2, $\sum$). Experiment 8 is the reference method from [32]. The 0/1 indicates which method is used for the experiment. i.e. "PLD:1, IPD:0, WPLD:0, WIPD:0, v4:0, rB:1" indicates that the point-plane distance was used (PLD:1), while the inverse point-point distance was not added to the problem (IPD:0), the plane-based weighting functions were deactivated (WIPD:0, WPLD:0), the optimization skipped step 2 (v4:0), and the residual balancing was enabled (rB:1).

The results revealed seven new residual and weighting combinations that achieved better results than the reference method.

5.2.5 Use Case Scenario

Loghmani et al. [9] recently introduced a novel dataset that is designed to verify deep learning results on real-life RGB-D data gathered with a robot.

Even for a trained expert in some situations it is impossible to assign the correct label

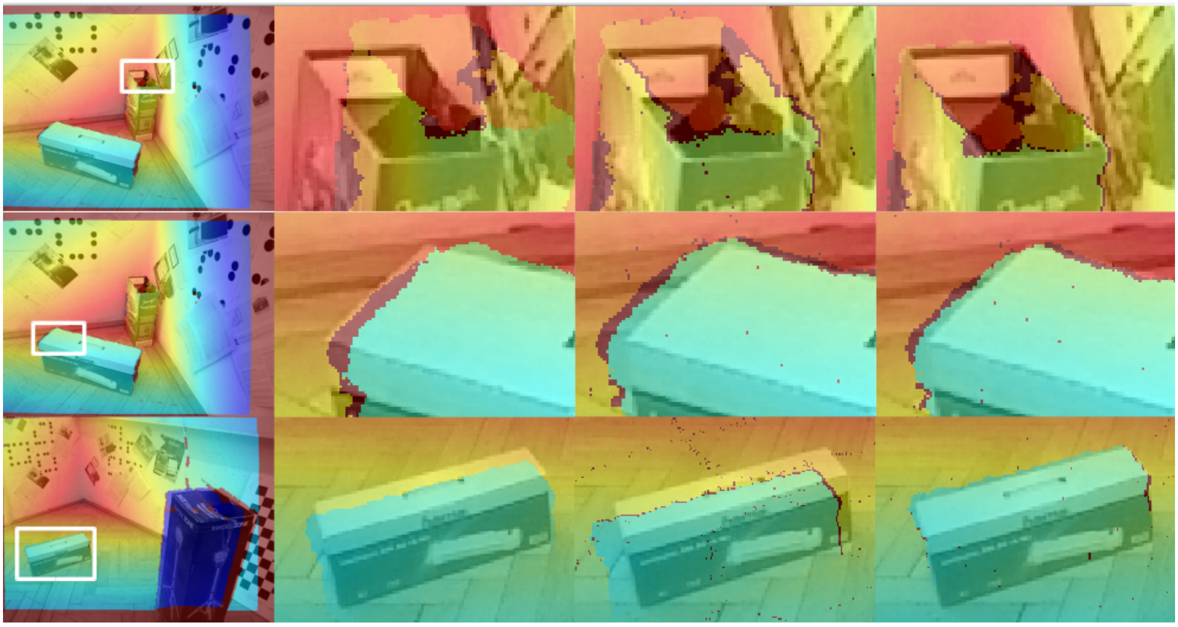


Figure 5.6: Qualitative results of the calibration. From left to right: scene, initial parameterization, reference method (IPD, [32]), the introduced method that gives the best qualitative result, using PLD, IPD, and WIPD. Figure best viewed in color.

to regions of a depth image because the information is ambiguous. However, usually it is easy to spot the correct label and boundaries in the RGB domain. For this reason, the dataset is labeled in the RGB domain. To extract the correct labels for the depth data, the camera is calibrated and the data is corrected using the presented method. The depth data is projected into the RGB domain and achieves more precise labeling results than with a manual calibration. Figure 5.7 shows the results with and without the presented calibration.

5.3 Discussion

This chapter presents an autonomous calibration method for RGB-D setups. The approach makes use of an SfM reconstruction as calibration target and formulates an optimization method that incorporates plane priors to improve the pixel-wise alignment of RGB and depth images. It introduces a new residual that uses the point-plane distance for a better convergence behavior and adds different weighting strategies that i) balance the residuals independent of their dataset-dependent over- or underrepresentation and ii) increase the influence of plane points which tend to be more stable.

The evaluation contains over 300 experiments performed on ten datasets. It qualitatively and quantitatively evaluates different variants of the newly introduced methods against a reimplementation of a state-of-the-art auto-calibration method. Therefore, two novel quantitative evaluation methods are defined that use the cloud-cloud distance for an impartial analysis of the auto-calibration results. The quantitative methods could

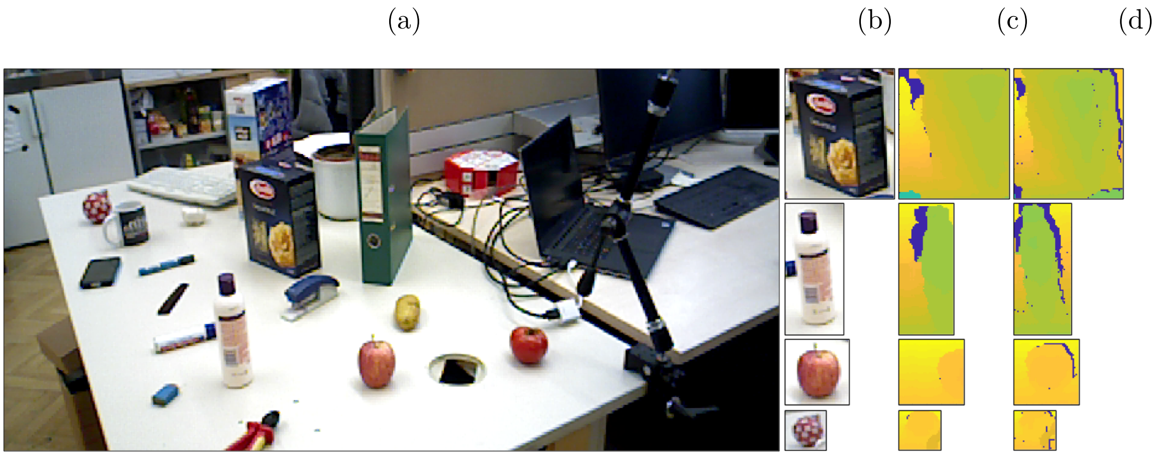


Figure 5.7: Annotation results for a scene (a) of the dataset presented in [9]. (b) Shows the manual annotation in the RGB domain. The last two columns show the depth annotation results for a manual calibration (c) and with the presented calibration (d).

be used in the field to check if the autonomous calibration was successful. The results of the qualitative and the quantitative evaluation show significant improvements which are reflected in more precisely aligned RGB and depth images, as well as more precise depth measurements. I.e. the results revealed seven new residual and weighting combinations that achieved better results than the reference method. Moreover, the performance of the reference method was improved by adding the equalized plane weighting (WIPD). The best qualitative results was achieved by combining the point-plane distance with an equalized plane weighted inverse point-point distance (PLD, IPD, WIPD). The results also show that the quantitative measures (QMSfM, QMDepth) confirm and support the qualitative results (man) that were obtained manually.

The usability of the calibration method is demonstrated in the use-case scenario of semi-automatically annotating the depth information of a novel RGB-D dataset for deep learning, recorded with a robot [9]. In the presented use-case scenario the RGB data has been labeled manually, but given a good camera calibration it is easily possible to use existing RGB classification methods to label depth data by simply utilizing a perspective projection.

The introduced auto-calibration method calibrates the intrinsic and extrinsic parameters of an RGB-D sensor and corrects the depth information. In order to relate the improved depth information to a reference coordinate system it is necessary to find a rigid transformation between the involved coordinate systems. To our knowledge so far there is no auto-calibration method for RGB-D sensors that combines the calibration of the sensor intrinsics parameter, the sensor extrinsic parameters, a depth offset correction method, and the calibration of the sensor pose relative to a reference coordinate system into one holistic approach. Hence, the next chapter develops a camera pose (hand-eye) calibration method that can be easily combined with the presented auto calibration approach into one holistic solution.

Chapter 6

Towards Holistic Autonomous Calibration

Chapter 5 introduces a method for the autonomous calibration of RGB-D setups. It covers the estimation of the intrinsic camera parameters, the rigid transformation between the RGB and depth camera, as well as a depth offset compensation. With that, the camera itself is completely calibrated. However, on a robotic system the sensor information might be useless if they cannot be related to other components on the robot, for instance the navigation system to avoid detected obstacles or grasp detected objects. Hence, it is necessary to define the rigid transformation between the cameras and a fixed reference coordinate system on the robot. The rigid transformation is usually manually measured and error prone.

We introduced in [39] a method to replace the manual measurement with an autonomous calibration approach. It uses the robot's kinematic configuration, together with predefined circular, respectively linear trajectories and a ground plane detection. Although the method proves its applicability in practice, it has some limitations: it requires the robot to perform specific motions which strongly depend on the robot's kinematic configuration and does not take any positioning errors into account.

This chapter breaks up these restrictions and briefly presents first results of an extrinsic pose calibration algorithm that could extend the introduced auto-calibration method to form a holistic calibration approach. It filters the input data using a motion blur detection algorithm to gain robustness and uses the filtered input data to reconstruct the motion of the robot to calibrate the rigid transformation between the camera and a reference system. The optimal method does not rely on predefined trajectories and is not restricted to a special kinematic configurations. The introduced solution is related to the classic hand-eye calibration problem, except that there is no precise estimation of the robot trajectory available and it aims to calibrate the rigid transformation between two sensors using the three dimensional (x,y,θ) motion of the robot.

6.1 Camera Pose Calibration

This section introduces a method which is tailored to use information that is already available on most state-of-the-art robotic systems (cf. Figure 6.1). The approach calibrates the rigid transformation between the two coordinate systems and makes use of the camera trajectory and the robot's trajectory. It formulates a simple target

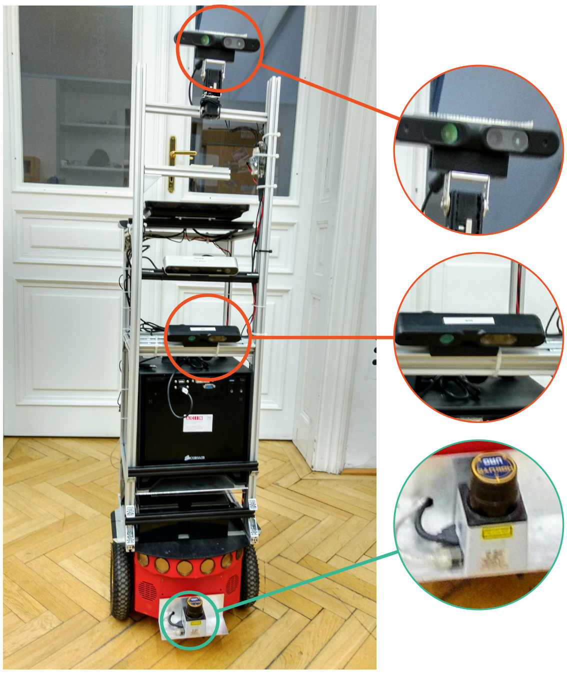


Figure 6.1: Sensor configuration of the experimental platform. Orange: RGB-D sensors, blue: Laser scanner.

function which is used together with a Levenberg-Marquardt optimization and a ground plane detection to estimate the required parameters.

6.1.1 Robot Trajectory Estimation

Many state-of-the-art indoor robots use a line laser scanner for robust localization and mapping. More accurately, the navigation and relocalization system estimates the current pose of the laser scanner relative to a map. The robot makes use of a rigid transformation between the sensor and the other robot components to relate them to a global coordinate system.

The research robot used in this work and depicted in Figure 6.1 combines the laser measurements with a Monte Carlo Localization (MCL) method introduced by Fox et al. in [88]. MCL is a sample based algorithm that uses particles (samples) to represent the current pose estimate and the corresponding multi-modal distribution of the robot pose and enables a global relocalization and tracking of the robot. The algorithm is known to be efficient, fast, and robust under various conditions.

However, the calibration approach is agnostic to the underlying localization method. If necessary, the MCL algorithm can be replaced with any other state-of-the-art localization method.

6.1.2 Camera Trajectory Estimation

Similar to the auto calibration approach in Chapter 5, the camera trajectory is estimated using the SfM technique presented in Moulon et al. [81]. It is combined with an additional bundle adjustment [57] step to get the optimal camera trajectory and the corresponding structure of the environment which is used to initialize the scale of the camera trajectory (cf. Chapter 5). With that it can be easily combined with the previously introduced auto-calibration method.

The camera trajectory could be alternatively estimated with a dedicated camera tracking method as presented in [89]–[91].

6.1.3 Extrinsic Pose Calibration

The extrinsic pose calibration uses the camera trajectory and the robot trajectory to estimate the rigid transformation $\mathbf{T_{Ext}}$. $\mathbf{T_{Ext}}$ transforms the motion of the robot from one reference system into the other.

Figure 6.2 shows that $\mathbf{T_{Ext}}$ can be expressed together with the relative pose changes of the two coordinate systems as

$$\mathbf{T_{Ext}} \circ {}^{\mathbf{L,i}}\mathbf{T_{L,i+1}} \circ \mathbf{T_{Ext}^{-1}} = s \cdot {}^{\mathbf{C,i}}\mathbf{T_{C,i+1}} \tag{6.1}$$

where ${}^{\mathbf{L,i}}\mathbf{T_{L,i+1}}$ is the transformation between the robot's pose at time i and $i+1$, s as the scale, and ${}^{\mathbf{C,i}}\mathbf{T_{C,i+1}}$ as the transformation between two camera poses at time i and $i+1$.

This definition has two major advantages: First, the relative motion between two consecutive frames tends to have less errors. Second, it is independent of the origin of the two coordinate systems.

With that, it is possible to formulate a residual with

$$\mathbf{r} = \mathbf{T_{ext}} \circ {}^{\mathbf{L,i}}\mathbf{T_{L,i+1}} \circ \mathbf{T_{ext}^{-1}} s^{-1} \cdot ({}^{\mathbf{C,i}}\mathbf{T_{C,i+1}})^{-1} = [x,y,z,a_x,a_y,a_z]^T \tag{6.2}$$

where $[x,y,z,a_x,a_y,a_z]^T$ is a 6D pose, with the translation $[x,y,z]$ and the rotation in angle-axis notation $[a_x,a_y,a_z]$. If the rigid transformation is correctly parameterized $\mathbf{T_{ext}}$, $[x,y,z,a_x,a_y,a_z]^T = \mathbf{0^T}$. That results in an optimization problem of the form

$$\min_{\mathbf{T_{ext}}} \sum_i \rho(\|r_i\|^2). \tag{6.3}$$

where ρ represents a Tukey loss function to reject gross outliers.

Since a classic service robot moves only in a 3D space (x,y,θ), but the camera pose has to be determined in 6D, additional assumptions have to introduced to overcome the ambiguity of the solution. Similar to our prior work presented in [39], the method detects the ground plane to estimate the height of the camera and fixes it during the optimization.

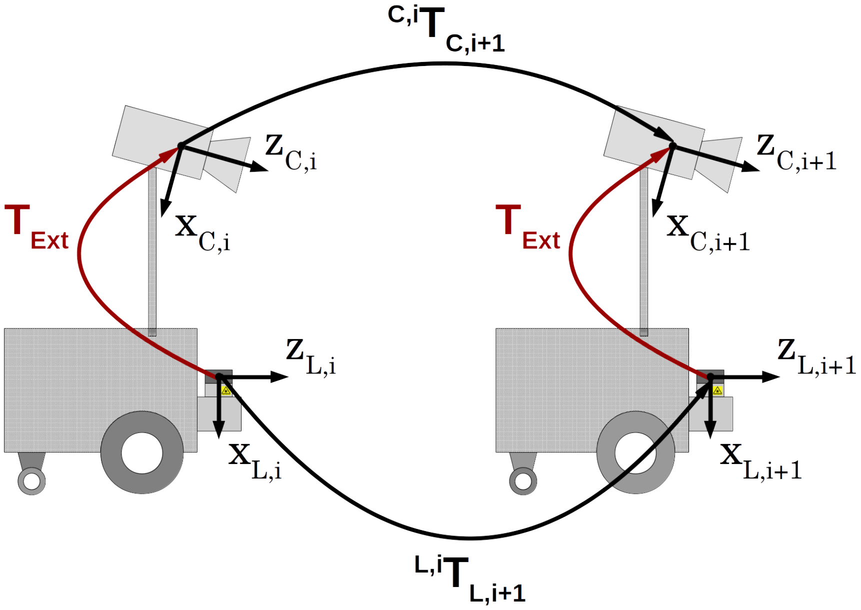


Figure 6.2: Transformations for camera pose calibration.

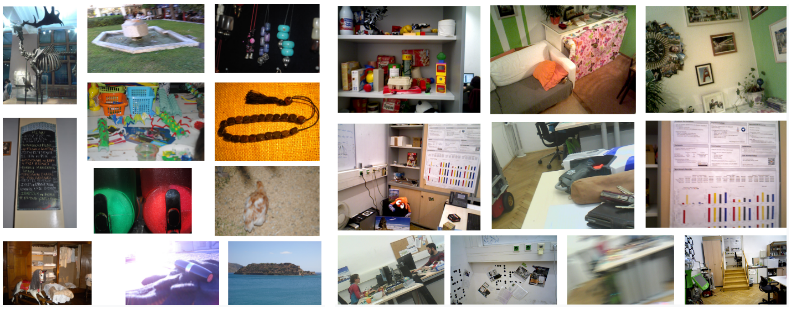
6.1.4 Motion Blur Detection

To filter out motion blurred images, a motion blur detection method is introduced which is based on the work of Pertuz et al. [92]. Six focus measures are extracted and used as features for a machine learning based motion blur detection: modified Laplacian (LAPM) [93], diagonal Laplacian (LAPD) [94], variance of Laplacian (LAPV) [95],sum of wavelet coefficients (WAVS) [96], variance of wavelet coefficients (WAVV) [96],ratio of wavelet coefficients (WAVR) [97]. An additional feature is added which is a combination of [93] and [95], variance of modified Laplacian (LAPMV), to end up with seven features for the classification problem.

Three classic machine learning algorithms are tested and benchmarked together with the introduced features. KNN ($k = 3$, $k = 5$), a Random Forrest (RF), and an easy to implement Decision Stump. All algorithms are trained on the CERTH motion blur dataset [98] and on a newly created dataset (v4rMB). While the CERTH dataset covers a variety of images, including high quality topographies and outdoor scenes, the v4rMB dataset is focused on indoor scenes recorded with a low quality color camera (cf. Figure 6.3).

6.2 Evaluation

This section presents the first results of the motion blur detection algorithm and the extrinsic pose calibration method.



(a) CERTH. (b) v4rMB.

Figure 6.3: Two different datasets that are used to train the motion blur detection algorithm.

6.2.1 Datasets

Three different datasets are created to evaluate the method. A synthetic dataset to validate the implementation, a real-life dataset recorded with a robot, and an evaluation dataset to verify that the rigid transformation is correctly estimated.

Synthetic Dataset

First, a synthetic sine-shaped trajectory is generated. The ground truth extrinsic transformation $\mathbf{T}_{\mathbf{ext},\mathbf{GT}}$ is used to express the sine-shaped trajectory in an other coordinate system, resulting in a second trajectory and completing the synthetic dataset.

Real-Life Dataset

Second, real-life sequences are recorded with an experimental robot, equipped with a laser ranger, and an RGB-D camera. The MCL is periodically triggered, while the RGB-D information and the laser scans are stored together with the most recent pose update.

Evaluation Dataset

Third, an evaluation dataset is collected that contains a corner of a room in the camera's and laser's field of view. This dataset is used to evaluate the alignment between the laser and the RGB-D data after applying the estimated rigid transformation.

6.2.2 Implementation and Evaluation Framework

The implementation makes use of several libraries and frameworks.

The optimization is implemented using the Ceres optimization library while the PCL [84] is used to detect the ground plane to estimate the height of the camera.

Several ROS [70] packages are used for recording the input data on a robot system and for the evaluation of the results: The input dataset is recorded using rosbag and the built-in message synchronization method to extract synchronized pairs of camera

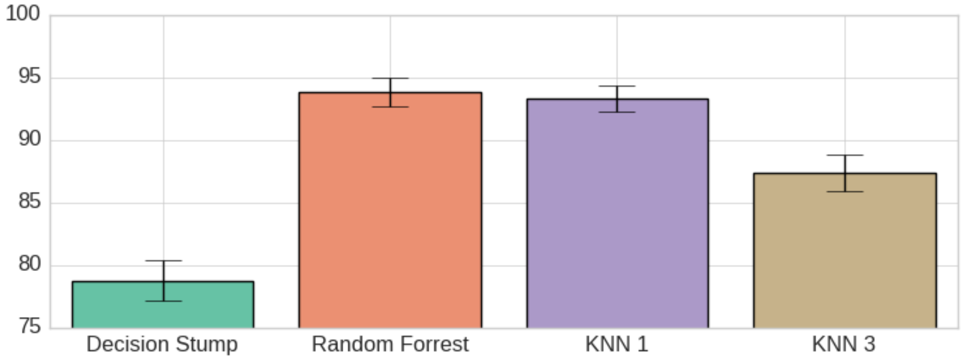


Figure 6.4: Accuracy of algorithms trained on all data.

images and pose estimations. The robot uses the amcl package to localize the robot in order to extract its current pose based on the laser scan. The calibrated rigid transformation is published into the ROS framework as TF and picked up by the point_cloud_to_laser_scan package to extract a virtual laser scan from the depth cloud.

The extracted virtual laser scan is compared with the line laser scanner using CloudCompare[1].

Weka [99] is used to evaluate the three different machine learning algorithms and their suitability as motion blur detection algorithm.

6.2.3 Result

This section wraps up the first results towards a holistic autonomous calibration method.

Motion Blur Detection

The three machine learning algorithms are trained on the v4rMB dataset and on a combined dataset that consists of both, the CERTH dataset and the v4rMB motion blur dataset. In order to get impartial results for the different classifiers, the selected algorithms are evaluated using a ten-fold cross validation. The results of the different algorithms are compared using a paired t-test to check if one of the algorithms significantly performs better than the others.

Figure 6.4 shows the results of the four different classifiers on the combined dataset. All classic machine learning approaches achieved good results, but the Random Forrest and KNN 1 clearly outperform the other two methods.

The experiments are repeated solely on the v4rMB dataset which is tailored to indoor scenes captured with a VGA resolution color camera. Figure 6.5 shows that the decision stump performs on the indoor scenes as good as the random forest, and achieves an accuracy of 92%.

The decision stump is selected as motion blur detection method, since the target platform will face indoor scenes and due to its simplicity and efficient implementation.

---

[1] CloudCompare (version 2.9) [GPL software]. (2017). Retrieved from http://www.cloudcompare.org/

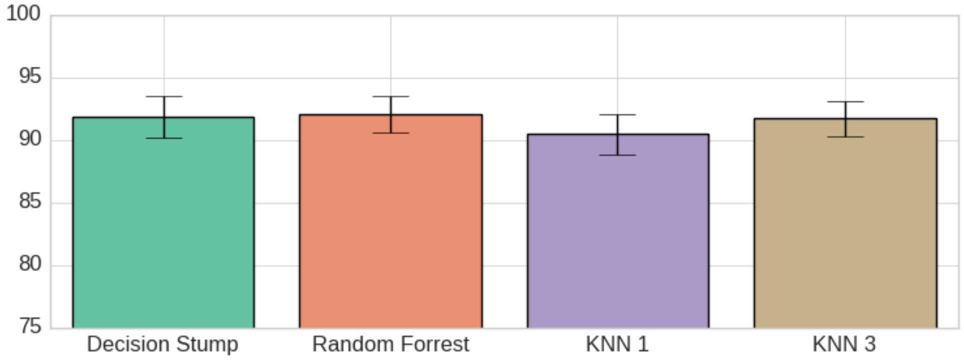


Figure 6.5: Accuracy of algorithms trained on v4rMB scenes.

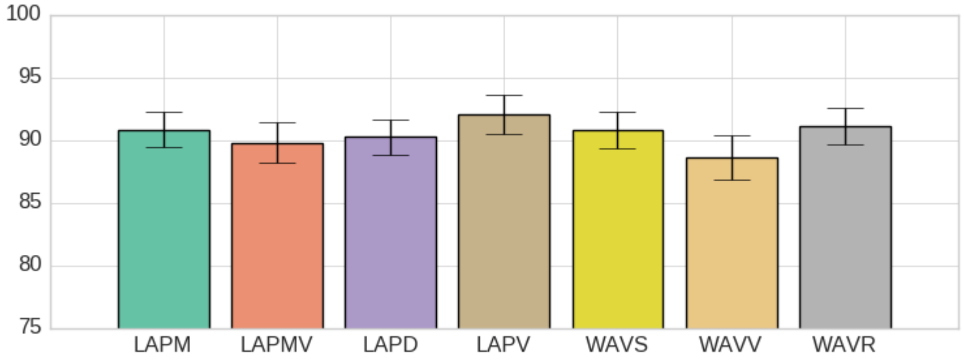


Figure 6.6: Decision stump on different features.

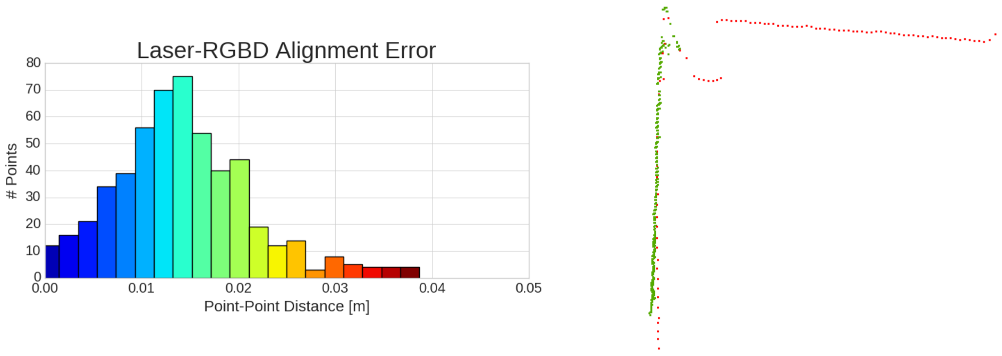
The experiments are repeated with the decision stump for every introduced feature. Figure 6.6 shows that the decision stump performs best together with LAPV as feature. Hence, the decision stump is implemented and used together with LAPV to filter the input data.

Extrinsic Pose Calibration

The synthetic dataset is used to verify the correctness of the implementation and to verify the implementation of the introduced method. The first tests revealed that the convergence behavior strongly depends on the initialization of the scale s which has to be sufficiently close to the real value. Similar to Chapter 5 the SfM map is used together with depth measurements to estimate the scale.

The first results on the real-life dataset show that the method succeeds in extracting the correct rigid transformation between the laser and the depth camera. The histogram of the cloud-cloud distance between the laser scan and the projected depth data shows an overall low alignment error of approximately 0.01m (cf. Figure 6.7a). Figure 6.7b shows that the laser scan and the projected depth information are precisely aligned, indicating a correct rigid transformation between the two sensors.

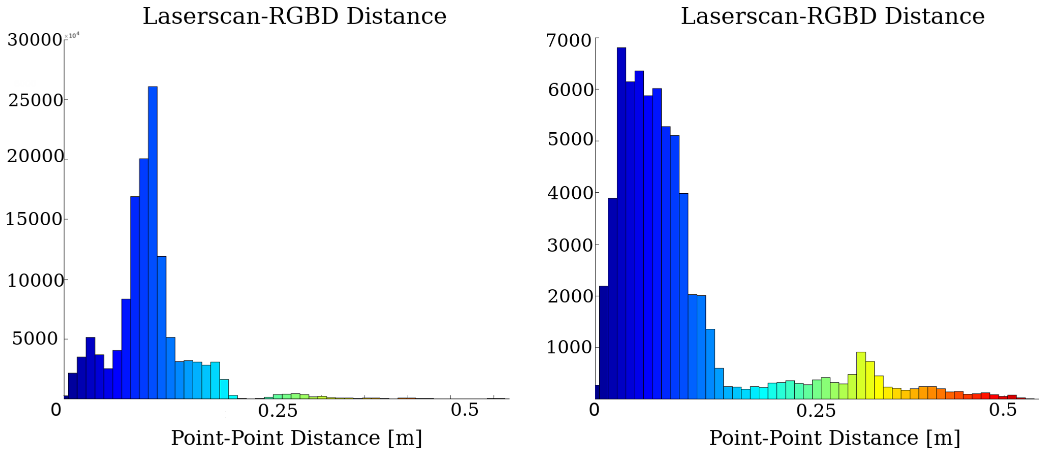
In order to evaluate the impact of the motion blur detection on the calibration,



(a) Histogram. (b) Scans.

Figure 6.7: (a) Histogram of the distance between the RGB-D depth sensor estimations and the laser-scan. (b) A pseudo-laser scan is extracted from the point cloud which is compared with the laser scan data of the robot. The two scans are well aligned and indicate a correct rigid transformation between the two systems.

the experiments are run with and without enabling the input data filtering. Figure 6.8 shows that our algorithm performs significantly better with the enabled motion blur detection. The overall alignment error that indicates the correctness of the rigid transformation decreases if the filtering is enabled.



(a) Without motion blur detection. (b) With motion blur detection

Figure 6.8: Results for the distance between the RGB-D pseudo laser scan and the real laser scan.

6.3 Discussion

This chapter presents first results towards a holistic auto calibration method for RGB-D cameras that are mounted on a robot. It is tailored to extend the method presented in

Chapter 5 and similarly uses an SfM method to extract the camera pose and a sparse map of the environment. Two trajectories and the ground plane are used to optimize a rigid transformation between the camera trajectory and the reference trajectory, while the sparse map is used together with the depth sensor to correctly initialize the scale for the optimization problem. Additionally, the approach introduces a simple and easy to implement motion blur detection algorithm that uses a single feature together with a decision stump to filter out blurred images. First results highlight the potential of this method to find a correct rigid transformation and show that the motion blur detection improves the optimization results.

The introduced method is a first step towards a holistic autonomous calibration of an RGB-D sensor on a robot system.

Chapter 7

Conclusion

This chapter finally presents a summary of the individual sections and gives a short outlook on future work.

7.1 Summary

Chapter 2 evaluates ten different depth sensors using five metrics, aiming to achieve representative and comparable results to benchmark different depth sensors in the context of robotic vision. Therefore, 510 data points are semi-automatically collected, each based on 100 depth frames. The results provide valuable information about state-of-the-art depth sensors for research in robotic perception and related applications.

Our investigation suggests the use of far-range Structured Light cameras for any application where the quality of the surface representation is more relevant than the trueness of the depth measures. This can include common robot tasks such as object modeling and recognition within the manipulation distance of a robot, i.e. distances below $2m$ for approaching and handling an object. Moreover the Asus Xtion and Structure IO sensor are able to gather data under all tested lighting conditions for all materials. Hence, they are especially useful for robots operating under uncontrolled conditions.

The ZR300 offers a low trueness for $< 4m$ but may fail to gather depth data under bright lighting conditions. Applications where the trueness is more relevant than the precision of the measurements fit the domain of this sensor.

The D435 provides a remarkable wide range from $0.2m$ to $7m$ and performs especially well for depth ranges $< 1m$. However, it failed during the experiments to gather depth measurements under bright lighting conditions and had scattered results for precision and trueness.

For large distances $> 4m$, the tested ToF sensor (KinectV2) gathers the most reliable measurements, even under bright lighting conditions.

The Ensenso Active Stereo camera offers the best trueness within its narrow range from $0.5 - 1m$. It satisfies applications requiring low biased measurements from a sensor that can be used out of the box.

The RealSense ZR300, R200, and D435 offer various parameters to adapt the sensor properties to the scene. During the experiments the factory presets for high accurate measurement have been used without any adaptation to the current lighting and material.

Chapter 3 introduces an easy to use and intuitive depth error simulation model, which makes use of a virtual stereo camera with slightly modified intrinsic parameters. It aims to simulate the characteristic error behavior of the investigated depth sensors and is targeted to be used together with synthetic datasets. The generic model is evaluated against state-of-the art depth sensors and successfully replicates systematic depth errors of various sensor technologies. It shows close-to-real-life results for the trueness and the precision of the sensor. In explicit it models non-linear, and radial-shaped, local errors that are characteristic for Structured Light and Active Stereo cameras as well as linear increasing errors which are characteristic for ToF cameras.

A feasibility study applies the generic model to a synthetic dataset and analyzes the results of a state-of-the-art reconstruction method under different depth error characteristics. The tested reconstruction algorithm showed to be sensitive to different error types which impact the surface quality as well as the trueness of the reconstruction. It also revealed that the surface quality of a reconstruction does not necessarily reflect the trueness of the results. Ideally, new algorithms should be evaluated against ground-truth data, e.g. laser scans of the scene. If ground-truth real-life data is not available, the introduced model can be used to generate close-to-real-life depth data which allows an impartial analysis of the algorithm.

Chapter 4 presents two camera-model-free methods to compensate systematic depth errors using planar targets. Since the two methods do not use a camera-model to compensate the depth error, they are agnostic to the sensor technology of the target system.

The first method parameterizes for several distances several individual lower dimensional compensation lattices. The second method optimizes a single lower dimensional compensation lattice together with an exponential function to inherently model the distance dependency of the depth error. It allows to directly interpolate compensation values for distances that were not originally observed in the input set.

Both models are evaluate on synthetic and real-life datasets. The synthetic datasets are augmented using the depth error simulation model presented in Chapter 3. The results show that both methods improve the trueness and the precision of the depth data. While the depth-discrete error model performs better on the synthetic dataset, both methods equally perform on real-life data.

A case study evaluates the depth-continuous compensation model together with a reconstruction algorithm. The algorithm reconstructs a scene using the depth data before and after the depth-continuous compensation model has been applied. The case study reveals that the model significantly improves the results of the state-of-the-art reconstruction algorithm, giving more precise and accurate reconstruction results.

Chapter 5 presents a method for the autonomous auto calibration of RGB-D sensor setups. It improves the state of the art by adding plane priors to the optimization problem which favor more stable points and have the ability to guide the solution along planes.

In total over 300 experiments are performed and a comprehensive result summary is given. Two novel quantitative evaluation methods are defined that use the cloud-cloud distance for an impartial analysis of the auto-calibration results. The results of the qualitative and the quantitative evaluation show significant improvements which are reflected in more precisely aligned RGB and depth images, as well as more precise depth measurements. I.e. the results revealed seven new residual and weighting combinations that achieved better results than the reference method.

A use-case scenario demonstrates the usability of the auto calibration method. It applies the calibration method to a dataset which is used to semi-automatically annotating the depth information of a novel RGB-D dataset for deep learning.

Chapter 6 provides first results of a camera pose calibration method which calibrates the camera pose relative to a reference coordinate system. It introduces a novel, easy to implement motion blur detection algorithm to filter the input data. The calibration method uses the 3D motion of the robot together with the RGB-D camera to estimate the rigid transformation between the navigation sensor and the depth camera.

The convergence behavior of the method is validated using a synthetic dataset. The calibration results are evaluated using real-life data record with an experimental robot platform. The robot is equipped with a state-of-the-art sensor setup that consists of a RGB-D camera and a laser scanner that is used for the pose estimation of the robot. The results show that the method accurately estimates the extrinsic camera pose. The motion blur detection algorithm further improves the results by filtering out unreliable camera- or robot poses. The introduced method can be easily combined with the autonomous calibration method presented in Chapter 5, to form a holistic autonomous calibration method for RGB-D sensors.

7.2 Outlook

This section presents a short outlook for every individual chapter.

Future work could extend the sensor study by evaluating the different sensors under outdoor lighting conditions and could add new sensors that continuously enter the market. Experiments I and II (trueness and precision) are easy to reproduce. Users with new sensors should be able to gather data in a similar way to i) benchmark their sensor against the presented results and ii) apply the introduced easy to use error model. Furthermore, the experiments regarding precision, trueness, and lateral noise could be extended for different viewing angles. The setup may be extended by replacing standard drivers with more advanced methods. E.g. [100] implements a method to calculate a disparity map for Structured Light sensors, and [101] adds a filter to the Freenect2 driver to extend the sensor range, which may be of relevance to robotics applications.

Chapter 3 shows that the simulation model can be easily combined with existing datasets to generate realistic, close-to-real-life depth data together with ground truth information. Future work may incorporate the models into a simulation environment to model close-to-real-life depth errors. The simulated data could be used to generate large scale, more realistic synthetic datasets, which are well suited for the training of data-driven machine learning methods.

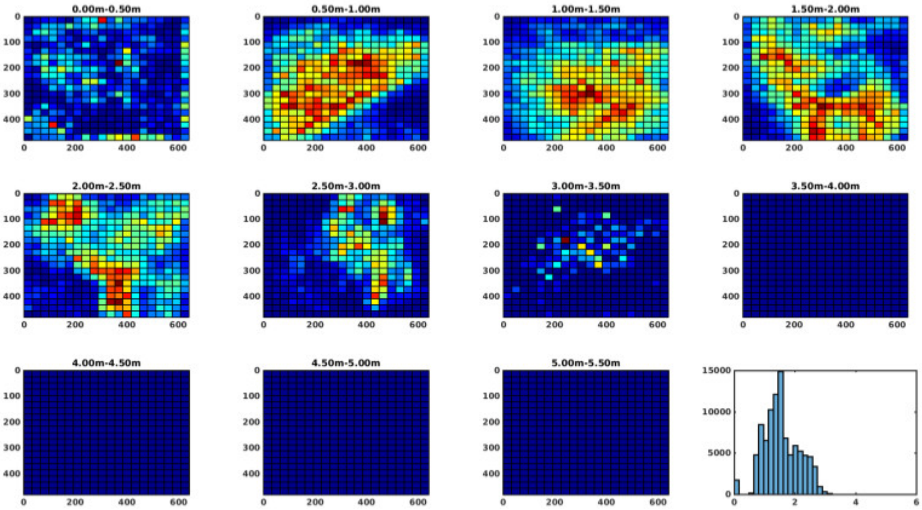


Figure 7.1: Coverage statistic for one of the used datasets for auto calibration. The color information encodes the number of SfM point re-projections for the corresponding depth pixel. The histogram gives additional information regarding the covered distances.

The offset compensation methods presented in Chapter 4 require an input dataset that contains flat surfaces which might be hard to collect. Future work could adapt the method to remove this constraints. The depth-discrete error compensation model does not inherently allow to interpolate correction values that are not contained in the input dataset. However, it might be possible to linearly interpolate between the separate models to get compensation values for depth distances that are not inherently modeled. That might give comparable or even better results than the depth-continuous model.

Although the time for the Jacobian estimation has been already drastically reduced by incorporating handcrafted residuals, the whole optimization itself takes several minutes. Since the slowest part is the linear solver, future work could analyze the approximated Hessian in order to use the most appropriate solver for the problem structure.

As next step the auto-calibration method and the quantitative evaluation methods presented in Chapter 5 can be utilized to fully autonomous (re)calibrate an RGB-D camera on a robotic system and to automatically estimate the quality of the calibration. It might be possible to replace the SfM reconstruction or add a high accurate laser scan of the environment to further improve the calibration results. Future work could investigate the inverse depth parameterization for the point-plane distance and may answer the question how the calibration could work in an online fashion, even in dynamic environments.

As a next step data awareness could be added to the algorithm. For instance by extracting a coverage statistic for the input data of the optimization problem (cf. Figure 7.1). This statistic could be used to optimize the trajectory of the robot to specifically add information that is not included in the input data. This statistic could be also used to extract a pixel-wise confidence value for the compensated depth image.

A crucial step towards a robust holistic camera calibration is to robustify and further investigate the extrinsic camera pose calibration. One way to gain robustness is to identify unreliable trajectory points in order to reduce their influence on the optimization problem. So far the introduced method makes use of a Tukey loss function to reduce the influence of outliers. However, an even more sophisticated method might be to incorporate the Mahalanobis distance [102] to incorporate the uncertainty of the camera poses. The necessary inverse covariance (information matrix) could be directly extracted from the bundle adjustment problem after resolving the gauge ambiguity. This could be realized either by fixing two camera poses to remove the gauge freedom or by incorporating the work presented in [103]. The necessary covariance propagation for the transformations could be derived using a linear error propagation. Blanco presents in [104] a tutorial on SE(3) transformation and summarizes how uncertainty can be incorporate into different transformation representations. Barfoot and Furgale [105] investigated 2014 how uncertainties can be associated with 3D poses. Both works might be a good starting point to improve the optimal hand-eye-calibration. Together with a Cholesky decomposition [106] and the Mahalanobis distance it should be possible to incorporate the uncertainty into the optimization problem, to achieve more accurate calibration results.

Acknowledgment

This work was partially supported through the Horizon 2020 Programme, grant No. 645376 FLOBOT, funding from the Austrian Science Fund, grant No. I1856-N30 ALOOF, and by the Vienna Science and Technology Fund, grant No. ICT15-045 RALLI as well as the Federal Ministry of Science and Research of Austria grant SPA 04/84 FRANC.

Parts of this work originated during an internship at DAQRI [1].

---

[1] DAQRI: `https://daqri.com/`

Appendix A

3D Reconstructions of the ICL-NUIM Dataset

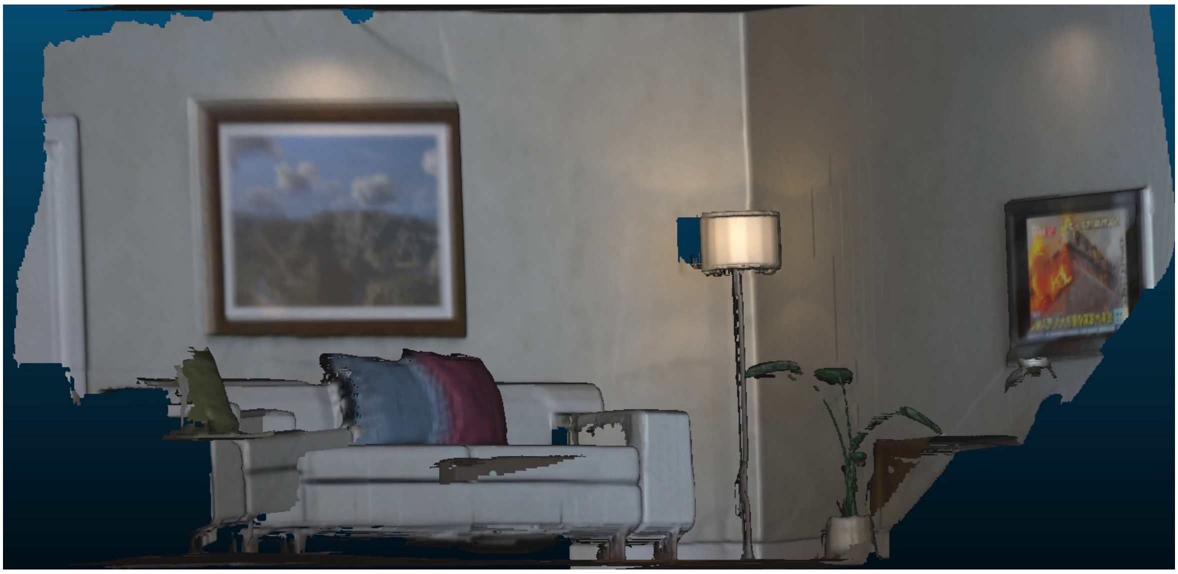


Figure A.1: Impact of the baseline error on the reconstruction.

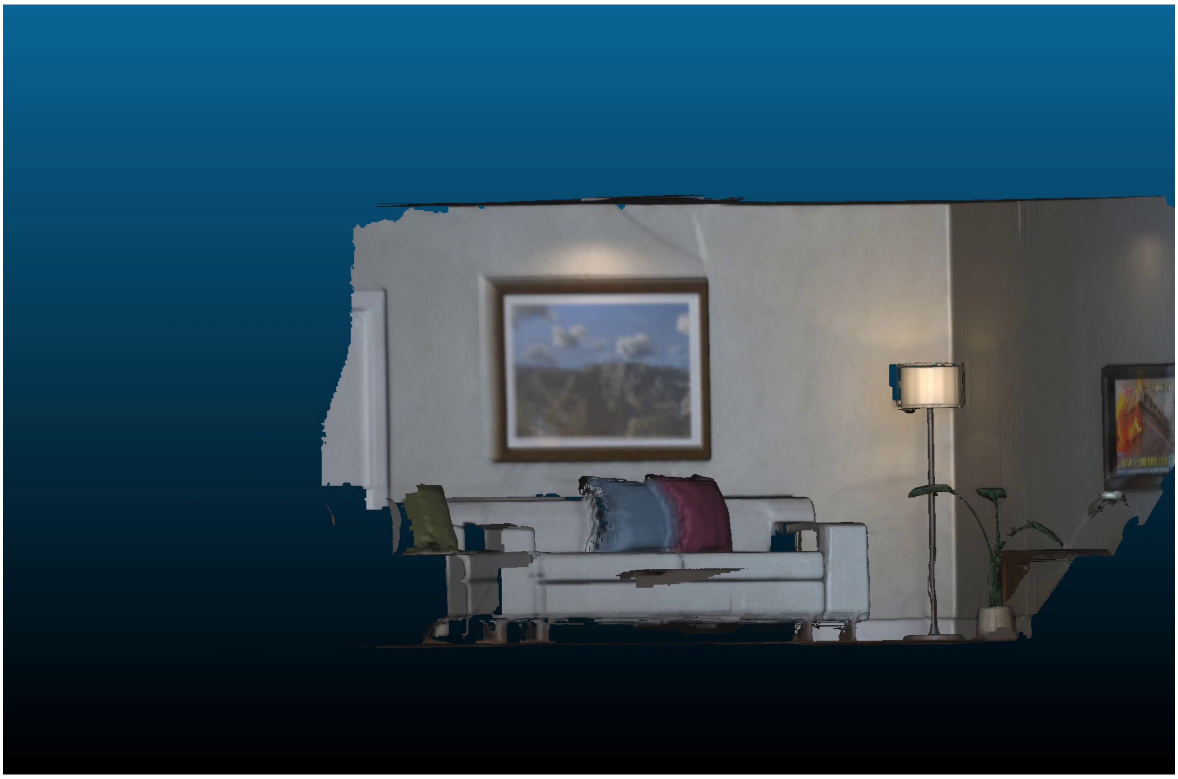


Figure A.2: Impact of the principal point error on the reconstruction.

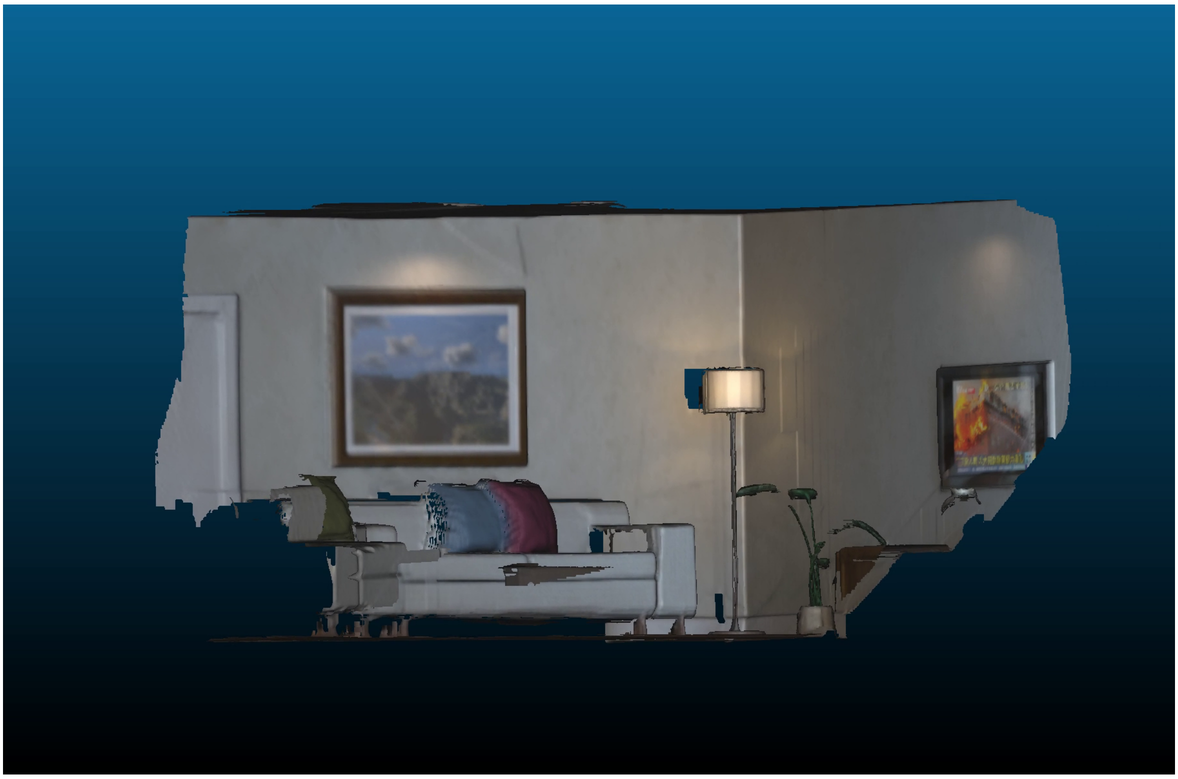


Figure A.3: Impact of the focal length error on the reconstruction.

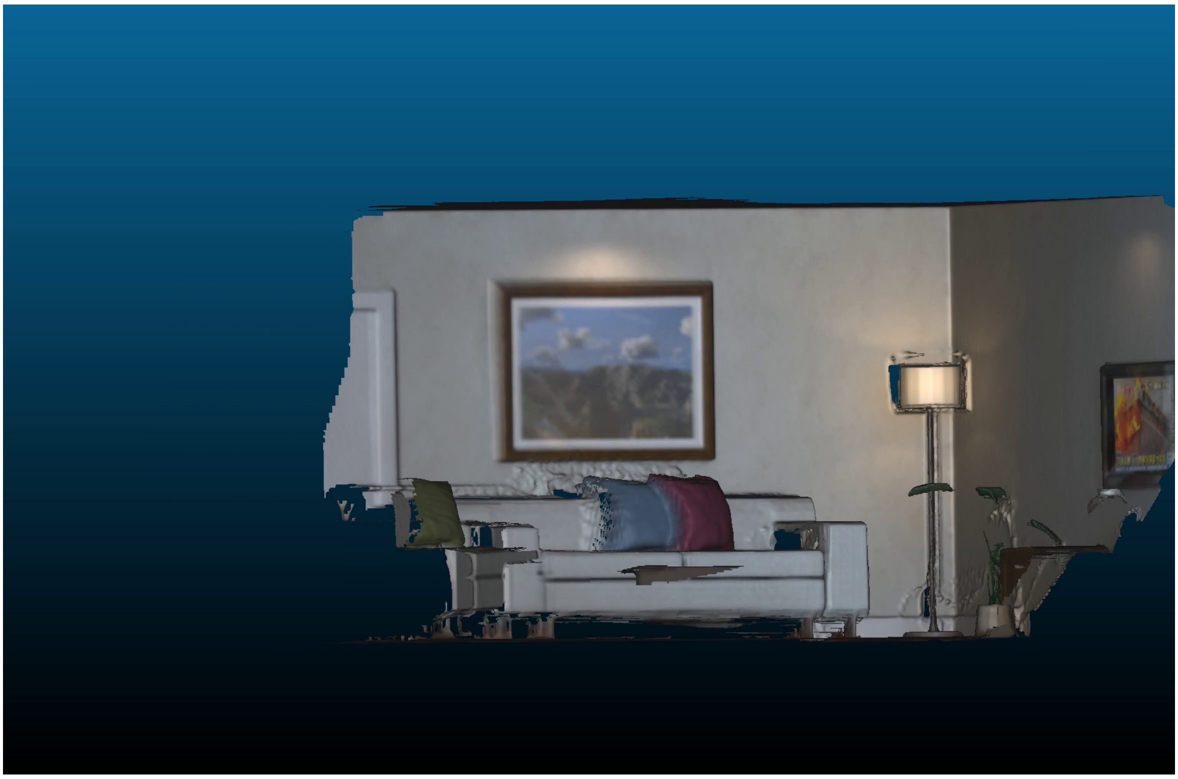


Figure A.4: Impact of the interpolation error on the reconstruction.

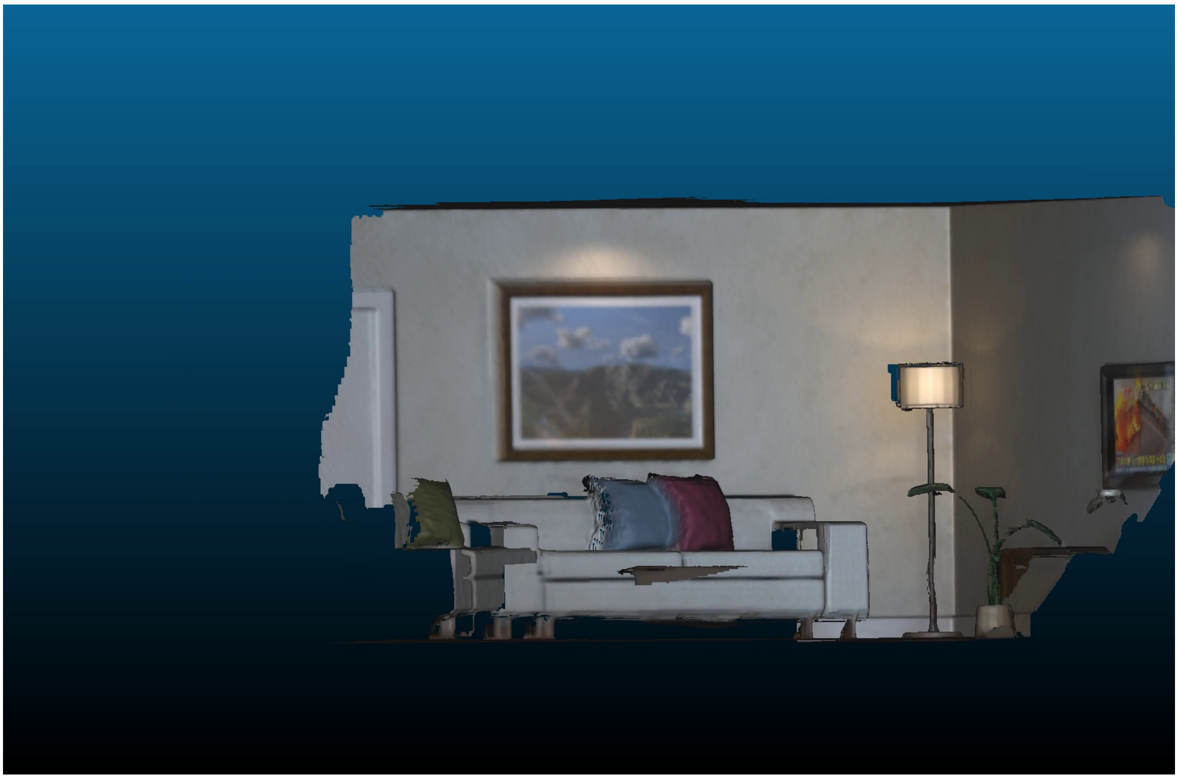


Figure A.5: Impact of the quantization error on the reconstruction.

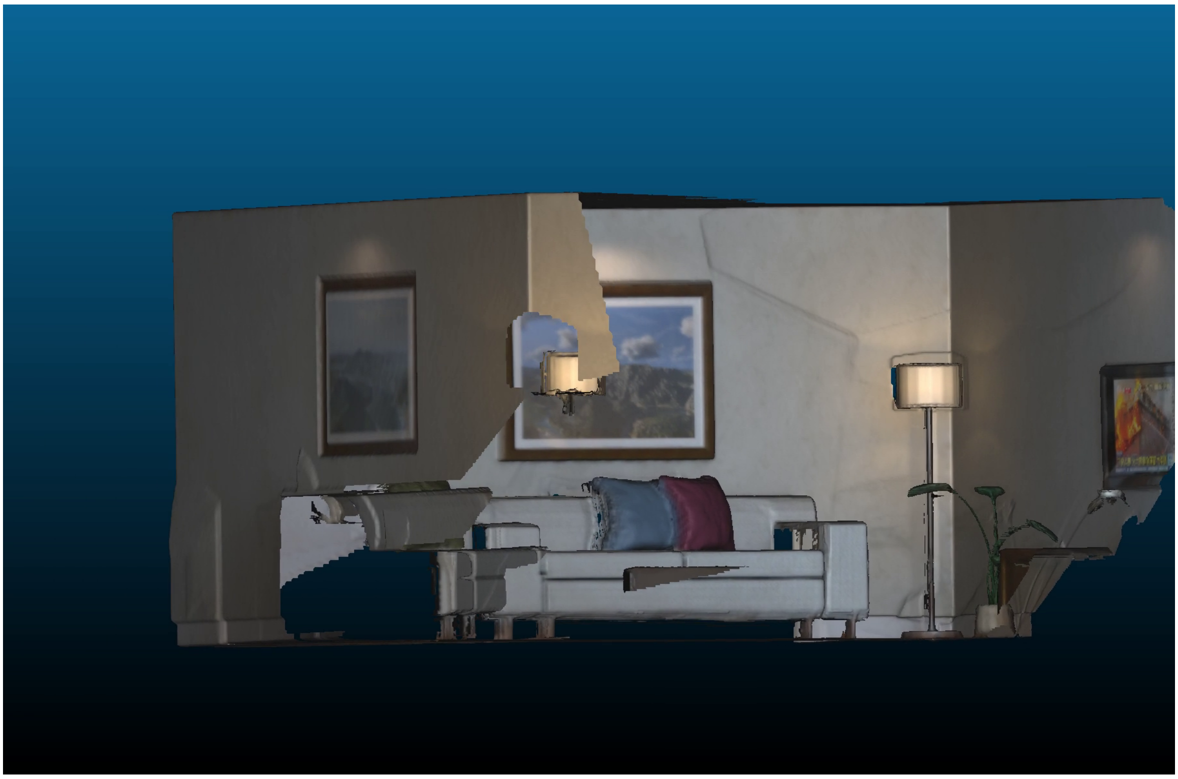


Figure A.6: Impact of the tangential distortion error on the reconstruction

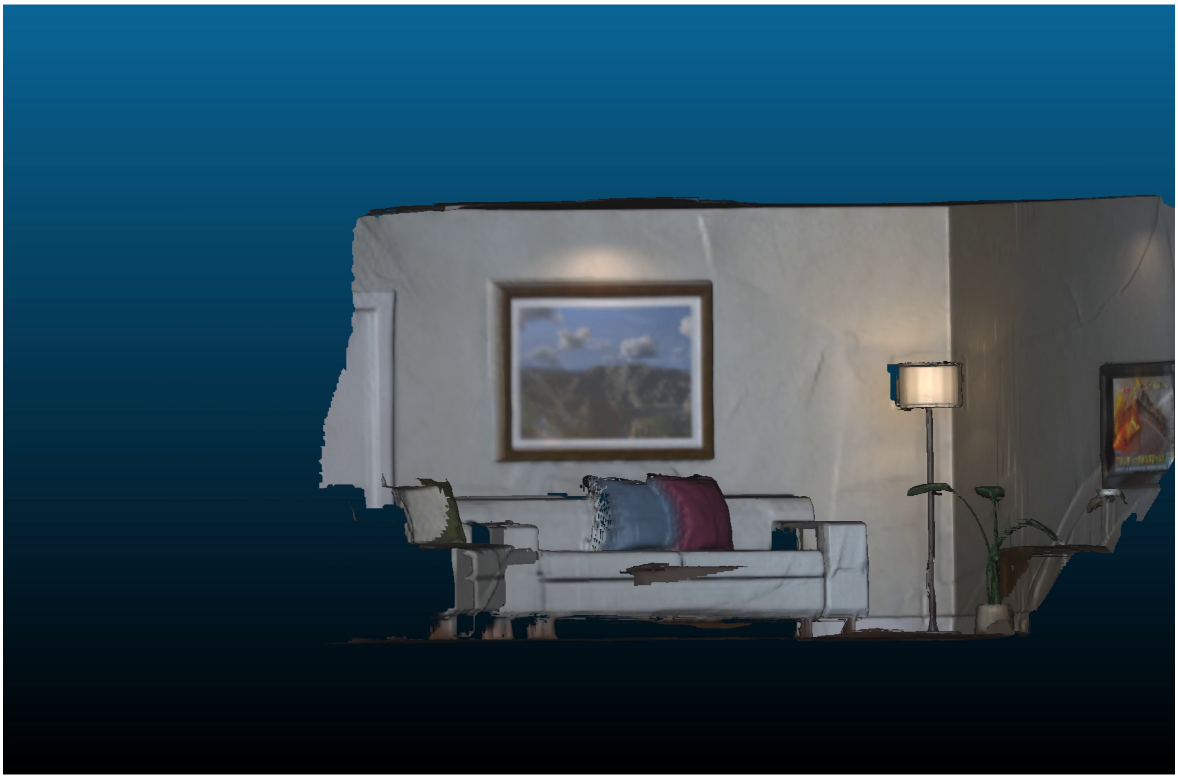


Figure A.7: Impact of the radial distortion error on the reconstruction.

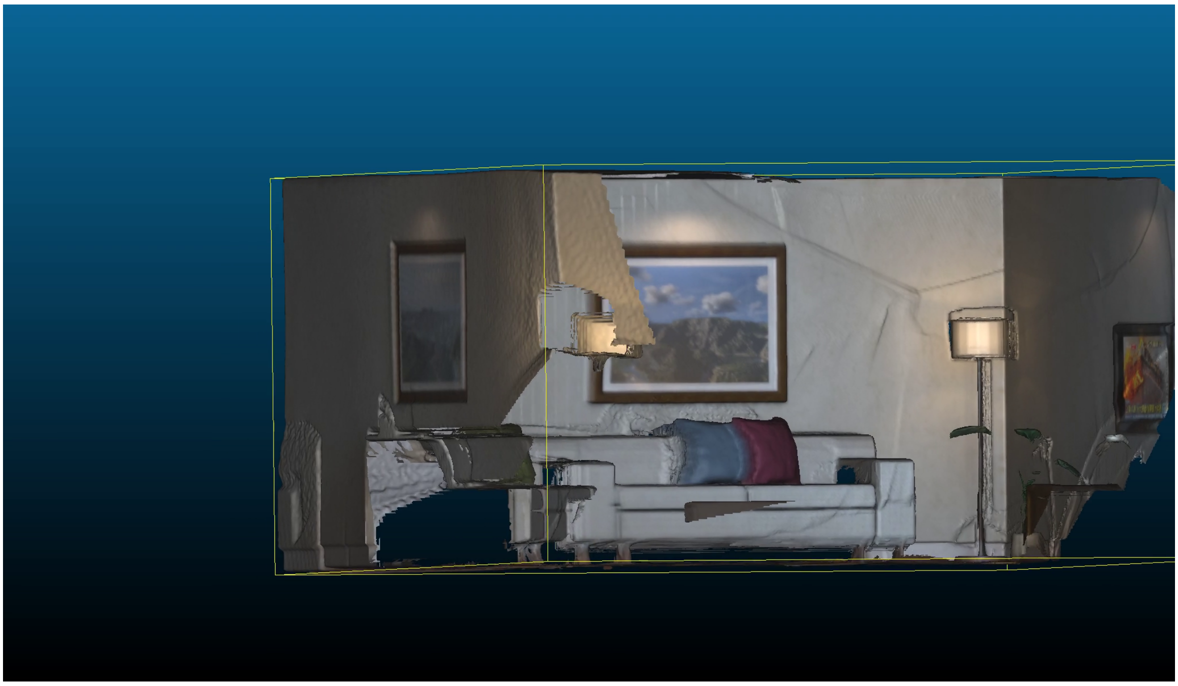


Figure A.8: Impact of the all errors in combination on the reconstruction.

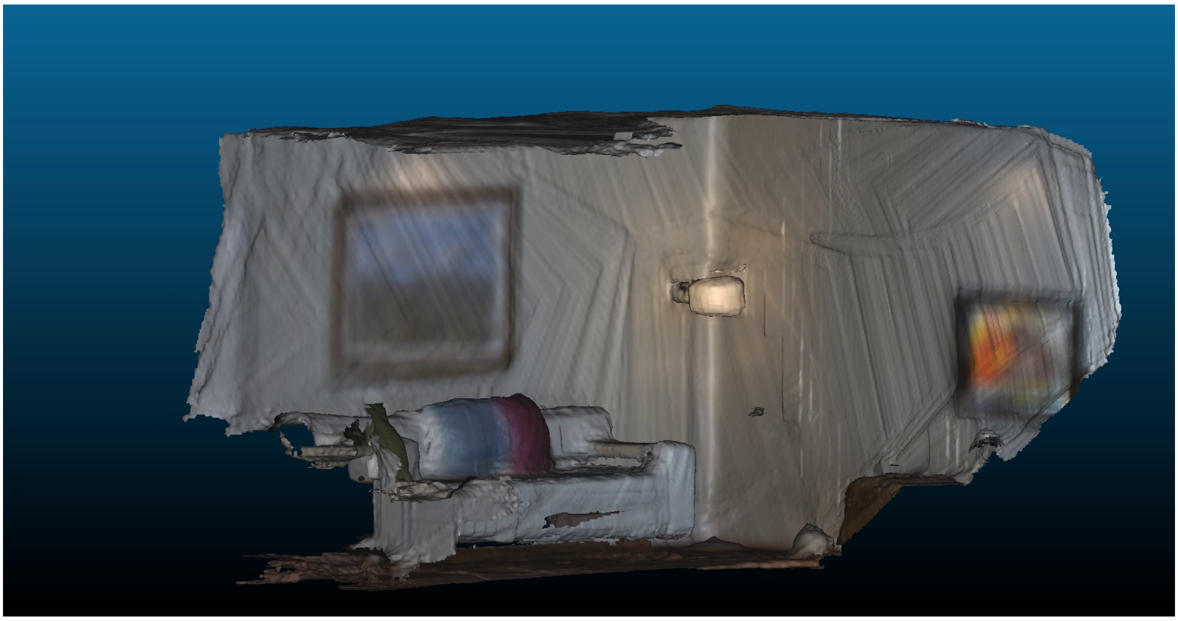


Figure A.9: Impact of the camera pose error on the reconstruction.

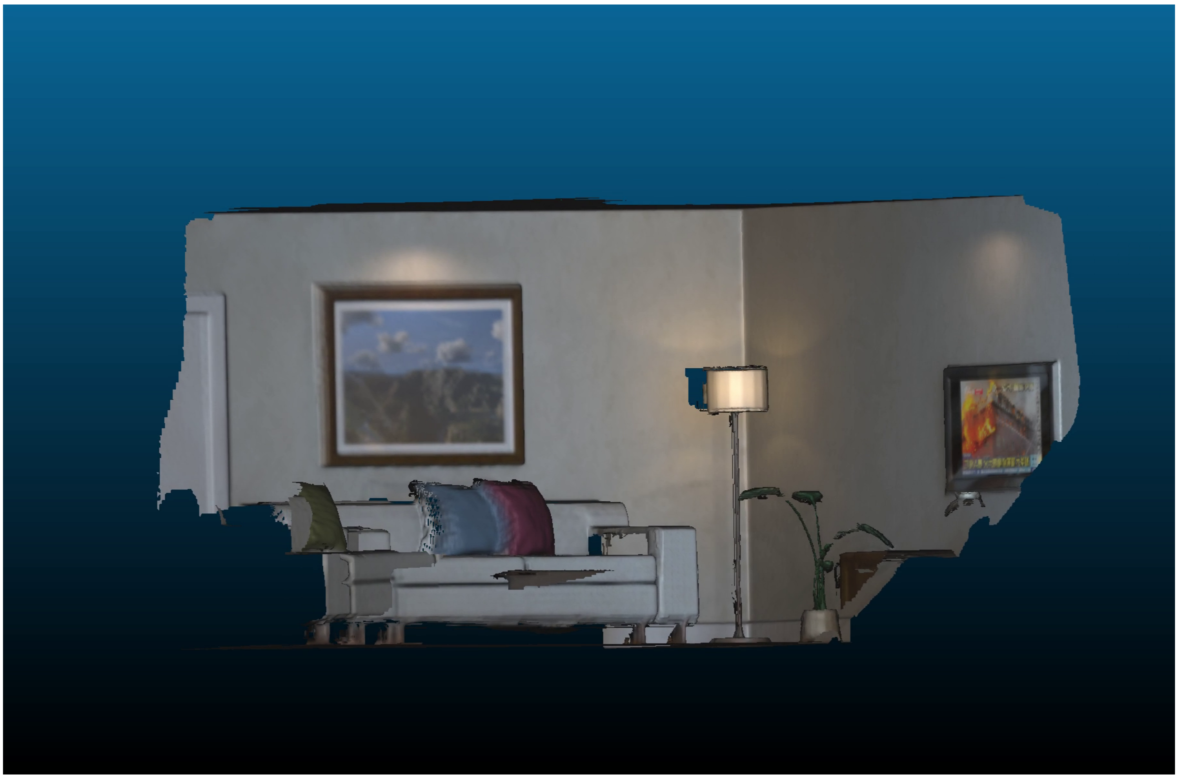


Figure A.10: Ground truth reconstruction.

Appendix B

Bicubic Interpolation

Equation (4.14) uses the constant 16×16 matrix $\mathbf{A}$,

$$\mathbf{A} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -\frac{5}{2} & 2 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{1}{2} & \frac{3}{2} & -\frac{3}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{4} & 0 & -\frac{1}{4} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{4} & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 \\
-\frac{1}{2} & \frac{5}{4} & -1 & \frac{1}{4} & 0 & 0 & 0 & 0 & \frac{1}{2} & -\frac{5}{4} & 1 & -\frac{1}{4} & 0 & 0 & 0 & 0 \\
\frac{1}{4} & -\frac{3}{4} & \frac{3}{4} & -\frac{1}{4} & 0 & 0 & 0 & 0 & -\frac{1}{4} & \frac{3}{4} & -\frac{3}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & -\frac{5}{2} & 0 & 0 & 0 & 2 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 \\
-\frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{5}{4} & 0 & -\frac{5}{4} & 0 & -1 & 0 & 1 & 0 & \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\
1 & -\frac{5}{2} & 2 & -\frac{1}{2} & -\frac{5}{2} & \frac{25}{4} & -5 & \frac{5}{4} & 2 & -5 & 4 & -1 & -\frac{1}{2} & \frac{5}{4} & -1 & \frac{1}{4} \\
-\frac{1}{2} & \frac{3}{2} & -\frac{3}{2} & \frac{1}{2} & \frac{5}{4} & -\frac{15}{4} & \frac{15}{4} & -\frac{5}{4} & -1 & 3 & -3 & 1 & \frac{1}{4} & -\frac{3}{4} & \frac{3}{4} & -\frac{1}{4} \\
0 & -\frac{1}{2} & 0 & 0 & 0 & \frac{3}{2} & 0 & 0 & 0 & -\frac{3}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 \\
\frac{1}{4} & 0 & -\frac{1}{4} & 0 & -\frac{3}{4} & 0 & \frac{3}{4} & 0 & \frac{3}{4} & 0 & -\frac{3}{4} & 0 & -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\
-\frac{1}{2} & \frac{5}{4} & -1 & \frac{1}{4} & \frac{3}{2} & -\frac{15}{4} & 3 & -\frac{3}{4} & -\frac{3}{2} & \frac{15}{4} & -3 & \frac{3}{4} & \frac{1}{2} & -\frac{5}{4} & 1 & -\frac{1}{4} \\
\frac{1}{4} & -\frac{3}{4} & \frac{3}{4} & -\frac{1}{4} & -\frac{3}{4} & \frac{9}{4} & -\frac{9}{4} & \frac{3}{4} & \frac{3}{4} & -\frac{9}{4} & \frac{9}{4} & -\frac{3}{4} & -\frac{1}{4} & \frac{3}{4} & -\frac{3}{4} & \frac{1}{4}
\end{bmatrix}, \tag{B.1}$$

Appendix C

Autonomous Calibration of RGB-D Setups

Table C.1: Qualitative and quantitative results of the presented calibration method and combinations outperforming the state of the art. Example: Method that gives the overall best result (see column "$\sum$"): PLD enabled. Method that gives the best qualitative result (see column "man"): PLD, IPD, and WIPD enabled.

Experiments							Results			
#	PLD	IPD	WPLD	WIPD	v4	rB	QMDepth	QMSfM	man	$\sum$
1	1	0	0	0	0	0	**211**	**198**	**228**	***637***
2	0	1	0	0	1	1	***214***	**217**	157	**588**
3	1	0	1	0	0	1	**197**	**179**	**189**	**565**
4	0	1	0	1	0	0	**197**	144	**221**	**562**
5	1	1	1	0	0	1	**194**	***218***	149	**561**
6	1	1	0	1	0	0	167	151	***240***	**558**
7	0	1	0	0	0	1	189	**209**	150	548
*8	0	1	0	0	0	0	172	**185**	**190**	547
9	1	0	1	0	0	0	**199**	167	176	542
10	1	0	0	0	1	0	187	158	**183**	528
11	1	1	1	1	0	0	186	147	169	502
12	1	1	1	0	1	1	170	210	110	490
13	1	1	0	0	1	1	155	174	145	474
14	1	1	1	0	1	0	141	175	149	465
15	0	1	0	1	1	1	152	165	135	452
16	1	1	0	0	1	0	122	161	164	447
17	1	1	1	1	0	1	126	165	148	439
18	1	0	1	0	1	1	147	138	153	438
19	0	1	0	0	1	0	127	147	157	431
20	1	1	0	0	0	0	118	151	155	424
21	1	1	1	0	0	0	130	157	122	409
22	0	1	0	1	0	1	116	138	128	382
23	0	1	0	1	1	0	161	75	132	368
24	1	1	0	0	0	1	122	140	104	366
25	1	0	1	0	1	0	124	98	123	345
26	1	1	1	1	1	1	96	118	127	341
27	1	1	0	1	1	0	119	71	143	333
28	1	1	1	1	1	0	119	85	112	316
29	1	1	0	1	1	1	73	91	67	231
30	1	0	0	0	1	1	71	73	70	214
31	1	1	0	1	0	1	31	34	32	97
32	1	0	0	0	0	1	2	5	26	33

* Reference method introduce in [32]

Bibliography

[1] I. Asimov and J. M. Côté, *Futuredays: A nineteenth-century vision of the year 2000*. H. Holt, 1986.

[2] K. Čapek, *R.U.R. - Rossumovi Univerzální Roboti*. 1920.

[3] G. Halmetschlager, J. Prankl, and M. Vincze, "Towards Agricultural Robotics for Organic Farming," *Proceedings of the OAGM and ARW Joint Workshop on Computer Vision and Robotics*, 117ff, 2016.

[4] A. Grünauer, G. Halmetschlager-Funek, J. Prankl, and M. Vincze, "The power of GMMs: unsupervised dirt spot detection for industrial floor cleaning robots," in *Annual Conference Towards Autonomous Robotic Systems*, Springer, 2017, pp. 436–449.

[5] G. Halmetschlager-Funek, J. Prankl, and M. Vincze, "Towards Autonomous Auto Calibration of Unregistered RGB-D Setups: The Benefit of Plane Priors," in *In Proceedings of 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 5547–5554.

[6] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for human classification in 3D LiDAR-based tracking," in *In Proceedings of 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, 2017, pp. 864–871.

[7] D. Fischinger, P. Einramhof, K. Papoutsakis, W. Wohlkinger, P. Mayer, P. Panek, S. Hofmann, T. Koertner, A. Weiss, A. Argyros, *et al.*, "Hobbit, a care robot supporting independent living at home: First prototype and lessons learned," *Robotics and Autonomous Systems*, vol. 75, pp. 60–78, 2016.

[8] C. V. Nguyen, S. Izadi, and D. Lovell, "Modeling kinect sensor noise for improved 3d reconstruction and tracking," in *Second International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, IEEE, 2012, pp. 524–530.

[9] M. R. Loghmani, B. Caputo, and M. Vincze, "Recognizing Objects In-the-wild: Where Do We Stand?" In *In Proceedings of 2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2170 –2177.

[10] T. Fäulhammer, R. Ambruş, C. Burbridge, M. Zillich, J. Folkesson, N. Hawes, P. Jensfelt, and M. Vincze, "Autonomous learning of object models on a mobile robot," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 26–33, 2017.

[11] K. Tateno, F. Tombari, and N. Navab, “Real-time and scalable incremental segmentation on dense SLAM,” in *In Proceedings of 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2015, pp. 4465–4472.

[12] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, “Real-time 3D reconstruction at scale using voxel hashing,” *ACM Transactions on Graphics (TOG)*, vol. 32, no. 6, p. 169, 2013.

[13] Q.-Y. Zhou and V. Koltun, “Depth camera tracking with contour cues,” in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 632–638.

[14] ASUS. (2017). Specifications Xtion Pro Live. retrieved 2018-06-28, [Online]. Available: `https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/specifications/`.

[15] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli, “Depth mapping using projected patterns,” pat., US Patent 8,150,142, 2012.

[16] D. Eigen, C. Puhrsch, and R. Fergus, “Depth map prediction from a single image using a multi-scale deep network,” in *Advances in neural information processing systems*, 2014, pp. 2366–2374.

[17] F. Liu, C. Shen, and G. Lin, “Deep convolutional neural fields for depth estimation from a single image,” in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5162–5170.

[18] E. Brachmann and C. Rother, “Learning less is more-6d camera localization via 3d surface regression,” in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4654–4662.

[19] K. Gwn, K. Reddy, M. Giering, and E. A. Bernal, “Generative adversarial networks for depth map estimation from RGB video,” in *In Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).*, IEEE, 2018, pp. 1258–12 588.

[20] Structure. (2017). Specifications Structure IO. retrieved 2018-06-28, [Online]. Available: `https://structure.io/embedded`.

[21] Orbbec. (2017). Specifications Orbbec Astra Pro. retrieved 2018-06-28, [Online]. Available: `https://orbbec3d.com/product-astra-pro/`.

[22] Microsoft. (2017). Specifications Microsoft Kinect. retrieved 2018-06-28, [Online]. Available: `https://developer.microsoft.com/en-us/windows/kinect/hardware`.

[23] Intel Corporation. (2018). Intel RealSense Camera D435. retrieved 2018-06-28, [Online]. Available: `https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf`.

[24] ——, (2017). Intel RealSense Camera ZR300. retrieved 2018-06-28, [Online]. Available: `https://click.intel.com/media/ZR300-Product-Datasheet-Public-002.pdf`.

[25] ——, (2017). Intel RealSense Camera R200. retrieved 2018-06-28, [Online]. Available: `https://software.intel.com/sites/default/files/managed/d7/a9/realsense-camera-r200-product-datasheet.pdf`.

[26] ——, (2017). Intel RealSense Camera F200. retrieved 2018-06-28, [Online]. Available: `https://communities.intel.com/docs/DOC-24012`.

[27] ——, (2017). Intel RealSense Camera SR300. retrieved 2018-06-28, [Online]. Available: `https://software.intel.com/sites/default/files/managed/0c/ec/realsense-sr300-product-datasheet-rev-1-0.pdf`.

[28] Ensenso. (2017). Specifications N35 Series. retrieved 2018-06-28, [Online]. Available: `https://www.ensenso.com/support/modellisting/?id=N35-804-16-IR`.

[29] ISO and DIN, "5725-1: 1994," *Accuracy (trueness and precision) of measurement methods and results–part*, vol. 1,

[30] N. Sünderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford, and P. Corke, "The limits and potentials of deep learning for robotics," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 405–420, 2018.

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *In Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[32] B. Zeisl and M. Pollefeys, "Structure-based auto-calibration of RGB-D sensors," in *In Proceedings of 2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016, pp. 5076–5083.

[33] A. Teichman, S. Miller, and S. Thrun, "Unsupervised Intrinsic Calibration of Depth Sensors via SLAM," in *In Proceedings of Robotics: Science and Systems*, vol. 248, 2013, p. 3.

[34] J. Quenzel, R. A. Rosu, S. Houben, and S. Behnke, "Online depth calibration for RGB-D cameras using visual SLAM," in *In Proceedings of 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2227–2234.

[35] BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, and OIML, "International vocabulary of metrology–basic and general concepts and associated terms, 2008," *JCGM*, vol. 200, pp. 99–12, 2008.

[36] B. N. Taylor and C. E. Kuyatt, *Guidelines for evaluating and expressing the uncertainty of NIST measurement results.* US Department of Commerce, Technology Administration, National Institute of Standards and Technology Gaithersburg, MD, 1994.

[37] G. Halmetschlager-Funek, M. Suchi, M. Kampel, and M. Vincze, "An Empirical Evaluation of Ten Depth Cameras: Bias, Precision, Lateral Noise, Different Lighting Conditions and Materials, and Multiple Sensor Setups in Indoor Environments," *IEEE Robotics Automation Magazine*, pp. 1–1, 2018, ISSN: 1070-9932.

[38] G. Halmetschlager, J. Prankl, and M. Vincze, “Probabilistic near infrared and depth based crop line identification,” in *In Workshop Proceedings of IAS-13*, 2014, pp. 474–482.

[39] F. Malekghasemi, G. Halmetschlager-Funek, and M. Vincze, “Autonomous Extrinsic Calibration of a Depth Sensing Camera on Mobile Robots,” *Proceedings of the Austrian Robotics Workshop*, pp. 29–35, 2018.

[40] J. Han, L. Shao, D. Xu, and J. Shotton, “Enhanced computer vision with microsoft kinect sensor: A review,” *IEEE transactions on cybernetics*, vol. 43, no. 5, 1318–onfere1334, 2013.

[41] M. R. Andersen, T. Jensen, P. Lisouski, A. K. Mortensen, M. K. Hansen, T. Gregersen, and P. Ahrendt, “Kinect depth sensor evaluation for computer vision applications,” *Electrical and Computer Engineering Technical Report ECE-TR-6*, vol. 1, no. 6, 2012.

[42] J. Smisek, M. Jancosek, and T. Pajdla, “3D with Kinect,” in *Consumer depth cameras for computer vision*, Springer, 2013, pp. 3–25.

[43] C. Pramerdorfer, *Depth data analysis for fall detection*, TUW, Master Thesis, 2013.

[44] K. Berger, K. Ruhl, C. Brümmer, Y. Schröder, A. Scholz, and M. Magnor, “Markerless Motion Capture using multiple Color-Depth Sensors,” in *In Proceedings of Vision, Modeling and Visualization (VMV)*, Eurographics, 2011, pp. 317–324.

[45] S. Foix, G. Alenya, and C. Torras, “Lock-in time-of-flight (ToF) cameras: A survey,” *IEEE Sensors Journal*, vol. 11, no. 9, pp. 1917–1926, 2011.

[46] B. Langmann, K. Hartmann, and O. Loffeld, “Depth Camera Technology Comparison and Performance Evaluation.,” in *ICPRAM (2)*, Citeseer, 2012, pp. 438–444.

[47] H. Yamazoe, H. Habe, I. Mitsugami, and Y. Yagi, “Depth error correction for projector-camera based consumer depth cameras,” *Computational Visual Media*, vol. 4, no. 2, pp. 103–111, 2018.

[48] C. Mei and P. Rives, “Calibration between a central catadioptric camera and a laser range finder for robotic applications,” in *In Proceedings of 2006 IEEE International Conference on Robotics and Automation (ICRA)*, 2006, pp. 532–537.

[49] D. Scaramuzza, A. Harati, and R. Siegwart, “Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes,” in *In Proceedings of 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 4164–4169.

[50] Y. M. Kim, D. Chan, C. Theobalt, and S. Thrun, “Design and calibration of a multi-view TOF sensor fusion system,” in *In Proceedings of 2008 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).*, 2008, pp. 1–7.

[51] F. Basso, A. Pretto, and E. Menegatti, "Unsupervised intrinsic and extrinsic calibration of a camera-depth sensor couple," *In Proceedings of 2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6244–6249, 2014.

[52] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *In Proceedings of Third International Conference on 3-D Digital Imaging and Modeling.*, IEEE, 2001, pp. 145–152.

[53] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and vision computing*, vol. 10, no. 3, pp. 145–155, 1992.

[54] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," in *Sensor Fusion IV: Control Paradigms and Data Structures*, International Society for Optics and Photonics, vol. 1611, 1992, pp. 586–607.

[55] S. N. Sinha, M. Pollefeys, and L. McMillan, "Camera network calibration from dynamic silhouettes," in *null*, IEEE, 2004, pp. 195–202.

[56] G. Carrera, A. Angeli, and A. J. Davison, "SLAM-based automatic extrinsic calibration of a multi-camera rig," in *In Proceedings of 2011 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2011, pp. 2652–2659.

[57] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustmentâ€"a modern synthesis," in *International workshop on vision algorithms*, Springer, 1999, pp. 298–372.

[58] S. Miller, A. Teichman, and S. Thrun, "Unsupervised extrinsic calibration of depth sensors in dynamic scenes," in *In Proceedings of 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2013, pp. 2695–2702.

[59] S. Li, P. N. Pathirana, and T. Caelli, "Multi-kinect skeleton fusion for physical rehabilitation monitoring," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*, IEEE, 2014, pp. 5060–5063.

[60] J. Levinson and S. Thrun, "Unsupervised calibration for multi-beam lasers," in *Experimental Robotics*, Springer, 2014, pp. 179–193.

[61] W. Maddern, A. Harrison, and P. Newman, "Lost in translation (and rotation): Rapid extrinsic calibration for 2D and 3D LIDARs," in *In Proceedings of 2012 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2012, pp. 3096–3102.

[62] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Automatic Targetless Extrinsic Calibration of a 3D Lidar and Camera by Maximizing Mutual Information.," in *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence.*, 2012, pp. 2053–2059.

[63] J. Schmidt, F. Vogt, and H. Niemann, “Calibration–Free Hand–Eye Calibration: A Structure–from–Motion Approach,” in *Pattern Recognition*, W. G. Kropatsch, R. Sablatnig, and A. Hanbury, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 67–74.

[64] S. Schreiberhuber, J. Prankl, and M. Vincze, “Towards ScalableFusion: Feasibility Analysis of a Mesh Based 3D Reconstruction,” in *In Proceedings of the OAGM Workshop 2018*, 2018, pp. 47–52.

[65] L. Keselman, J. I. Woodfill, A. Grunnet-Jepsen, and A. Bhowmik, “Intel (r) realsense (tm) stereoscopic depth cameras,” in *In Proceedings of 2017 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).*, IEEE, 2017, pp. 1267–1276.

[66] L. Li, “Time-of-flight camera–an introduction,” *Technical white paper*, no. SLOA190B, 2014.

[67] M. Hansard, S. Lee, O. Choi, and R. P. Horaud, *Time-of-flight cameras: principles, methods and applications*. Springer, 2012, pp. 95.

[68] A. Menditto, M. Patriarca, and B. Magnusson, “Understanding the meaning of accuracy, trueness and precision,” *Accreditation and quality assurance*, vol. 12, no. 1, pp. 45–47, 2007.

[69] J. Canny, “A computational approach to edge detection,” in *Readings in Computer Vision*, Elsevier, 1987, pp. 184–203.

[70] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “ROS: an open-source Robot Operating System,” in *ICRA Workshop on Open Source Software*, vol. 3, 2009, p. 5.

[71] A. Handa, T. Whelan, J. McDonald, and A. Davison, “A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM,” in *In Proceedings of 2014 IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, 2014.

[72] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, “KinectFusion: Real-time dense surface mapping and tracking,” in *In Proceedings of 10th IEEE international symposium on Mixed and augmented reality (ISMAR)*, IEEE, 2011, pp. 127–136.

[73] *CloudCompare (version 2.9.1) [GPL software]. (2018).* `http://www.cloudcompare.org/`.

[74] M. A. Treiber, “Optimization for computer vision,” *Advances in Computer Vision and Pattern Recognition.*, pp. 978–979, 2013.

[75] K. Levenberg, “A method for the solution of certain non-linear problems in least squares,” *Quarterly of applied mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[76] D. W. Marquardt, “An algorithm for least-squares estimation of nonlinear parameters,” *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[77] S. Agarwal, K. Mierle, and Others, *Ceres Solver*, `http://ceres-solver.org`.

[78] M. J. Black and A. Rangarajan, "On the unification of line processes, outlier rejection, and robust statistics with applications in early vision," *International Journal of Computer Vision*, vol. 19, no. 1, pp. 57–91, 1996.

[79] A. E. Beaton and J. W. Tukey, "The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data," *Technometrics*, vol. 16, no. 2, pp. 147–185, 1974.

[80] I. Kolár, J. Slovák, and P. W. Michor, "Natural operations in differential geometry," 1999.

[81] P. Moulon, P. Monasse, and R. Marlet, "Global fusion of relative motions for robust, accurate and scalable structure from motion," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 3248–3255.

[82] D. Gutiérrez-Gómez, W. Mayol-Cuevas, and J. J. Guerrero, "Inverse depth for accurate photometric and geometric error minimisation in RGB-D dense visual odometry," in *In Proceedings of 2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 83–89.

[83] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[84] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *In Proceedings of 2011 IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011.

[85] P. Moulon, P. Monasse, R. Marlet, and Others, *OpenMVG. An Open Multiple View Geometry library.* `https://github.com/openMVG/openMVG`.

[86] L. Dagum and R. Menon, "OpenMP: an industry standard API for shared-memory programming," *IEEE computational science and engineering*, vol. 5, no. 1, pp. 46–55, 1998.

[87] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," in *In Proceedings of 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2012, pp. 573–580.

[88] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *AAAI/IAAI*, vol. 1999, no. 343-349, pp. 2–2, 1999.

[89] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *In Proceedings of 2004 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*, IEEE, vol. 1, 2004, pp. I–I.

[90] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *In Proceedings of 2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2014, pp. 15–22.

[91] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[92] S. Pertuz, D. Puig, and M. A. García, “Analysis of focus measure operators for shape-from-focus,” *Pattern Recognition*, vol. 46, pp. 1415–1432, 2013.

[93] S. K. Nayar and Y. Nakagawa, “Shape from focus,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 8, pp. 824–831, 1994, ISSN: 0162-8828.

[94] A. Thelen, S. Frey, S. Hirsch, and P. Hering, “Improvements in shape-from-focus for holographic reconstructions with regard to focus operators, neighborhood-size, and height value interpolation,” *IEEE Transactions on Image Processing*, vol. 18, no. 1, pp. 151–157, 2009.

[95] J. L. Pech-Pacheco, G. Cristóbal, J. Chamorro-Martinez, and J. Fernández-Valdivia, “Diatom autofocusing in brightfield microscopy: a comparative study,” in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, IEEE, vol. 3, 2000, pp. 314–317.

[96] G. Yang and B. J. Nelson, “Wavelet-based autofocusing and unsupervised segmentation of microscopic images,” in *In Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, vol. 3, 2003, pp. 2143–2148.

[97] H. Xie, W. Rong, and L. Sun, “Wavelet-Based Focus Measure and 3-D Surface Reconstruction Method for Microscopy Images,” in *In Proceedings of 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006, pp. 229–234.

[98] E. Mavridaki and V. Mezaris, “No-reference blur assessment in natural images using fourier transform and spatial pyramids,” in *Image Processing (ICIP), 2014 IEEE International Conference on*, IEEE, 2014, pp. 566–570.

[99] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2016.

[100] S. Ryan Fanello, C. Rhemann, V. Tankovich, A. Kowdle, S. Orts Escolano, D. Kim, and S. Izadi, “Hyperdepth: Learning depth from structured light without matching,” in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5441–5450.

[101] F. J. Lawin, P.-E. Forssén, and H. Ovrén, “Efficient Multi-Frequency Phase Unwrapping using Kernel Density Estimation,” in *European Conference on Computer Vision (ECCV)*, Springer International Publishing AG, 2016, pp. 170–185.

[102] P. C. Mahalanobis, “On the generalized distance in statistics,” National Institute of Science of India, 1936.

[103] D. D. Morris, K. Kanatani, and T. Kanade, “Uncertainty modeling for optimal structure from motion,” in *International Workshop on Vision Algorithms*, Springer, 1999, pp. 200–217.

[104] J.-L. Blanco, “A tutorial on SE(3) transformation parameterizations and on-manifold optimization,” *University of Malaga, Tech. Rep*, vol. 3, 2010.

[105] T. D. Barfoot and P. T. Furgale, “Associating uncertainty with three-dimensional poses for use in estimation problems,” *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 679–693, 2014.

[106] J. E. Gentle, *Numerical linear algebra for applications in statistics*. Springer Science & Business Media, 2012.

Georg
Halmetschlager-Funek

Curriculum Vitae

"How can we improve and robustify machine vision systems
by using multiple camera views?"

Personal Information

Name	**Georg Halmetschlager-Funek, MSc**.
Nationality	**Austria**.
Date of Birth	**14.02.1987**.
Place of Birth	**Vienna**.
Civil Status	**Married**.

Education

08/2015 – 12/2018	**PhD**, *Machine Vision for Robotic Systems*, TU Wien. Thesis:“Depth Sensors for Robot Vision”
10/2010 – 10/2014	**MSc**, *Automation and Control Engineering*, TU Wien. Thesis: “Crop Row Detection for the Autonomous Navigation of Field Robots”
09/2007 – 06/2010	**BSc**, *Mechatronics/Robotics*, University of Applied Science, FHTW.

Work Experience

01/2019 – present	**DAQRI**, *Austria*, Computer Vision Engineer. Research on Augmented Reality Systems.
02/2013 – 12/2018	**TU Wien, ACIN - Automation and Control Institute, Vision for Robotics Lab**, *Austria*, Project Researcher. Research on target-less autonomous camera calibration to improve 3D object segmentation and detection in challenging environments.
01/2010 – 05/2014	**Aplica Advanced Solutions GmbH, R&D**, *Austria*, Product Developer. Development and research on embedded, wireless, time-synchronized measurement systems and interface technology.

Projects

2015 – 2018	**EU H2020 Project “FLOBOT - Floor Washing Robot for Professional Users”**, *http://www.flobot.eu/*, Multi-view 3D object segmentation and detection in challenging real-life environments using stereo cameras and RGBD cameras.

2013 – 2015	**National Project, Sparkling Science "FRANC - Field Robot for the Advanced Navigation in bio-Crops"**, *http://franc.acin.tuwien.ac.at/*, Machine vision system for row guided in-field navigation using stereo cameras and infrared sensitive cameras.

Publications

- G. Halmetschlager-Funek, M. Suchi, M. Kampel, and M. Vincze, An Empirical Evaluation of Ten Different Depth Sensors for Robotic Systems *in Robotics and Automation Magazine*, vol. 26, no. 1, 2019.
- G. Halmetschlager-Funek, J. Prankl, and M. Vincze, Towards Autonomous Auto Calibration of Unregistered RGB-D Setups for Robotic Systems *in Proceedings of IROS*, 2018.
- A.Grünauer,G. Halmetschlager, J. Prankl, and M. Vincze, The Power of GMMs: Unsupervised Dirt Spot Detection for Industrial Floor Cleaning Robots *in Proceedings of TAROS*, 2017.
- G. Halmetschlager, J. Prankl, and M. Vincze, Towards Agricultural Robotics for Organic Farming *in Workshop Proceedings of ARW*, 2016.
- G. Halmetschlager, J. Prankl, and M. Vincze, Increasing the Precision of Generic Crop Row Detection and Tracking and Row End Detection *in Workshop Proceedings of IROS Workshop on Agri-Food Robotics*, 2015.
- G. Halmetschlager, J. Prankl, and M. Vincze, Evaluation of Different Importance Functions for a 4D Probabilistic Crop Row Parameter Estimation ,*in Workshop Proceedings of ARW*, 2015.
- G. Halmetschlager, J. Prankl, and M. Vincze, Probabilistic NIR and Depth Based Crop Line Identification,*in Workshop Proceedings of IAS-13*, 2014.
- G. Halmetschlager, J. Prankl, M. Vincze, Crop Row Detection for the Autonomous Navigation of Field Robots, *Master Thesis*, 2014.

Teaching

2016 – present	**TU Wien**, Laboratory on Basics in Industrial Robotics.
2017 – 2018	**FHTW**, Sabbatical Replacement for Prof. M. Vincze. Basic Course in Machine Vision and Mobile Robotics.

Additional Information

Awards and Honors	Thesis Award: Faculty Prize, 2015. MSc, Graduated with Honors, 2014. Graduation Ceremony Speech, 2010 & 2014. Scholarship for Academic Excellence, 2008 & 2009.
Miscellaneous	Elected Department Representative (Mechatronics/Robotics), 2009-2010. Elected Year Representative, 2008-2010.
Conferences & Summerschools	IROS2018, TAROS 2017, ARW/OAGM 2017, ICVSS 2016, BMVASS 2016, ARW/OAGM 2016, IROS 2015, ICRA 2015, ARW 2015, IAS-13 2014.

Vienna. May, 2019.

Georg Halmetschlager-Funek, MSc