Real-Time Processing of Range Data Focusing on Environment Reconstruction

DISSERTATION
zur Erlangung des Grades eines Doktors
der Ingenieurwissenschaften

vorgelegt von
M.Sc. Damien Lefloch

eingereicht bei der
Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen
Siegen 2017

Betreuer und erster Gutachter
Prof. Dr.-Ing. Andreas Kolb
Universität Siegen

Zweiter Gutachter
Prof. Tim Weyrich
University College London

Tag der mündlichen Prüfung
26 January 2018

Gedruckt auf alterungsbeständigem holz- und säurefreiem Papier

Acknowledgments

This dissertation would not have been possible without the constant support of persons during all these years and this section is meant to give them proper thanking.

I am first deeply grateful to my Ph.D. supervisor Prof. Dr. Andreas Kolb, who gave me such a great opportunity to work at his research group with full financial support during all these years. Thank you for your long discussions, for your time to teach me to write high quality papers and your continuous guidance.

I am also grateful to the German Research Foundation (DFG) for funding my last years of work as part of the research training group GRK 1564 *Imaging New Modalities*.

Big thanks go to all the co-authors of my publications. In particular, to Maik Keller, the perfect colleague to work with, my curiosity on online 3-D reconstruction blossoms thanks to you. The fastest responsive person I know and always providing nice comments. You are a great person and I just have one regret which is "to never have been to one of your concert". Prof. Dr. Tim Weyrich, it was always a pleasure to collaborate and write papers with you. You always point out questions that needed to be answered in the paper before reaching its final version. I was literally amazed by your efficiency to jump on a topic and find related works so rapidly. Thomas Högg, a far-distant colleague, for his coding skills and help on designing proper Graphics Pipeline. Hamed Sarbolandi, my second office mate, thank you for the nice discussions and your help in designing some experimental setups. A special thank goes to Markus Kluge, a new colleague arriving a few months before my departure, who offers me his great support to finish my latest publication.

Furthermore, I thank my office mates, Martin Pätzold and Farhoosh Alghabi for bringing so much great conversation and funs during this long working period. Additional thanks to my Bro Martin Pätzold who guides me to the proper approach for compressing robustly unit vectors. Julian Bader, more than a colleague, who gave me a place to sleep in his flat at Siegen when I was working until very late in my office. Ulrich Schipper and Martin Lambers for their helps on teaching me how to set-up our virtual reality room. Many thanks to Hendrik Hochstetter, as Martin Pätzold and Ulrich Schipper, for the great fun playing soccer.
I also thank Willi Gräfrath for his support on administrative things.
I would also thank the students I have been working with for some projects. Roberto Cespi helped to acquire data for the intensity related calibration of Time-of-Flight cameras. And Sebastian Schmitz, who writes OpenGL shaders to render blended surface elements using the elliptic weighted average approach.

Finally, deep thanks to my family for their great support. More precisely, my lovely wife Jun, you were

always supportive during a difficult time of my life and offering me many extra time slots to finish my thesis. Last but certainly not least, thanks to my father and my parents-in-law who take care of my sweet little girl Freya.

SEPT. 2017

DL.

M.Sc. Damien Lefloch

Real-Time Processing of Range Data Focusing on Environment Reconstruction

Abstract

With the availability of affordable range imaging sensors, providing real-time three-dimensional information of the captured scene, new types of Computer Vision applications arise. Such applications range from designing new Human-Computer interfaces (known as Natural User Interfaces) to the generation of highly detailed reconstructions of complex scenes (for example to keep track of cultural heritage or crime scenes), to autonomous driving and augmented reality.

These depth sensors are mostly based on two efficient technologies: the structured-light principle (such as the Xbox 360 version of the Kinect camera) and the time-of-flight (ToF) principle (as cameras implemented by **pmd**techonologies). When ToF cameras measure the time until the light emitted by their illumination unit is backscattered to their smart detectors, the structured-light cameras project a known light pattern onto the scene and measure the amount of distortion between the emitted light pattern and its image. Both technologies have their own advantages and weak points.

This dissertation is composed of 4 contributions. First, an efficient approach is proposed to compensate motion artifact of ToF raw images. Thereafter, a work on online three-dimensional reconstruction application has been investigated to improve the robustness of the camera tracker by segmenting moving objects. The second major contribution lies on a robust handling of noise on raw data, during the full reconstruction pipeline, proposing a new type of information fusion which considered the anisotropic nature of noise present on depth data, leading to faster convergence of high-quality reconstructions. Finally, a new method has been designed which uses surface curvature information to robustly reconstruct fine structures of small objects, as well as limiting the total error of camera drift.

M.Sc. Damien Lefloch

Zusammenfassung

Durch die Verfügbarkeit von kostengünstigen Nahfeldsensoren, die 3D daten der aufgenommenen Szene in Echtzeit erstellen, entstehen neue Anwendungen im Bereich Computer Vision. Diese Anwendungen reichen von der Erstellung neuer Mensch-Maschine-Schnittstellen (bekannt als Natural User Interfaces) über die Erstellung von sehr detaillierten Rekonstruktionen komplexer Szenen (z.B. in der Spuren an Tatorten oder Kulturstätten) bis hin zu Autonomem Fahren und Erweiterter Realität.

Diese Tiefensensoren basieren hauptsächlich auf zwei effizienten Technologien, dem: Structured-Light (SL) Prinzip (wie in der Xbox 360 Kinect Kamera) and Time-of-Flight (ToF) Prinzip (wie Kameras der Firma **pmd**technologies). Während ToF-Kameras die Zeit zwischen Lichtemission der Beleuchtungseinheit und Empfang der Rückstreuung auf dem "smart detectors" messen, projizieren SL Kameras ein bekanntes Lichtmuster in die Szene und messen die Verzerrung zwischen ausgesendetem Muster und dem resultierenden Bild. Beide Technologien haben ihre Vor- und Nachteile. Diese Dissertation besteht aus vier Beiträgen. Wir schlagen einen effizienten Ansatz vor, um Bewegungsartefakte von ToF-Rohbildern zu kompensieren. Danach arbeiten wir an 3D-Rekonstruktionsanwendungen und verbessern die Robustheit des Kameratrackings durch die Segmentierung von bewegten Objekten.

Der zweite Beitrag liegt in der robusten Handhabung von Rauschen in den Rohdaten über die ganze Verarbeitungskette der Rekonstruktion. Hier wird eine neue Art der Informationsfusion verwendet, welche die anisotropischen Eigenschaften von Rauschen in den Tiefendaten berücksichtigt und damit eine schnellere Konvergenz für hochqualitative Rekonstruktionen erzielt.

Abschließend wird eine Methode entworfen welche die Information über die Oberflächenkrümmung verwendet um auch feine Strukturen von kleinen Objekten robust zu rekonstruieren. Zusätzlich wird der Gesamtfehler des Kameradrifts eingeschränkt.

Nomenclature

CCD . . . . . Charge-Coupled Device
CMOS . . . . . Complementary Metal Oxide Semi-conductor
CUDA . . . . . Compute Unified Device Architecture
CWIM . . . . . Continuous Wave Intensity Modulation
DoF . . . . . Degree(s) of Freedom
GPU . . . . . Graphics Processing Unit
ICP . . . . . Iterative Closest Point
NIR . . . . . Near Infrared
OpenGL . . . . . Open Graphics Library
PBF . . . . . Point-Based Fusion
PCL . . . . . Point Cloud Library
RADAR . . . . . RAdio Detection And Ranging
RANSAC . . . . . RAdom SAmple Consensus
RMSE . . . . . Root Mean Square Error
SDA . . . . . Standard Deviation Average
SL . . . . . Structured-Light
SLAM . . . . . Simultaneous Localization And Mapping
SNR . . . . . Signal-to-noise Ratio
SSAO . . . . . Screen-Space Ambient Occlusion
SVD . . . . . Singular Value Decomposition
ToF . . . . . Time-of-Flight
TSDF . . . . . Truncated Signed Distance Function
VGA . . . . . Video Graphics Array : display standard (640×480)
WLS . . . . . Weighted Least Squares

List of Symbols

px	pixel unit
x	horizontal axis of an image (unit in px)
y	vertical axis of an image (unit in px)
$\boldsymbol{u} = (x, y)^\top \in \mathbb{R}^2$	pixel position in the camera image
s_x, s_y	pixel size of the camera chip for both axis (unit in m.px^{-1})
f	camera focal (unit in m)
f_x, f_y	camera focal for both axis (unit in px)
c_x, c_y	camera principal point for both axis (unit in px)
$\boldsymbol{K} \in \mathbb{P}^{3\to2}$	intrinsic camera matrix
$t \in \mathbb{N}$	timestamp or frame ID
$\boldsymbol{T}^{t\to(t-1)} \in \mathbb{SE}^3$	rigid transformation between two consecutive frames
$i, j \in \mathbb{N}$	point indices
$\boldsymbol{p}_i^t$	i-th input point in frame t
$l \in \mathbb{N}$	iteration index or pyramid level
$\widehat{\boldsymbol{x}} \in \mathbb{R}^n$	normalized vector ($\widehat{\boldsymbol{x}} = \boldsymbol{x}/\|\boldsymbol{x}\|$)
$\widehat{\boldsymbol{n}} \in \mathbb{R}^3$	normal vector
$\kappa_1, \kappa_2 \in \mathbb{R}$	first and second principal curvature values
$\widehat{\boldsymbol{e}}_1, \widehat{\boldsymbol{e}}_2 \in \mathbb{R}^3$	corresponding directions of curvature ($\langle \widehat{\boldsymbol{e}}_1, \widehat{\boldsymbol{e}}_2 \rangle = 0$)
$\mathcal{D}^t(\boldsymbol{u}) \in \mathbb{R}$	input depth map
$\mathcal{W}^t(\boldsymbol{u}) \in \mathbb{R}$	reliability map
$\mathcal{V}^t(\boldsymbol{u}) \in \mathbb{R}^3$	*vertex map* of object points corresponding to $\boldsymbol{u}$
$\mathcal{N}_{\mathcal{V}^t(\boldsymbol{u})}$	local neighbourhood set $\mathcal{N}$ of point $\mathcal{V}^t(\boldsymbol{u})$
$\|\mathcal{N}\| \in \mathbb{N}$	number of elements (cardinality) of set $\mathcal{N}$
$\mathcal{N}^t = \{\widehat{\boldsymbol{n}}\}$	normal map (per-point attribute)
$\mathcal{R}^t \in \mathbb{R}$	point-radius map (per-point attribute)
$\mathcal{K}^t = \{\widehat{\boldsymbol{e}}_1, \kappa_1, \kappa_2\}$	curvature map (per-point attribute)
$\mathcal{I}_t(\boldsymbol{u}) = \{i\}$	index map

I dedicated this dissertation to my father Jean-Louis who made lot of sacrifices for offering me the opportunity to pursue my studies, and my uncle Bernard who regrettably left too early.

Contents

List of figures

List of Tables

It is paradoxical, yet true, to say, that the more we know, the more ignorant we become in the absolute sense, for it is only through enlightenment that we become conscious of our limitations. Precisely one of the most gratifying results of intellectual evolution is the continuous opening up of new and greater prospects.

Nikola Tesla (∗1856 – †1943)

1 Introduction

HUMANS (or animals in general) perceive their environment with both eyes (binocular images) and mostly understand the surrounding world using the single vision sense. One could argue that the perception of the surrounding world required only a 2-D colour image to be solved. However, what would be the difference between a photograph of a person's face and a photograph of a high-resolution A4 printed image of the same face? The sole use of 2-D information is not enough to perceive differently regarding both of these cases. Thus 3-D information is a key point for proper perception. Still recently, several news reported that the face identification of Samsung's Galaxy S8 smart-

phone can easily be bypassed by a single photograph[1].

For this reason, a wide range of applications is now relying not solely on the 2-D image information, but use additional inputs such as depth information in order to either be more robust (e.g. face identification problem) or solve problems that were not possible via 2-D images only. Many applications are now using depth sensors to capture environments in high quality. With the availability of affordable range imaging sensors, providing real time 3-D information of the captured scene, new types of Computer Vision applications arise. Such applications range from designing new Human-Computer interfaces (known as Natural User Interfaces) to the generation of high detailed reconstructions of complex scenes (for example to keep track of cultural heritage or crime scenes), to autonomous driving and augmented/virtual reality. These depth sensors are mostly based on two different types of technology: the structured-light (SL) principle (such as the Xbox 360 version of the Kinect camera) and the time-of-flight (ToF) principle (as cameras implemented by **pmd**techonologies).

Nowadays, the 3-D camera technology is matured enough to be directly integrated into mobile devices (e.g. the Asus Zenfone AR[2], the Lenovo Phab 2 Pro[3], and the iPhone X[4]). In this dissertation, frameworks for dense reconstruction applications using depth sensors are described based on new 3-D algorithms and high-quality surface rendering. Challenges such as the real time constraint and the sensor uncertainty are addressed and solutions are presented for capturing large scale or low-feature environments.

1.1 Context

This study was conducted as part of the research training group GRK 1564 *Imaging New Modalities* combining experts in different domains such as electronics, physics, nanotechnology and material science, and computer science. This graduate school located at the University of Siegen is focusing on developing new long range ToF cameras, as well as graphene-based devices or Terahertz sensors, and computer graphics and vision algorithms. These algorithms are used to visualise and process raw data coming from these different sensors. That is where this thesis is contributing. New algorithms for raw depth data should be implemented to solve different problems or improve current state-of-the-art methods. Online dense 3-D reconstruction using depth sensors was the main application that drives this work. Solving such general problem leads to a variety of applications as robots explorations, security, augmented/virtual reality, entertainments, etc.

Collaborations with Microsoft Research in Cambridge, and the University College London led to various

[1] source : http://www.businessinsider.fr/us/samsung-galaxy-s8-facial-recognition-tricked-with-a-photo-2017-3/

[2] Asus Zenfone AR: https://www.asus.com/us/Phone/ZenFone-AR-ZS571KL/

[3] Lenovo Phab 2 Pro: http://www3.lenovo.com/us/en/virtual-reality-and-smart-devices/augmented-reality/-phab-2-pro/Lenovo-Phab-2-Pro/p/WMD00000220

[4] iPhone X: https://www.apple.com/iphone-xr/

publications as well as fruitful discussions during this full study.

1.2 Goals and Challenges

In this dissertation the topic of 3-D range imaging is investigated in a twofold way: depth map preprocessing, improving the overall quality of the input data as well as the accumulation of range images into a consistent 3-D model, is introduced. Concerning the depth map preprocessing topic, the main challenge lies in the fact that depth maps given by range sensors have usually lower image resolution compared to the ones provided by standard colour cameras, making the calibration process more difficult, and that depth cameras suffer from intrinsic and extrinsic error characteristics which highly increase the measurement uncertainty.

A part of this work focuses on reducing the sensor uncertainty of ToF sensors by first calibrating the camera system to retrieve its intrinsic characteristics (see Chapter 2) and second, by tackling a common extrinsic problem that occurs in dynamic environments known as ToF motion artifacts. Chapter 3 introduces this problem and describes a solution to improve the quality of depth maps given by ToF sensor. Decreasing the noise uncertainty of depth data is a valid step for any application that uses as input depth data. Further contributions of this work are related to the 3-D reconstruction application.

The main challenges of online dense 3-D reconstruction can be split into two categories. The first category describes practical challenges which are summarized as being able to densely reconstruct **large scenes** in **real-time** using as small as possible **memory footprint**. The second category describes challenges that are directly linked to the final output of the method: How to achieve high quality 3-D reconstruction? This also relates to the quality (or the signal-to-noise) of the input data. This work gives different solutions to those problems.

Chapter 5 focuses on the robustness of 3-D reconstruction methods for different types of scenes (large environments, low-depthfeature scenes) using two different depth cameras with their unique sensor uncertainty. Low-depth feature scenes are composed of objects with small depth variations (such as a brick wall). This chapter shows how measurement uncertainty has a huge impact on the final output of 3-D reconstruction systems. The standard algorithm used to track the ego motion of the camera is not robust enough against noise leading to poor quality reconstruction for low-depthfeature scenes. This chapter shows how difficult it is for real-time 3-D reconstruction systems to properly handle high sensor uncertainty in the case of low-depthfeature scenes. Chapter 5 gives solutions to tackle this problem.

Measurement uncertainty is also a valid information for 3-D reconstruction methods if properly modelled and used. Chapter 6 describes a method that uses measurement uncertainty to better fuse data together. Data accumulation is an important step for online dense 3-D reconstruction methods. The main challenge of the data accumulation is that the full information must be shrunk into a common representation that uses

a small memory footprint without deteriorating the original information (i.e., without losing detail). This chapter focuses on methods to improve the fusion of depth data using the anisotropic nature of the noise characteristic of range sensors.

In addition, methods presented in this dissertation were essentially driven by the achievements of the following general goals:

- **real-time constraint** as mentioned previously this constraint is an important feature which provides direct feedback to the user, allowing them to automatically adapt themselves to the current results. The real-time constraint is always achieved in this work by the implementation of several GPU-based processing pipelines (using both OpenGL and CUDA frameworks).

- **high quality results** the real-time constraint should not have the counter effect to lower the quality of the result provided by any of these methods. Concerning the online 3-D reconstruction topic, the presented methods achieve high-quality results in term of fine and detailed reconstruction for large-scale sequence, but also high quality in term of drift reduction and robust camera tracking.

- **modularity** each of the GPU-based pipelines usually offers the possibility to be extended easily.

1.3 Contributions

This dissertation is composed of four results (**R1**, **R2**, **R3** and **R4**).

In Lefloch et al. [LHK13], a new approach to compensate motion artifacts of ToF raw phase images has been proposed (see Chapter 3). Extending the original work by Lindner and Kolb [LK09], the contributions in the first result **R1** are:

R1.1 improve the computation speed of the method by limiting the number of Optical Flows required,

R1.2 evaluation of the correction using both simulated and real data showing the robustness of the approach.

Collaborating with Maik Keller, a new online 3-D reconstruction framework [KLL+13] was proposed using augmented points (or surface elements) as model representation. This thesis will present the work that has been achieved regarding online 3-D reconstruction in dynamic environments (see Section 4.2). The following contributions are part of the second result **R2**:

R2.1 a fast and robust algorithm to segment dynamic objects in the scene,

R2.2 improve the robustness of the camera tracker by rapidly updating and reflecting the current reconstruction model via the information of segmented dynamic objects.

The third and fourth results **R3** and **R4** are still in the direction of online 3-D reconstructions and are based on our original work proposed by Keller et al. [KLL+13] without being necessarly bound to it. Both approaches could be used with a different model representation such as the original Kinect-Fusion method proposed by Newcombe et al. [NDI+11].

Based on the original work of Keller et al. [KLL+13], these two last results focus on high quality dense 3-D reconstruction using low-cost depth sensors that are subject to strong input noise. Both approaches were implemented using the original point-based fusion due to its simplicity, but are not restricted to this model representation. For example, the volumetric grid proposed in the original Kinect-Fusion by Newcombe et al. [NDI+11] is also a valid candidate for both methods.

To improve the overall quality of dense reconstruction of low-depth feature scenes, more impacted by the noise uncertainty of the sensor, a new method [LKS+17] has been developed (see Chapter 5) that uses surface curvature information. The contributions of the third result **R3** are:

R3.1 the first online reconstruction design to systematically incorporate curvature as an independent surface attribute,

R3.2 an iterative closest point (ICP) variant that considers curvature for both dense correspondences searching and weighting for increased stability,

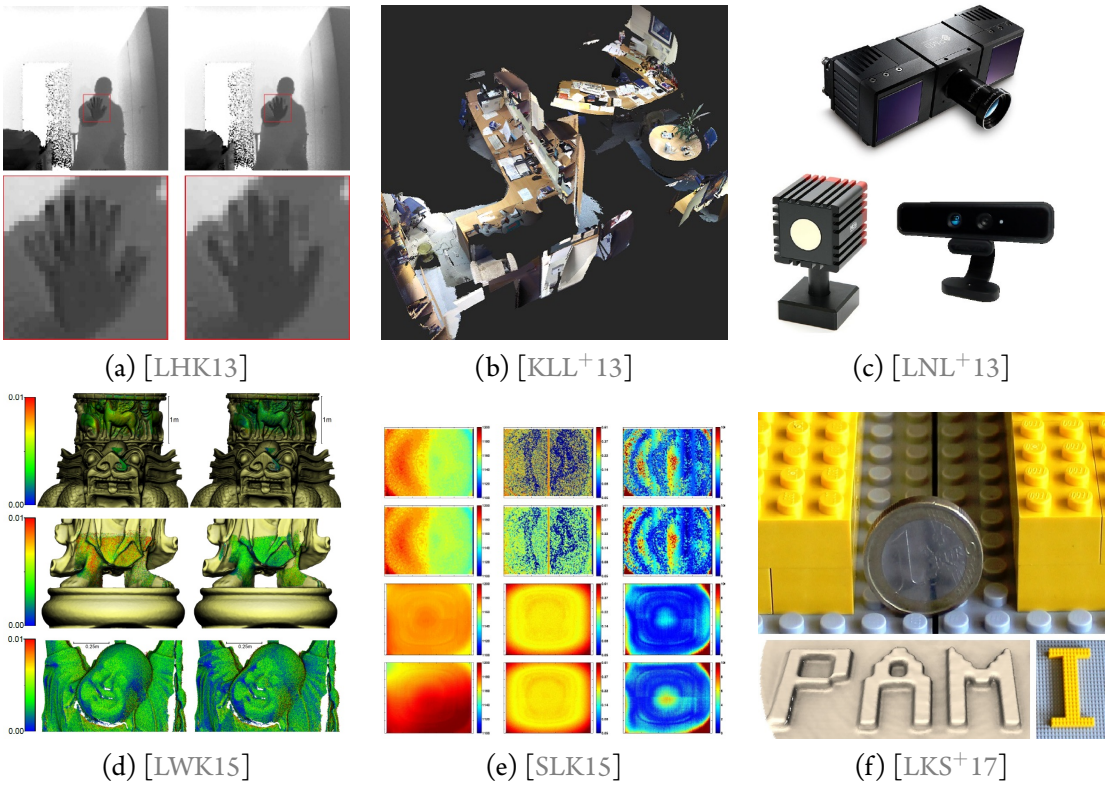
R3.3 a method to efficiently blend curvatures in the fusion stage,

R3.4 fast and high quality, curvature-aware local surface reconstruction using a local and lightweight representation of the model representation (called *index map*).

The last result took an orthogonal approach to robustly tackle the noise in raw depth data. A new dense reconstruction framework [LWK15] has been proposed which stores per model point a reliability matrix directly given by the noise uncertainty of the corresponding measurement (see Chapter 6). The contributions of the last result **R4** are:

R4.1 a novel symmetric anisotropic distance measure that is applied to establish more robust correspondences between input and model points in the fusion step,

R4.2 a novel anisotropy-aware fusion technique for accumulation of anisotropic input data into the model which leads to a better convergence and high quality of reconstruction,



(a) [LHK13] (b) [KLL+13] (c) [LNL+13]

(d) [LWK15] (e) [SLK15] (f) [LKS+17]

Figure 1.3.1: Teaser images of the publication that will be presented in this thesis.

R4.3 a data compression scheme for point-based model representation implying efficient storage of attributes per point without loss of quality.

Figure 1.3.1 shows a teaser image for all the works accomplished during this study. This thesis will present most of those publications.

The following list is referring all publications, in chronological order, that were achieved during this study:

P.1 Damien Lefloch, Thomas Högg, and Andreas Kolb. **Real-time motion artifacts compensation of tof sensors data on gpu**. *SPIE*, May 2013.

P.2 Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. **Real-time 3d reconstruction in dynamic scenes using point-based fusion**. In *Proceedings of IEEE International Conference on 3D Vision*, June 2013.

P.3 Damien Lefloch, Rahul Nair, Frank Lenzen, Henrik Schäfer, Lee Streeter, Michael J Cree, Reinhard

Koch, and Andreas Kolb. **Technical foundation and calibration methods for time-of-flight cameras**. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, September 2013.

P.4 Thomas Högg, Damien Lefloch, and Andreas Kolb. **Real-time motion artifact compensation for pmd-tof images**. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, September 2013.

P.5 Thomas Högg, Damien Lefloch, and Andreas Kolb. **Time-of-flight camera based 3d point cloud reconstruction of a car**. *Computers in Industry*, December 2013.

P.6 Damien Lefloch, Tim Weyrich, and Andreas Kolb. **Anisotropic point-based fusion**. In *International Conference on Information Fusion*, July 2015.

P.7 Hamed Sarbolandi, Damien Lefloch, and Andreas Kolb. **Kinect range sensing: Structured-Light versus Time-of-Flight Kinect**. *Computer Vision and Image Understanding*, October 2015.

P.8 Damien Lefloch, Markus Kluge, Hamed Sarbolandi, Tim Weyrich, and Andreas Kolb. **Comprehensive use of curvature for robust and accurate online surface reconstruction**. *IEEE Trans. Pattern Analysis and Machine Intelligence*, January 2017.

This thesis mainly relates to the publications **P.1**, **P.2**, **P.3**, **P.6**, **P.7** and **P.8**, while the other publications relate to collaborative work .

1.4 Thesis Outline

This dissertation is a product of several works that have been published in various publications during the time spent in the *Computer Graphics and Multimedia Systems Group* at the University of Siegen. Therefore, this thesis directly reflects all contributions of those publications. This thesis is decomposed into five main parts:

- Chapter 2 introduces basic concept of camera geometry and different range camera principles with a focus on the ToF technology and the resulting characteristic errors. It also describes both works regarding depth sensors foundation [LNL+13] and evaluation [SLK15]. The first one was published as a chapter in the "Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications" Springer book and the second one in the "Computer vision and image understanding" journal.

- Chapter 3 presents an efficient method to compensate motion blur on depth data given by ToF cameras. Motion blur on ToF data is a well-known problem and is intrinsically linked to the ToF acquisition principle. This work was published in the proceedings of "Three-Dimensional Imaging, Visualization, and Display" (SPIE).

- Chapter 4 gives a detailed introduction to online 3-D reconstruction pipelines by first describing the related works and presenting all different steps required to solve this challenging problem. The last sections of this chapter (see Section 4.2) is describing the contribution of this work on the online dense reconstruction in dynamic environments, presented in the "Proceedings of the Joint 3DIM/3DPVT Conference" (3DV).

- Chapter 5 is presenting a new online 3-D reconstruction framework that uses curvature information to improve the overall quality of reconstruction. This work was published in the prestigious "IEEE Transactions on Pattern Analysis and Machine Intelligence" (PAMI).

- Finally, Chapter 6 describes another contribution to the topic of online 3-D reconstruction that uses the anisotropic behaviour of depth measurement uncertainties to improve the data fusion process. This work was presented in the "International Conference on Information Fusion" (FUSION).

- Throughout this thesis, the reader will be referred to the Appendix A, located at the end of this dissertation, for more details on specific methods or algorithms.

The last chapter concludes the thesis giving a summary of the main presented contributions and discussing about future works and improvements.

Tell me and I forget. Teach me and I remember. Involve me and I learn.

Benjamin Franklin (∗1706 – †1790)

2 Fundamentals

Theoretical background is essential to clearly understand this thesis. The basics of camera geometry will be first introduced with the corresponding notations which will be used along with the thesis. This is followed by the presentation of two different range camera principles used during this work: the Time-of-Flight (ToF) and the Structured-light (SL) camera principles.

2.1 Perspective Camera Model

The following section is separated into two parts describing a simple camera model known as *the pinhole model*. First, all camera parameters which belong intrinsically to the camera will be described; for a complete

overview of the notations used in this work, refer to the list of symbols. Since these parameters are intrinsic to the camera they are usually fixed, and thus required a single processing step to be determined known as calibration. They describe how 3-D points, expressed in camera-centred coordinate, are projected to the camera image plane. The second part focuses on the scene related parameters that are extrinsic to the camera and which relates 3-D scene coordinates (also known as world coordinates) to the camera coordinates.

2.1.1 Intrinsic Parameters

The pinhole model is the simplest camera model that mathematically describes how cameras are composing images. 3-D points expressed in the camera-centred coordinates are linearly projected to the image plane of the camera, i.e., the imaging chip. The camera chip is based either on the Complementary Metal Oxide Semiconductor (CMOS) technology, or on the Charge-Coupled Device (CCD) technology. Both technologies have advantages and drawbacks and are widely used in the market. The CMOS technology is less power demanding than the CCD technology. However, due to its principle, each individual pixel line of the chip is acquired at a different time frame leading to problem for high-speed horizontal motion (a problem known as the Rolling Shutter).

Typically, the intrinsic parameters are defined by the linear calibration matrix $\boldsymbol{K}$ which internally holds the camera focal length f, the pixel size s_x, s_y, and the optical image centre $\{c_x, c_y\}$ of the imaging chip. The focal length f (usually expressed in mm) corresponds to the distance between the pinhole centre (camera centre) and the image plane. The pixel size, usually expressed in μm px^{-1}, refers to the length of each dimension of the pixel. And the optical image centre, also known as principal point, is the 2-D projection of the optical centre to the image plane.

The intrinsic camera matrix $\boldsymbol{K}$ is defined as:

$$\boldsymbol{K} = \begin{pmatrix} f_x & a & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.1}$$

where $f_x = \frac{f}{s_x}$, $f_y = \frac{f}{s_y}$, and a refers to the axis skew that causes shear distortion. The axis skew a is generally set to zero for most cameras since the angle of pixel axes is very near of $\frac{\pi}{2}$ radians. From now on, a is set to zero since all cameras used in this thesis have a zero-skew value. Furthermore, the inverse of the

intrinsic matrix can be explicitly computed by:

$$\boldsymbol{K}^{-1} = \begin{pmatrix} \frac{1}{f_x} & 0 & -\frac{c_x}{f_x} \\ 0 & \frac{1}{f_y} & -\frac{c_y}{f_y} \\ 0 & 0 & 1 \end{pmatrix} \tag{2.2}$$

It is a transformation that first translates a 2-D pixel point to the image coordinates centred at the principal point and then normalises it by both focal dimensions. Applying the inverse of the intrinsic matrix to an homogeneous pixel point $\boldsymbol{u} = (x \;\; y \;\; 1)^\top$ leads to a normalised image point coordinate $\bar{\boldsymbol{u}} = (\bar{x} \;\; \bar{y} \;\; 1)^\top$. Additionally, if the distance $\mathcal{D}(\boldsymbol{u})$at pixel $\boldsymbol{u}$ is known, then the corresponding 3-D vertex point $\mathcal{V}(\boldsymbol{u})$ can be computed by simply scaling it by $\mathcal{D}(\boldsymbol{u})$:

$$\mathcal{V}(\boldsymbol{u}) = \begin{pmatrix} X \\ Y \\ Z = \mathcal{D}(\boldsymbol{u}) \end{pmatrix} = \bar{\boldsymbol{u}}\, \mathcal{D}(\boldsymbol{u}) = \boldsymbol{K}^{-1}\, \boldsymbol{u}\, \mathcal{D}(\boldsymbol{u}). \tag{2.3}$$

This is known as a *back-projection* transform.

Figure 2.1.1 shows the perspective projection principle via the pinhole model. In practical use, it is common to assume that the image plane is in front of the camera centre at the reflected focal-length distance. Note also, how 3-D points belonging to a surface object (blue and green circles) are projected to the image plane composing the final captured image. To generate this figure, the **TotemPole** data set raw images were used which was originally presented in the work of Zhou and Koltun [ZK13]. The 3-D reconstructed model was extracted by processing all input depths using the contribution presented in Chapter 5 and converted to a mesh via Poisson surface reconstruction [KH13]. Frame 166 refers to the visualisation of the virtual captured image.

The pinhole model, or perspective projection, ensures that a straight line in 3-D world coordinates is projected to a straight line in the captured image. However, in practice this linear assumption does not apply.

Lens distortion Non-linear distortion of the image is caused by the camera lens. The most common distortion is called the radial distortion that manifests itself in distorting a square into a *barrel*. A second kind of distortion, known as tangential distortion, occurs when the lens is not perfectly aligned with the camera's optical centre. Nonetheless, the tangential distortion was almost non-existent for the cameras used in this thesis. Thus, we use the simple radial distortion model proposed by Zhang [Zha00] which retrieves a normalised distorted pixel coordinate $\tilde{\bar{\boldsymbol{u}}} = (\tilde{\bar{x}} \;\; \tilde{\bar{y}})^\top$ from a normalised corrected pixel (distortion-free)

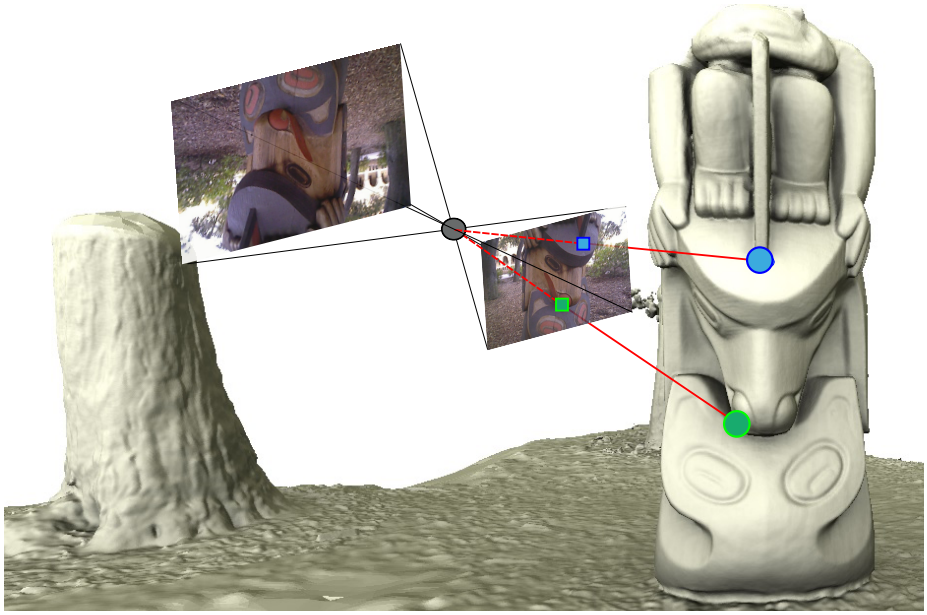


Figure 2.1.1: Visualisation of the surface point projections using a pinhole model. Green and blue circles represent surface points of one object of the scene. The green and blue squares are the projection of 3-D points (circles) into the image (virtual view of the camera). Image generated using the **Totem-Pole** data set given by Zhou and Koltun [ZK13] (rgb-frame 166).

$\bar{\boldsymbol{u}} = (\bar{x}\ \bar{y})^\top$. Here the radial distortion is modelled using two radial parameters k_1 and k_2 as:

$$\begin{aligned} \tilde{\bar{x}} &= \bar{x}\,(1 + k_1\ \|\rho\| + k_2\ \|\rho\|^2) \\ \tilde{\bar{y}} &= \bar{y}\,(1 + k_1\ \|\rho\| + k_2\ \|\rho\|^2) \end{aligned} \tag{2.4}$$

where $\|\rho\| = \bar{x}^2 + \bar{y}^2$.

For any undistorted pixel $\boldsymbol{u}$, one can retrieve easily the corresponding distorted pixel in the image by applying first $\bar{\boldsymbol{u}} = \boldsymbol{K}^{-1}\ \boldsymbol{u}$, the distortion scheme on Equation 2.4, and finally $\tilde{\boldsymbol{u}} = \boldsymbol{K}\ \tilde{\bar{\boldsymbol{u}}}$.

Note that for efficiency reason, one could pre-computed a look-up table (or undistorted map) that stores for each undistorted pixel $\tilde{\boldsymbol{u}}$, the coordinates of its distorted version $\boldsymbol{u}$. With $\boldsymbol{u} \in \mathbb{R}^2$, bilinear interpolation is usually used to compute the proper pixel value. Special care must be taken at edges during interpolation (with strong attention on depth data).

From now on, distortion will not be discussed in the following, assuming that a correction was already applied. In this way, the pinhole model can be correctly used.

From pixel to world unit It is possible to deduce some properties of the acquired scene using the intrinsic

parameters of the camera and the composed image of the scene. However, since there are an infinite number of camera parameters capturing the exact same image (e.g. scaling the focal length and the pixel size by the same number), an additional information of the scene should be known. For example, if the Y_i-Cartesian distance between the camera centre and the centre of an object i is known, then the distance between the camera and this object (also called object's depth d_i) can be retrieved using the intercept theorem:

$$d_i = f_y \frac{Y_i}{(y_i - c_y)}, \tag{2.5}$$

where y_i represents the y-dimension pixel coordinate of the top object point in the captured image (expressed in px unit).

2.1.2 Extrinsic Parameters

As stated in Section 2.1.1, the exact projection position on the imaging chip of a 3-D point in the field of view of the camera can be calculated using the pinhole model. However, in order to apply the camera projection, the 3-D points should be expressed in the camera coordinate $\{O_c X_c Y_c Z_c\}$ where O_c is the centre of the camera, X_c and Y_c are collinear to both axes of the image plane $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively, and Z_c has the same direction than the one given by the camera's optical axis. A transformation composed by a rotation and a translation is enough to transform the world coordinates to the camera frame. This transformation matrix is called extrinsic camera parameters and is describing the coordinates of the scene. Applying any change to the position or orientation of the camera will directly affect the extrinsic parameters.

Let $\{O_w X_w Y_w Z_w\}$ be the world coordinates and $\boldsymbol{T}^{w\to c}$ the extrinsic matrix that transforms the world coordinates to the camera frame $\{O_c X_c Y_c Z_c\}$. In homogeneous coordinates, $\boldsymbol{T}^{w\to c}$ is a 4×4 matrix:

$$\boldsymbol{T}^{w\to c} = \begin{pmatrix} & & & t_x \\ & R^{w\to c} & & t_y \\ & & & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{2.6}$$

where $R^{w\to c}$ is a rotation matrix $\in \mathbb{R}^{3\times 3}$. A rotation matrix has only 3 degrees of freedom (3-DoF) known as the 3 Euler angles. Since the translation component has also 3 parameters t_x, t_y and t_z, the matrix $\boldsymbol{T}^{w\to c}$ is a linear transformation with 6-DoF.

Additionally, it is useful to determine the inverse transformation $\boldsymbol{T}^{c\to w} = \left(\boldsymbol{T}^{w\to c}\right)^{-1}$ that transforms any point expressed in the camera frame to the world coordinates. For example, if one wants to know the 3-D line equation in world coordinates of a camera pixel ray (computed by back-projection), such an inverse

transform is required. Due to linearity of this transform, the inverse is simply computed as:

$$\left(\boldsymbol{T}^{w\to c}\right)^{-1} = \begin{pmatrix} & R^{c\to w} = R^{\top\, w\to c} & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} & & & -t_x \\ & I_{3\times 3} & & -t_y \\ & & & -t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.7}$$

A homogeneous 3-D point $\boldsymbol{P} = (X_p^w\ Y_p^w\ Z_p^w\ 1)^\top$ expressed in world coordinates is transformed to the camera frame by simply applying the extrinsic matrix $\boldsymbol{T}^{w\to c}$:

$$\begin{pmatrix} X_p^c \\ Y_p^c \\ Z_p^c \\ 1 \end{pmatrix} = \begin{pmatrix} & R^{w\to c} & & t_x \\ & & & t_y \\ & & & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_p^w \\ Y_p^w \\ Z_p^w \\ 1 \end{pmatrix}. \tag{2.8}$$

with:

$$\begin{aligned} X_p^c &= r_{11}X_p^w + r_{12}Y_p^w + r_{13}Z_p^w + t_x \\ Y_p^c &= r_{21}X_p^w + r_{22}Y_p^w + r_{23}Z_p^w + t_y \\ Z_p^c &= r_{31}X_p^w + r_{32}Y_p^w + r_{33}Z_p^w + t_z \end{aligned} \tag{2.9}$$

where $r_{ij}, (i,j) \in [1,2,3]^2$ refers to the element of the matrix $R^{w\to c}$ located at the i-th line and j-th column.

Finally, $\boldsymbol{P}$ can be projected to the image plane of the camera as follows:

$$Z_p^c \begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} & R^{w\to c} & & t_x \\ & & & t_y \\ & & & t_z \end{bmatrix}}_{\text{perspective transformation}} \begin{pmatrix} X_p^w \\ Y_p^w \\ Z_p^w \\ 1 \end{pmatrix}. \tag{2.10}$$

Calibration The process to retrieve all camera parameters is called calibration. However, since it is not the focus of this thesis, the reader is invited to check the method proposed by Zhang [Zha00] for more details on photometric calibration of rgb or monochrome cameras. The basic idea is to use, as input, several images (taken from different positions and orientations) of a known-size planar object (a checker-board). Corners of the checker-board are automatically detected and retrieved with sub-pixel resolution. The world coordinate system is placed on one of the 4 extreme corners of the checker-board so that all corners have

a simplified 3-D position (taking $Z = 0$) and a non-linear optimization (such as Levenberg-Marquadt optimisation) is used to precisely retrieve all camera parameters for each individual image. The objective function is basically the sum of all reprojection errors (for all images and all corners of the checker-board).

2.2 Non-contact Range Imaging Principles

This section briefly explains most of the techniques that are commonly implemented on range sensors in order to estimate depth information. Range imaging devices usually provide depth information as a single channel image (known as *range image* or *depth map*). Whereas pixel values of an intensity image are indirectly related to the surface geometry, the ones from range images encode the position of the surface directly.

Range images are usually represented in two basic forms. One is a list of 3-D coordinates (point cloud) expressed in the reference frame of the range device, for which no specific order is required. The other is a matrix of depth values along the directions of the (x, y) image axes, which makes spatial organisation explicit.

Depth maps are also referred to as 2.5-D images since they encode only the surface profile information.

Non-contact techniques can be organized in two sub categories, i.e. passive and active methods. Note that the focus will be further given to the active methods since this work is based on the processing of range data given by active depth sensors.

2.2.1 Passive methods

Passive methods are techniques that rely purely on 2-D imaging without "altering" the observed scenery. They are commonly known as *shape from* x techniques, where $x \in \{\text{stereo, motion, (de)focus, } \ldots\}$. The following will shortly introduce the *stereovision* principle.

Stereovision Stereovision or passive stereoscopy uses two cameras, similar to the human vision. The problem is the following: *Given two different views, can we obtain a depth map?* A sub-problem is: *Given two different views, can we obtain the camera pose of each view?* Both cameras generate images of the same scene at different viewpoints. The main problem of stereovision systems is to find reliable correspondences (relating left and right cameras) to retrieve the disparity (see the following section Section 2.2.2-i for more details). A stereovision system is ruled by several constraints. The best known and most used constraint is the epipolar geometry. The fundamental matrix (or essential matrix for calibrated systems) is expressing this constraint and can be computed via a set of features (see [HZ03]) for more details).

Figure 2.2.1 illustrates the Multi-view stereovision. The epipolar geometry constrains the correspondence search to a single line on the image plane (green line). Note that this constraint is only valid for static scenes.

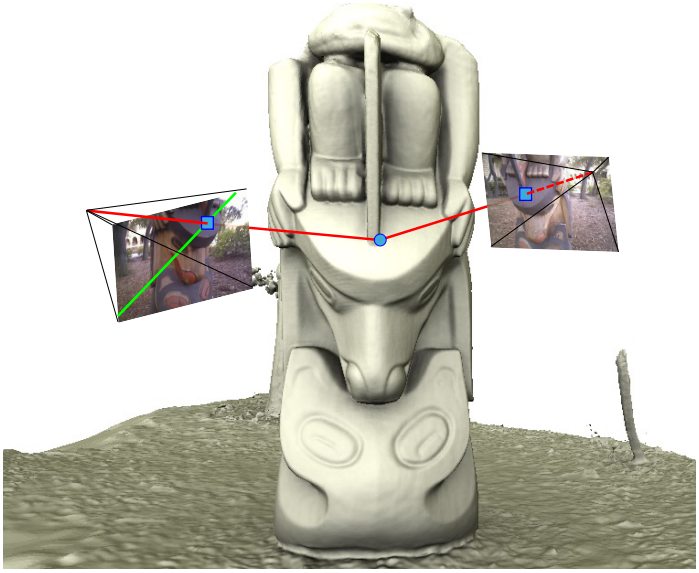


Figure 2.2.1: Visualisation of the stereovision principle. Red lines indicate the pixel ray projections. The green line is the pixel ray of the right image projected into the left camera image. Note that the green line passes through the projection of the surface point (blue circle projected into the left camera as blue square). This principle demonstrates the basic epipolar lines geometry. Image generated using the input **Totem-Pole** data set given by Zhou and Koltun [ZK13] (frame 166 for the left camera and frame 1984 for the right camera).

A surface point (blue circle) is projected to the camera focal plane through the pinhole model (blue square corresponds to the pixel where the surface point is projected). This pixel ray is projected to the left camera as the epipolar line (green line) which passes through the projection of the same surface point into the left camera focal plane. This simple principle called the epipolar geometry greatly limits the correspondence search and is used to compute efficiently pixel disparities (inversely proportional to the depth). To speed up the correspondence search, images are usually rectified. Both images are projected into a common plane. This has the effect to simplify the correspondence search to a single dimension (the baseline direction).

2.2.2 Active methods

In contrast to passive methods, active methods are techniques that are using an additional device "altering" the scenery. This additional device is usually a light emitter (laser, diodes, etc.) but not restricted to. For example, Radio Detection And Ranging (RADAR) is emitting electromagnetic waves. The following will briefly introduce the *Structured-Light* (SL) and the *Time-of-Flight* (ToF) principles.

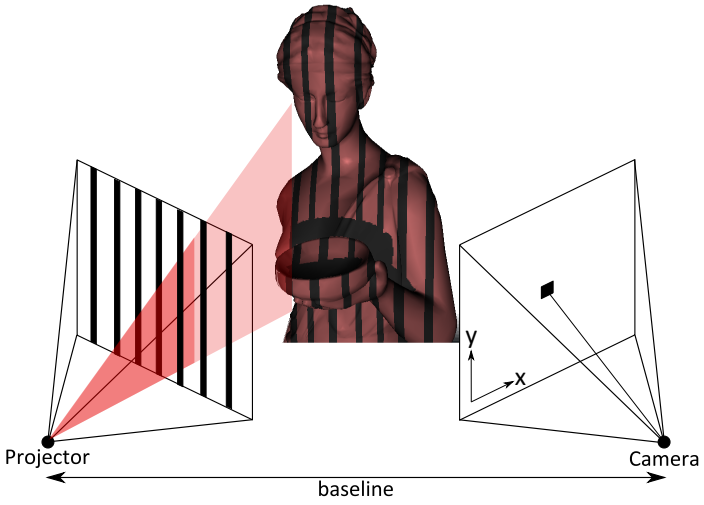


Figure 2.2.2: Structured-Light principle.

2.2.2-1 Structured-Light

Even though the principle of the SL-based range sensing is comparatively old, the launch of the Microsoft Kinect™ (Kinect^{SL}) in 2010 as an interaction device for the XBox 360 clearly demonstrates the maturity of the underlying principle.

The SL approach is categorised as an active stereovision technique. One or several known patterns is sequentially projected onto an object, which gets deformed by the object shape. The object is then observed by a camera from a different viewpoint. Analysing the distortion of the observed pattern, i.e., the disparity from the source pattern, one can extract depth information (Figure 2.2.2). Knowing the intrinsic parameters of the camera and additionally the *baseline* b (distance between the observing camera and the projector), the depth of pixel $\boldsymbol{u}$ can be computed using the corresponding disparity value $\mathcal{DI}(\boldsymbol{u})$ as $\mathcal{D}(\boldsymbol{u}) = \frac{b\,f}{\mathcal{DI}(\boldsymbol{u})}$. As the disparity $\mathcal{DI}$ is usually given in pixel-units, the focal length f is also converted to pixel units via $f_x = \frac{f}{s_{\text{px}}}$. In most cases, the camera and the projector are only horizontally displaced, thus the disparity values are all given as horizontal distances. In this case s_{px} resembles the horizontal pixel size s_x. Both depth range and accuracy relate to the baseline, i.e. longer baselines allow for robust depth measurements at far distances.

There are different options to design the projection patterns for a SL range sensor. Several approaches were proposed based on the SL principle to estimate the disparity resulting from the deformation of the projected light patterns. In the simplest case the stripe-pattern sequence realises a binary code which is used to decode the direction from an object point is illuminated by the beamer.

SL cameras, such as the Kinect^{SL}, use a low number of patterns (only one for the Kinect^{SL}), to obtain a

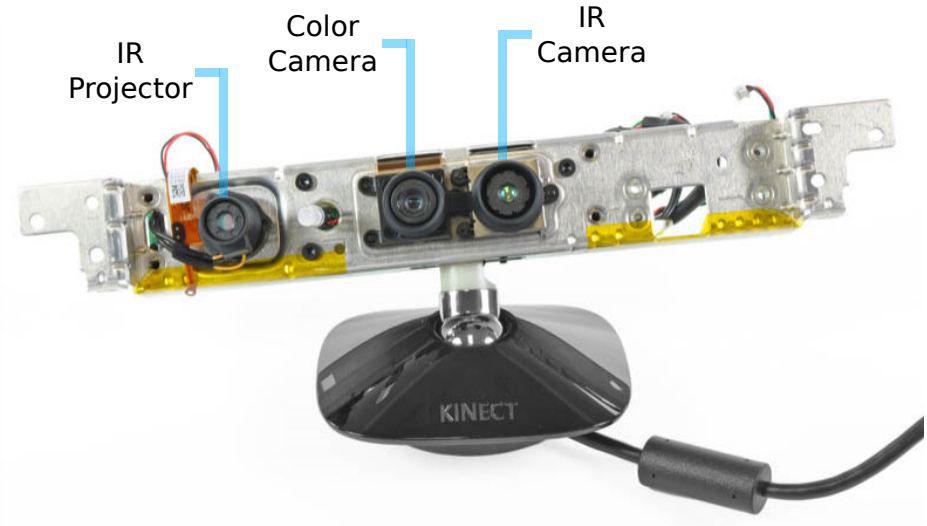


Figure 2.2.3: Components of the Kinect^{SL} camera. Original picture provided by **iFixit** available at http://www.ifixit.com/Teardown/Microsoft-Kinect-Teardown/4066/1

depth estimation of the scenery at a "high" frame rate (30 Hz). Typically, it is composed of a Near Infrared (NIR) laser projector combined with a monochrome CMOS camera which captures depth variations of object surfaces in the scene.

The Kinect^{SL} camera is based on the standard SL principle. The device is composed of two cameras, i.e. a colour RGB, a monochrome NIR camera, and a NIR projector including a 850 nm laser diode. The baseline between the projector and the camera is approximately 7.5 cm (see Figure 2.2.3). The NIR projector uses a known and fixed dot pattern to illuminate the scenery. See the ROS.org community website [KM12], for a detailed description of the most probable algorithm (deduced by several experimentations) used by the Kinect^{SL} camera to compute the final depth map (disparity map and calibrated system).

2.2.2-II Time-of-Flight

Time-of-Flight (ToF) is the name given to a variety of methods that measure the time that a specimen (object, light, particle, sound, etc.) takes to travel a certain distance through a specific medium.

In the 17th century, Galileo Galilei (1564–1642) made a first attempt to measure the speed of light since he was convinced that there was no such thing as infinite speed; unfortunately, his experiment fails to show that the speed of light was finite due to relatively small distances between the light "emitter" and the "receiver" (two different persons were alternatively lighting a torch at a distance of one mile apart). In fact, due to the colossal speed of light ($299,792\ \text{km} \cdot \text{s}^{-1}$), about 5μs only is needed for the light to travel one single mile.

This simple fact explains why he was not able to find a proper finite value for the speed of light.

Two centuries later, Armand Hippolyte Fizeau $(1819–1896)$ was the first scientist to successfully measure the speed of light on Earth in 1849. Even if his experiment leads to a 5% error from light velocity ground-truth (he measured a velocity of $315,300\,\mathrm{km}\cdot\mathrm{s}^{-1}$) and was worse than the deduction from astronomic observation based on light aberration, that was the first concrete experiment that measured the speed of light. Concerning astronomic observation, Ole Christensen Rømer $(1644–1710)$ calculated a a velocity of $300,000\,\mathrm{km}\cdot\mathrm{s}^{-1}$ from the revolution of Io, one of the Jupiter satellites by collecting the precise dates of the eclipses of Io over many years. From his data, he realised that when the Earth was nearest to Jupiter, eclipses of Io would occur about 11 minutes earlier than predicted, and conversely, 11 minutes later than predicted when the Earth was farthest from Jupiter.

2.3 Time-of-Flight cameras

ToF cameras provide an elegant and efficient way to capture 3-D geometric information of real environments in real time. However, due to their operational principle, ToF cameras are subject to a large variety of measurement error sources. Over the last decade, an important number of investigations concerning these error sources were reported and have shown that they were caused by factors such as camera parameters and properties (sensor temperature, chip design, etc.), environment configuration and the sensor hardware principle. ToF sensors usually provide two measurement frames at the same time from data acquired by the same pixel array; the *depth* and *amplitude* images. The latter image corresponds to the amount of returning active light signal and is also considered a strong indicator of quality/reliability of measurements. For ToF cameras, the on-board technology is more complicated than "standard" camera, and leads to different errors which strongly reduces the quality of the measurements.

The section is organized as follows: Part 2.3.1 gives an overview of the basic technological foundation of two different ToF camera principles. And Part 2.3.2 ends this section by presenting all different measurement errors of ToF sensors.

2.3.1 Signal Theory

Continuous Modulation Approach

Most of the ToF manufacturers built-in the following principle in their cameras such as **pmd**technologies[1], Mesa Imaging[2] or Soft Kinetic[3] (cf. Figure 2.3.1); or more recently, the second version of Microsoft Kinect

[1] http://www.pmdtec.com/
[2] http://www.mesa-imaging.ch/
[3] http://www.softkinetic.com/

Figure 2.3.1: Different ToF phase-based camera models available on the market. A PMD CamCube 2.0 (left), a swissranger SR 400 (middle), a DepthSense DS325 (right) and the second version of the Microsoft Kinect (bottom)

based on ToF technology (Kinect$^{\text{ToF}}$) originally designed for the Xbox One[4]. These cameras are able to retrieve range images at a frame rate of 30 Hz; **pmd**technologies has already designed faster device (the Camboard pico flexx) which operates at 90 Hz and has successfully integrated 3-D technology on Phablets (e.g. the ASUS Zenfone AR). Note that common ToF cameras usually modulate light at high frequency (e.g. ≈20 MHz) providing depth data smaller than 8 m and thus are highly suitable for middle range applications.

The continuous modulation principle, also known as a continuous wave intensity modulation [Lan00] (CWIM), is based on the correlation of the emitted signal o_τ shifted by an offset phase τ and the incident signal r resulting from the reflection of the modulated active illumination (NIR light) by the observed scene. CWIM is used to estimate the distance between the target (i.e., observed objects) and the source of the active illumination (i.e., the camera). CWIM ToF sensors directly implement the correlation function on the chip, composed of what is known in the literature as smart pixels [Lan00].

The correlation function $c(t)$ at a specific phase offset sample $\tau = 0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}$ is defined as

$$c_\tau(t) = r(t) * o_\tau(t) = \lim_{T\to\infty} \int_{-T/2}^{T/2} r(t) \cdot o_\tau(t)\ dt. \tag{2.11}$$

[4] http://www.xbox.com/fr-FR/xbox-one/accessories/kinect

Both emitted and incident signals can be expressed as a cosinusoidal function:

$$o_\tau(t) = \cos\left((\omega + f_m\tau)\cdot t\right), \qquad r(t) = I + A\cos\left(\omega t + \phi\right) \tag{2.12}$$

where $\omega = 2\pi f_m$ represents the angular frequency of f_m, I is the signal's offset due to DC background light, A the amplitude of the reflected signal and ϕ is the phase shift directly relating to the object distance. Using trigonometric relations [Lan00], one can simplify the correlation function as:

$$c_\tau = \frac{A}{2}\cos\left(\tau + \phi\right) + I. \tag{2.13}$$

There are three unknowns in Equation 2.13 so at least three measurements are required to perform a single estimation of distance, amplitude and offset. Typically, four samples of the correlation function c are sequentially acquired at specific discrete phase offsets $\mathcal{A}^i = c_\tau, \tau = i\cdot\frac{\pi}{2}, i = 0,1,2,3$. More measurements improve the precision but also incorporates additional errors due to the sequential sampling such as motion blur (see Section 3.2 to correct this problem). The measured amplitude A, phase ϕ and intensity I are given by:

$$\phi = \text{arctan2}\left(\mathcal{A}^3 - \mathcal{A}^1, \mathcal{A}^0 - \mathcal{A}^2\right), \tag{2.14}$$

$$I = \frac{1}{4}\cdot\sum_{i=0}^{3}\mathcal{A}^i, \tag{2.15}$$

$$A = \frac{1}{2}\cdot\sqrt{\left(\mathcal{A}^3 - \mathcal{A}^1\right)^2 + \left(\mathcal{A}^0 - \mathcal{A}^2\right)^2}. \tag{2.16}$$

Once the phase ϕ is reconstructed, the object distance d is easily computed using the speed of light in the dominating medium $c \approx 3\cdot 10^8 m\cdot s^{-1}$ and the modulation frequency of the active illumination f_m:

$$d = \frac{c}{4\pi f_m}\phi. \tag{2.17}$$

Since the described principle is mainly based on phase shift calculation, only a range of distances within one unambiguous range $[0, 2\pi]$ can be retrieved. This range depends on the modulation frequency f_m used during the acquisition giving a maximum distance of $d_{max} = \frac{c}{2f_m}$ that can be computed. The factor 2 relates to the fact that the active illumination needs to travel back and forth between the observed object and the camera. It is understood that in this simple depth retrieval calculation from the phase shift ϕ, simplifications are made which leads to possible measurement errors, e.g. the assumption that the active illumination

module and the ToF sensors are placed in the same position in space; which is physically impossible.

Pulse Based Approach

Conversely, pulse modulation is an alternative ToF principle which generates a pulse of light, of known width, coupled with a fast shutter observation. The 3DV System camera is using this class of technology also known as shuttered light-pulse (SLP) sensor to retrieve depth information. The basic concept lies on the fact that the camera projects a NIR pulse of light with a known duration and discretised the front of the reflected illumination. This discretisation is realised before the returning of the entire light pulse using a fast camera shutter. The portion of the reflected pulse signal describes the distance to the observed object. Conversely to the unambiguous range seen in CWIM approach, the depth of interest is directly linked to the duration of the light pulse and the duration of the shutter ($t_{\text{pulse}+\delta_s}$). This phenomenon is known as *light wall*. The intensity signal captured by the sensor during the shutter time is strongly correlated with the depth of the observed object, since nearer object will appear brighter. This statement is not fully exact, since the intensity signal also depends on the observed object reflectivity property. As Davis stated [DGB03], double pulse shuttering hardware provide a better depth measurement precision than the ones based on a single shutter.

For an in-depth evaluation of a pulse-based range sensing device (Hamamatsu prototype), please refer to the recent work of Sarbolandi et al. [SPK18].

The following section will describe the main error sources and characteristics of ToF range devices. Due to the lack of availability of pulse based range cameras (mainly all ToF range suppliers implement the CWIM principle), the presented error sources and characteristics will refer to CWIM-based ToF sensors but can be directly applied to the pulse-based principle for most of it.

2.3.2 Error Sources and Characteristics

In this section, a full understanding of ToF camera error sources is developed (errors identification and explanation). Methods to correct errors that are only related to extrinsic influences (e.g., the measured scene) will be discussed thoroughly in Chapter 3.

Besides integration time, that directly influences the Signal-to-Noise Ratio (SNR) of the measurement and consequently, the variance of the measured distance, the user can influence the quality of the measurements made by setting the f_m value to fit the application. As stated by Lange [Lan00], as f_m increases the depth resolution increases and conversely the maximum depth value of the unambiguous range decreases.

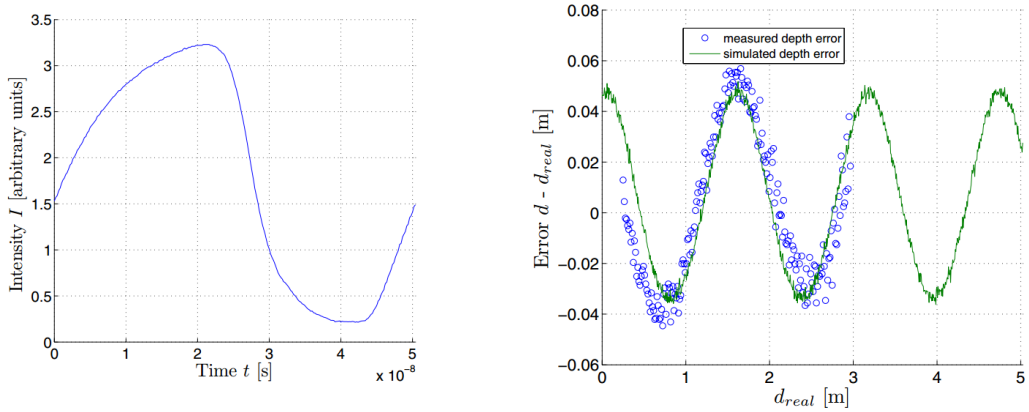


Figure 2.3.2: Left: Measured modulation of the PMD light source: Right: Mean depth deviation as a function of the real distance. Images courtesy of Schmidt et al. [SJ09]

Systematic Distance Error

Systematic errors occur when the formulas used for the reconstruction do not model all aspects of the actual physical imager. In CWIM cameras a prominent error is caused by differences between the actual modulation and correlation functions and the idealised versions used for calculations. In case of a sinusoidal modulation Section 2.3.1 , higher-order harmonics in the modulating light source (Figure 2.3.2) induce deviations from a perfect sine function. Applying the formula in part 2.3.1 to model the correlation of a real world physical light source leads to a periodic "wiggling" error which causes the calculated depth to oscillate around the actual depth. The actual form of this oscillation depends on the strength and frequencies of the higher order harmonics [Lan00, Rap07]. There are two approaches for solving this problem. The first approach is to sample the correlation function with more phase shifts and extend the formulas to incorporate higher order harmonics [DCC$^+$08]. With current two-tap sensor this approach induces more errors when observing dynamic scenes. The second approach is to keep the formulas as they are and estimate the residual error between true and calculated depth [LK06, SBK08]. The residual can then be used in a calibration step to eliminate the error. Alternatively, Payne et al. [PDCC10] employ a phase modulation of the amplitude signal to attenuate the higher harmonics in the emitted amplitude.

Intensity-related Distance Error

In addition to the systematic wiggling error, the measured distance is greatly altered by an error dependent of the total amount of incident light received by the sensor. Measured distances of lower reflectivity objects

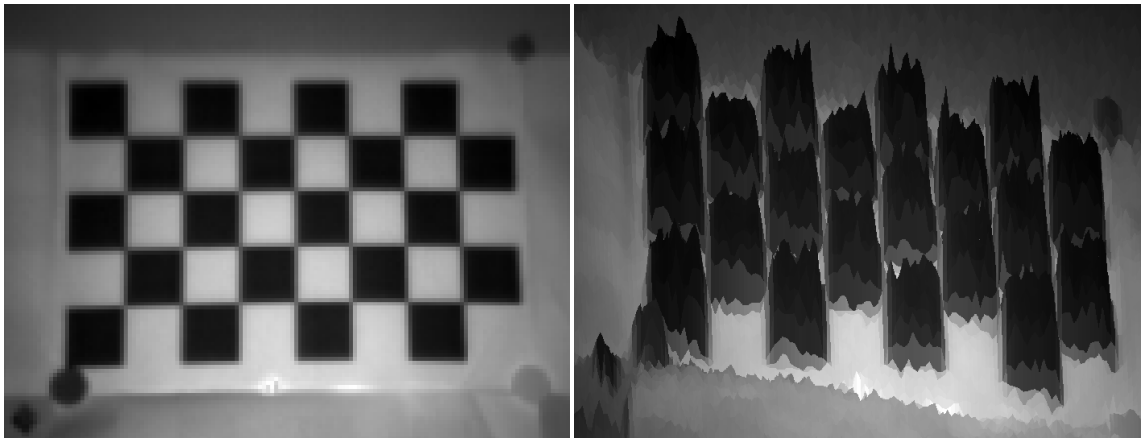


Figure 2.3.3: Impact of the intensity-related distance error on the depth measurement: The left image shows the intensity image given by a ToF camera. The right image shows the surface rendering obtained from the depth map, coloured by its corresponding intensity map. These images were generated using input images acquired by the PMD CamCube 2.0.

appear closer to the camera (up to 3 cm drift for the darkest objects for the PMD CamCube 2.0 camera). Figure 2.3.3 highlights this error effect using a simple black-and-white checkerboard pattern. This error is usually known as Intensity-related distance error and the causes are not fully understood yet [Lin10].

Nevertheless, Schmidt [Sch11] shows that ToF sensor has a nonlinear response during the conversion of photons to electrons. Lindner et al. [Lin10] claims that the origin of the intensity-related error is assumed to be caused by non-linearities of the semiconductor hardware.

A different point of view would be to consider the effect of multiple returns caused by inter-reflections in the sensor itself (scattering between the chip and the lens). Since the signal strength of low reflectivity objects is considerably weak, they will be more affected by this behaviour than for higher signal strength given by brighter objects. For more information about multi-path problems in ToF cameras, please refer to Section 2.3.2.

Depth Inhomogeneity

An important type of errors in ToF imaging, the so-called *flying pixels*, occurs along depth inhomogeneities. To illustrate these errors, a depth boundary is considered with one foreground and one background object. If the solid angle extent of a sensor pixel falls on the boundary of the foreground and the background, the recorded signal is a mixture of the light returns from both areas. Due to the nonlinearity of depth measurement errors and the phase ambiguity, the resulting depth is not restricted to the range between foreground and background depth but can attain any value of the camera's depth range. The fact that today' ToF sensors

provide only a low resolution promotes the occurrence of *flying pixels* of both kinds (resulted from depth boundaries and resulted from phase ambiguity).

The problem of depth inhomogeneities can be considered as a multipath problem, since also here light from different paths is mixed in one sensor cell. In the case of *flying pixels*, however, local information from neighbouring pixels can be used to detect them since they only occur at boundaries.

Motion Artifacts

As stated in Section 2.3.1, CWIM ToF imagers need to sample the correlation between the incident and the reference signals at least using 3 different phase shifts. Ideally, these raw images would be acquired simultaneously. Current two-tap sensors allow for two of these measurements to be made simultaneously, such that at least one more phase sample is needed. Usually, further raw images are acquired to counteract noise and compensate for different electronic characteristics of the individual taps, such as different gain factors. Since these (pairs) of additional exposures must be made sequentially, dynamic scenes lead to erroneous distance values at depth and reflectivity boundaries.

Methods for compensating motion artifacts will be discussed in the following chapter Section 3.1.

Multipath Interference

The standard CWIM model for range imaging assumes that the light return to each pixel of the sensor is from a single position in the scene. This assumption, unfortunately, is violated in most scenes of practical interest, thus multiple returns of light do arrive at a pixel and generally lead to erroneous reconstruction of range at that pixel. In fact, the light can travel multiple paths to intersect the viewed part of the scene and the imaging pixel—the *multipath interference* problem. Godbaz [God12] provides a thorough treatment of the multiple return problem, including a review covering full-field ToF and other ranging systems with relevant issues, such as point scanners (refer to Godbaz [God12] or Lefloch et al. [LNL+13] for more details).

Multipath interference can also occur intra-camera due to the light refraction and reflection of an imaging lens and aperture [Sha56, Bar64, ST91]. Such light scattering leads to distorted reconstructed ranges throughout the scene with larger influence on low reflective objects.

Other Error Sources

ToF sensors suffer from the same errors as standard camera sensors. The most important error source in the sensor is a result of the photon counting process in the sensor. Since photons are detected only by a certain probability, Poisson noise is introduced. See Seitz [Sei08] and the thesis by Schmidt [Sch11, Sec. 3.1] for detailed studies on the Poisson noise. An experimental evaluation of noise characteristics of different ToF

cameras has been performed in by Erz & Jähne [EJ09]. Besides from that other kind of noise, e.g. dark (fixed-pattern) noise and read-out noise, occur.

In ToF cameras, however, noise has a strong influence on the estimated scene depth, due to the following two issues

- The recorded light intensity in the raw channels is stemming from both active and background illumination. Isolating the active part of the signal reduces the SNR. Such a reduction could be compensated by increasing the integration time, which on the other hand increases the risk of an over-saturation of the cells, leading to false depth estimation. Therefore, a trade-off in the integration time must be made, often leading to a low SNR in the raw data, which occurs especially in areas with extremely low reflectivity or objects far away from the sensor.

- Since the estimated scene depth depends non-linearly on the raw channels (cf. Eqs. 2.14 and 2.17), the noise is amplified in this process. This amplification is typically modelled ([Lan00, FPR+09]) by assuming Gaussian noise in the raw data and performing a sensitivity analysis. By this simplified approach, it turns out that the noise variance in the final depth depends quadratically on the amplitude of the active illumination signal. In particular, the variance can change drastically within the different regions of the scene depending on the reflectivity and the distance of the objects.

2.4 Depth Map Pre-processing

Depth map pre-processing is an important step to improve the accuracy of applications that use depth maps as input. Firstly, methods to properly calibrate range cameras in order to improve the precision of the depth measurement will be introduced, followed by a presentation of filters that reduce the amount of noise in the depth map.

2.4.1 Range Camera Calibration

The focus of this section is the calibration of ToF depth cameras from image data. Due to the small image resolution of ToF cameras and the limited field of view, it is clear that the calibration process is challenging. ToF cameras also suffer from non-negligible lens distortions, making the calibration process harder since the image resolution of ToF devices is typically much lower than with modern optical cameras. Additionally, ToF cameras use their built-in NIR illumination to light the observable region, making far objects and image borders appear more darker since less light are collected on those regions. Early results show that the quality of the calibration using the approaches as described above is poor [LK06, KRI06].

However, there is also an advantage of using depth cameras, since the camera distance z can be estimated with high accuracy from the depth data, eliminating the f/z ambiguity. The calibration plane can be aligned with all depth measurements from the camera by plane fitting. Hence, all measurements are utilized simultaneously in a model-based approach that compares the estimated plane fit with the real calibration plane. More generally, a virtual model of the calibration plane is built, including not only geometry, but also surface colour, and is synthesised for comparison with the observed data. This *model-driven analysis-by-synthesis approach* exploits all camera data simultaneously, and allows furthermore to combine the ToF camera with additional colour cameras, which are rigidly coupled in a camera rig. The coupling of colour cameras with depth cameras is the key to high-quality calibration, since it combines the advantages of colour and depth data. High-resolution colour cameras with large field of view allow a stable and accurate pose estimation of the rig, while the depth data disambiguates z from f. The synthesis part is easily ported to GPU-hardware, allowing for fast calibration even with many input images[5]. For details about this approach, refer to [BK08, SBK08]. The approach allows further to include nonlinear depth effects, like the wiggling error, and reflectance-dependent depth bias estimates into the calibration [LK07].

However, the calibration is only valid at a specific camera temperature, since the behaviour changes with the temperature ([SFW08, Sch11]). Figure 2.4.1 shows the drift of depth measurement in function of acquisition time (linked with temperature drift) for both Kinect versions. To compensate depth temporal noise, 200 frames are averaged each 15 seconds over two hours of acquisition. Cameras are static and the acquired scene is a single white-wall in a constant-temperature black room (temperature variance of the room is below $0.1°$C). SDA refers to the *standard deviation average* that is the average of all standard deviation given by all valid pixels in the image. This experiment is given by the recent work of Sarbolandi et al. [SLK15] that compares and evaluates both versions of the Kinect cameras.

2.4.2 Outlier Removal

The following describes the problem of *flying pixels* and the different methods for correcting it. These methods can be separated into two groups; methods that directly process the 2-D raw image of the ToF cameras or methods that work with the 3-D point clouds.

Median filtering is a simple and efficient means for a rough correction of *flying pixels*, which are outside the objects' depth range (refer to [SBSS08] for a more involved filtering pipeline). Denoising methods to a certain extent are capable of dealing with *flying pixels*. The reason is that regions of depth inhomogeneities are typically one-dimensional structures and *flying pixels* appear only in a narrow band along these regions. Therefore, out-of-range *flying pixels* can be regarded as outliers in the depth measurement. Denoising meth-

[5]Software is available at `http://www.mip.informatik.uni-kiel.de/tiki-index.php?page=Calibration`

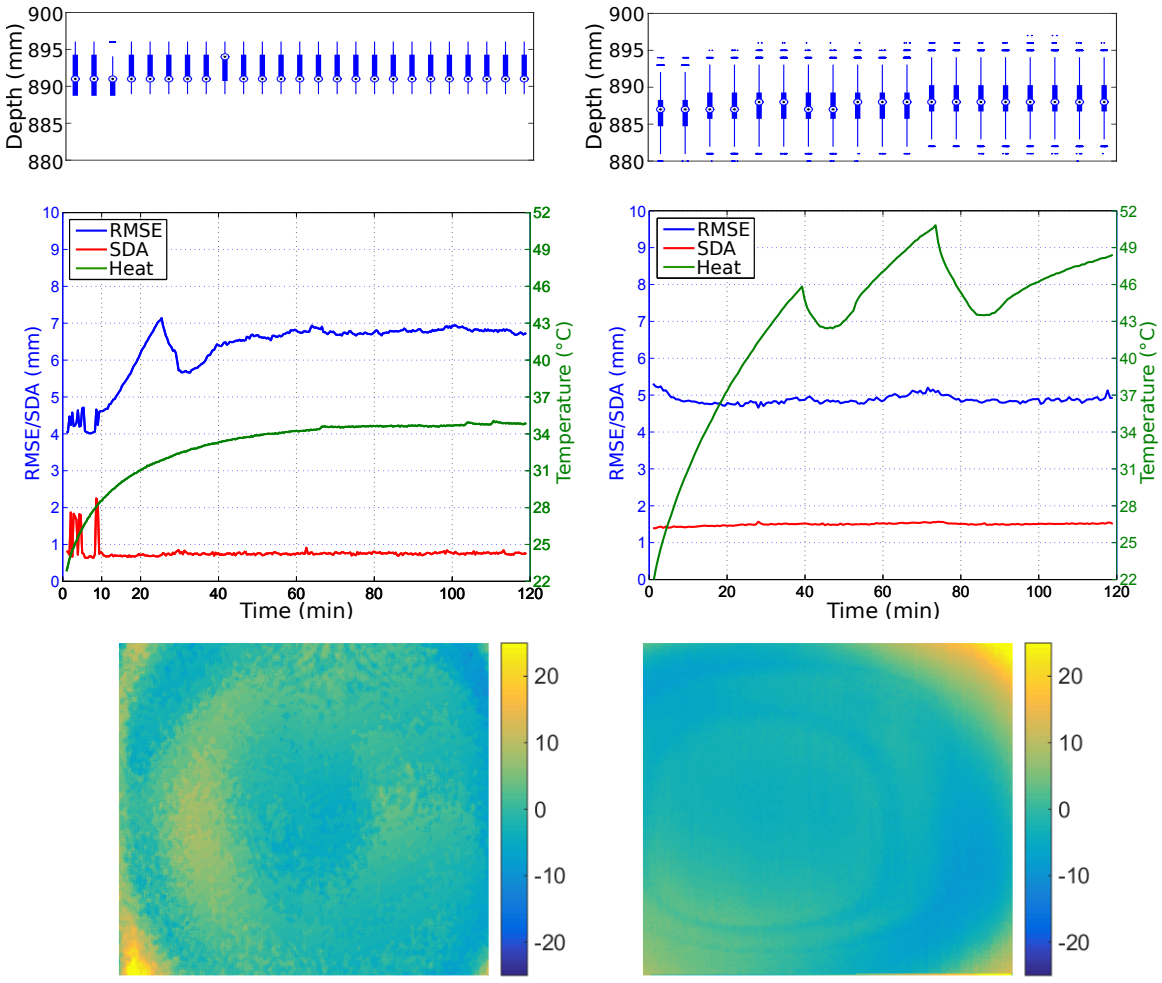


Figure 2.4.1: Box plot error mean and temperature versus warm-up time for Kinect^{SL} (left) and Kinect^{ToF} (right). The last row shows the depth error (mm) after 1 hour of acquisition. (*Image courtesy of [SLK15]*).

ods in general are robust against such outliers, and produces reconstructions with a certain spatial regularity.

Standard approaches identify such pixels, e.g., by confidence measures [RDP+11], and discard them. Since each ToF devices has its own set of inaccuracies, Reynolds et al. [RDP+11] proposed to train a random forest that takes as input the measured point of the ToF camera and output its corresponding confidence measure. Confidence measures were computed by simply looking at the difference of the ground-truth depth and the camera output. More straightforward methods consist of computing the confidence measure based on the amplitude of the signal output (basically low and high amplitude values lead to unreliable distance measure). However, those direct methods are not able to filter out unreliable measurement

such as *flying* pixels. The depth value of the discarded pixel is reconstructed using information from the surrounding pixels. The pixel has to be assigned to either background or foreground. Super-resolution approaches [LLK08, PBP08] allow to assign parts the pixel area to each of the objects.

Furthermore, when 3-D data (point clouds) is considered, geometrical information can be used to correct for *flying pixels*. For example, one can cluster the 3-D data to determine the underlying object surface ([MBE+08, SMD+08]).

Finally, another approach [STDT09, CSC+10] consists of fusing point clouds from *different* sources with sub-pixel accuracy. Here, it is substantial to reliably identify *flying pixels*, so that they can be removed before the actual fusion process. Missing depth data then is replaced by input from other sources.

2.4.2-i Bilateral Filter

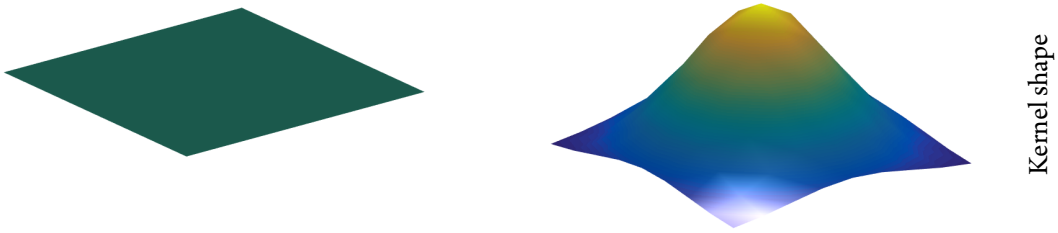
Since depth maps given by range cameras suffer from noise of variable magnitude (depending on object distance and the amplitude of the received signal), different denoising strategies could be applied. The simplest approach to smooth data would be to apply a mean filter. The mean filter is computing a local average within a kernel window (square kernel) centred at the pixel of interest. The weights of all kernel pixels are uniform. This process is also known as a 2-D convolution between the input image data and the mean kernel. The mean filter has been already a good filter to remove noise on data. However, since all kernel pixels contribute by the same strength to the filtered value, the mean filter is strongly deteriorating feature regions by rounding off sharp features. This is not acceptable if the method should preserve fine details on the processed data. The mean filter applied on pixel depth data $\mathcal{D}^t(\boldsymbol{u})$ is defined as:

$$\mathcal{D}^t_{\text{mean}}(\boldsymbol{u}) = \frac{1}{|\mathcal{N}(\boldsymbol{u})|} \sum_{\boldsymbol{u}' \in \mathcal{N}(\boldsymbol{u})} \mathcal{D}^t(\boldsymbol{u}'),$$

where $\mathcal{N}(\boldsymbol{u})$ represents the kernel window centred in $\boldsymbol{u}$.

A better filtering approach uses a Gaussian based convolution. The Gaussian filter is weighing each kernel pixel based on their Euclidean distance to the centre pixel. The weight is computed directly with a Gaussian function defined by a Euclidean distance parameter σ_d that regulated the width-radius size of the Gaussian. Note that σ_d is usually computed as the half-radius of the kernel window size. The Gaussian operates more robustly on edges compared to the mean filter, however, since this filter has no a priori knowledges of edges, it does not preserve fine details properly. Similarly to Equation 2.4.2-i, the Gaussian filter is defined as:

$$\mathcal{D}^t_{\text{gauss}}(\boldsymbol{u}) = \frac{1}{W_{\text{gauss}}} \sum_{\boldsymbol{u}' \in \mathcal{N}(\boldsymbol{u})} \mathcal{D}^t(\boldsymbol{u}')\ G_d(\|\boldsymbol{u} - \boldsymbol{u}'\|),$$



Mean filter Gaussian filter

Figure 2.4.2: The mean and Gaussian kernel shapes used to smooth data by 2-D convolution.

where $W_{\text{gauss}} = \sum_{\boldsymbol{u}' \in \mathcal{N}(\boldsymbol{u})} G_d(\|\boldsymbol{u} - \boldsymbol{u}'\|)$ and $G_d(\|d\|) = \exp\left(-\frac{\|d\|^2}{2\sigma_d^2}\right)$.

Figure 2.4.2 demonstrates the different kernel shapes of both mean and Gaussian filters. As discussed previously, the mean filter uses a constant function as kernel, whereas the Gaussian filter uses a Gaussian surface.

Since the filtering approach should smooth noisy data by keeping local fine structures, the denoising filter must have either incorporated an edge-aware smoothing (by previously detecting edges on the data) or either have an intrinsic formulation that preserves edges. The bilateral filter is still considered as one of the best choices for edge-preserving smoothing since it is easy to understand and quite efficient. The bilateral filter was first introduced by Tomasi et al. [TM98] and it consists of mixing two Gaussians to compute a proper weight for each kernel pixel. As the Gaussian filter, the weight is dependent on the Euclidean distance of the kernel pixel to the centre pixel. However, this weight is mixed with a second Gaussian weight based on the radiometric distance between the kernel pixel and the centre pixel. In this way, if the kernel pixel has a radiometric value which is too far from the centre pixel, the mixture weight will be low enough to not contribute to the filtered value. Using this simple combination of Gaussian functions, the bilateral filter becomes a really good candidate for smoothing data with edge-preserving feature. Similarly to Equation 2.4.2-i, the bilateral filter is defined as a product of two Gaussian functions by:

$$\mathcal{D}^t_{\text{bilat}}(\boldsymbol{u}) = \frac{1}{W_{\text{bilat}}} \sum_{\boldsymbol{u}' \in \mathcal{N}(\boldsymbol{u})} \mathcal{D}^t(\boldsymbol{u}')\; G_d(\|\boldsymbol{u} - \boldsymbol{u}'\|)\; G_r(\|\mathcal{D}^t(\boldsymbol{u}') - \mathcal{D}^t(\boldsymbol{u})\|),$$

where:

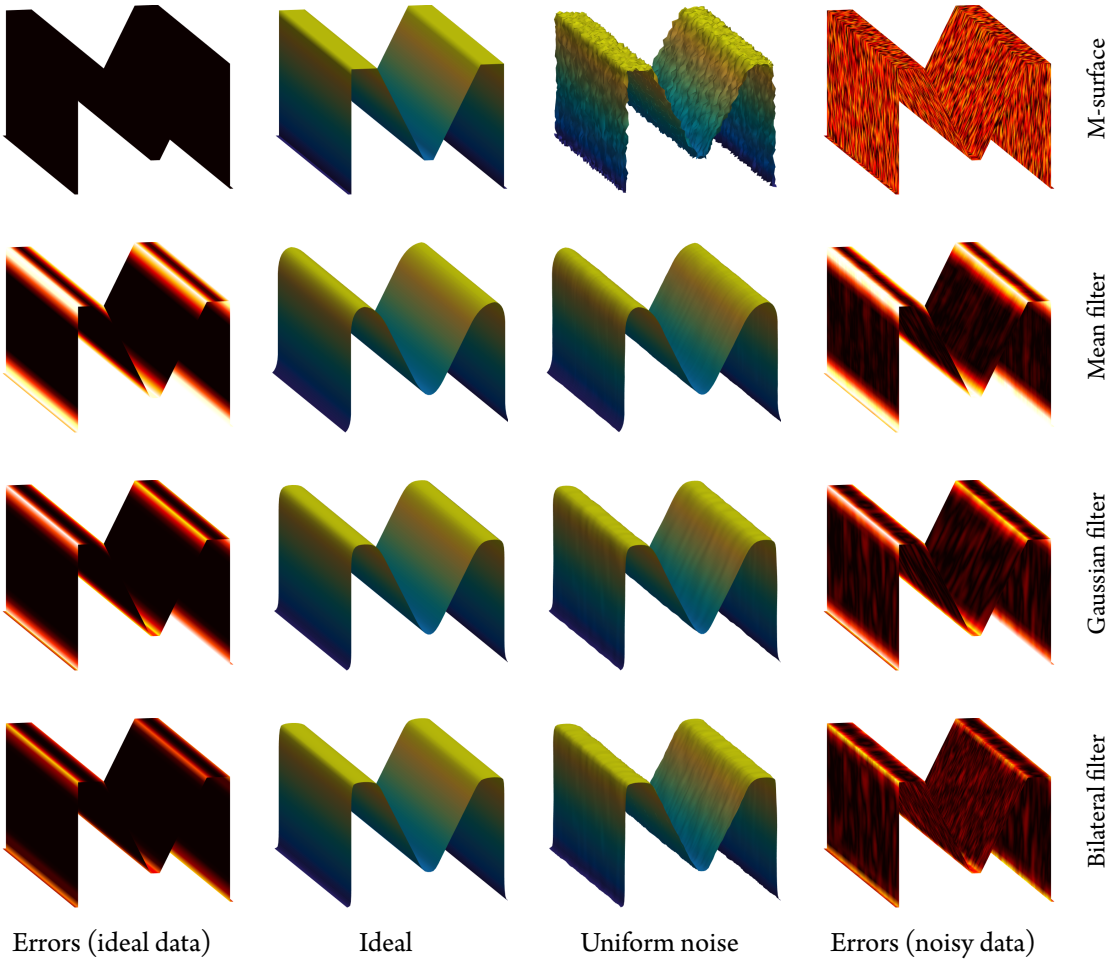


Figure 2.4.3: Comparison of different smoothing filters applied to M-shaped surface data. Two kinds of surface data are provided, one surface without noise and the other with uniformly distributed noise. The first and last columns show the difference between the ideal M-shaped data with the processed ideal data and the processed noisy data respectively.

- $W_{\text{bilat}} = \sum_{\boldsymbol{u}' \in \mathcal{N}(\boldsymbol{u})} G_d(\|\boldsymbol{u} - \boldsymbol{u}'\|)\, G_r(\|\mathcal{D}^t(\boldsymbol{u}') - \mathcal{D}^t(\boldsymbol{u})\|)$,
- $G_d(\|d_e\|) = \exp\left(-\frac{\|d_e\|^2}{2\sigma_d^2}\right)$,
- and $G_r(\|d_r\|) = \exp\left(-\frac{\|d_r\|^2}{2\sigma_r^2}\right)$.

Figure 2.4.3 shows a direct comparison between different smoothing filters applied on M-shaped surface.

The first row indicates the input data, left columns refer to the ideal surface data without any noise and the right columns represent the same surface data where uniformly distributed noise was added along the surface normal direction. The first and last columns show the colour-coded distance error between the ideal data with the processed ideal and processed noisy data respectively (minimum and maximum distances are in black and white colour respectively). The other rows represent three different filters (mean, Gaussian and bilateral). For each of the filters, the same kernel window size was used (7×7). The bilateral filter uses the same sigma as the Gaussian filter for the Euclidean distance $\sigma_d = 1.5$, whereas the radiometric sigma is set to $\sigma_r = 0.1$, which is applied to the depth value in the case of range data. This comparison clearly highlights the benefit of the bilateral filter against other filters. Even if the bilateral filter does not perform as well as the other filters on noisy data with homogeneous regions (such as the slopes of the M-shaped surface), all the edges are mostly preserved in comparison with the other filters that have tendencies to strongly smooth edges.

*It is easier to perceive error than to find truth, for the former lies on the surface and is easily seen, while the latter lies in the **depth**, where few are willing to search for it.*

Johann Wolfgang Goethe (∗1782 – †1832)

3

Depth Map Processing: Motion Artifact Compensation

Improving the accuracy of the input data used by algorithms is positively influencing the quality of their outputs. Due to their intrinsic principle, ToF data are subject to strong motion blur in dynamic environments. This chapter describes a new and fast method to compensate motion artifacts in ToF depth data.

3.1 Related Works

As stated in the previous chapter Section 2.3.2, motion artifacts occur in dynamic scenes at depth and reflectivity boundaries due to the sequential sampling of the correlation function. There are three (or arguably two) different approaches to reduce such artifacts. One way is by decreasing the number of frames obtained sequentially and needed to produce a valid depth reconstruction. As current two-tap sensors have different electronic characteristics for each tap, the raw values belonging to different taps cannot be combined without further calibration. In 3.1.1, a method proposed by Schmidt [Sch11] will be presented where each of these set of taps is dynamically calibrated, such that a valid measurement can be obtained with the bare minimum of 2 consecutive frames. Another approach commonly employed is composed of a detection step, where erroneous regions due to motion are found, followed by a correction step.

The methods presented in 3.1.2 differ in how these two steps are undertaken and in how much knowledge of the working principles is put into the system. The final approach proposed by Lindner and Kolb [LK09] is directly based on the estimation of scene motion between sub-frames using optical flow. This approach can be seen as an extension of the detect and repair approach, but as the detection is not only binary and the correction not only local it will be presented separately in 3.1.3.

3.1.1 Framerate enhancement

Current correlating pixels used in ToF cameras can acquire $Q = 2$ phase images simultaneously, shifted by 180° (i.e. π radians). N of these simultaneous measurements are made sequentially to obtain a sufficient sampling of the correlation function.

time	τ_0	$\tau_{\frac{\pi}{2}}$	τ_π	$\tau_{\frac{3\pi}{2}}$
tap 0	$\mathcal{T}_0^0$	$\mathcal{T}_0^1$	$\mathcal{T}_0^2$	$\mathcal{T}_0^3$
tap 1	$\mathcal{T}_1^2$	$\mathcal{T}_1^3$	$\mathcal{T}_1^0$	$\mathcal{T}_1^1$

Table 3.1.1: Illustration of raw frame $\mathcal{T}_{\text{tap index}}^{\text{phase index}}$ for $Q = 2$ taps and $N = 4$ acquisitions.

As shown by Erz et al. [EJ09, Erz11] these taps have different amplification characteristics, such that the raw values obtained from the taps cannot directly be used. Instead N must be chosen as 4 and the $\mathcal{A}^i$ used

in Equation 2.13 are calculated as:

$$\mathcal{A}^i = \sum_{k=0}^{Q-1} (-1)^k \, \mathcal{T}^i_k \tag{3.1}$$

For different intensity and depth, static sequences are obtained, and a linear model is fitted between $\mathcal{T}^i_0$ and $\mathcal{T}^i_1$. This model is then used to transform a phase image acquired by the first tap to the second one. In this way, the number of required phase acquisition is limited to 2. The full model with further extensions such as interleaved calibration can be found in [Sch11]. Note that this only reduces, but does not eliminate motion artifacts.

3.1.2 Detect and Repair Methods

Detect and repair approaches can be further categorised in methods that operate directly on the depth image [GYB04, LSHW07] and the methods that harness the relation between the raw data channels [Sch11, HHE11, HLCH12].

Filter-based methods

Gokturk et al. [GYB04] applied morphological filters on a foreground/background-segmented depth image to obtain motion artifact regions. These pixels are replaced by synthetic values using a spatial filtering process. Lottner et al. [LSHW07] proposed to employ data of an additional high-resolution 2-D sensor being monocularly combined with the 3-D sensor, effectively suggesting a joint filtering approach which uses the edges of the 2-D sensor to guide the filter.

Methods operating on raw data

Detection Schmidt [Sch11] calculates the temporal derivatives of the individual raw frames. Motion artifacts occur if the first raw frame derivative is near 0 (no change) whereas one of the other raw frames has a large derivative. This means that movement occurred between raw sub-frames. Lee et al. [LSKK12] operates on a similar principle evaluating the sums of two sub-frames.

Correction Finally, once regions with artifacts are detected, they need to be repaired. Here Schmidt uses the last pixel values with valid raw images, whereas Lee uses the spatially nearest pixel with valid data.

3.1.3 Flow-based Correction

So far, the detection step gave a binary output whether motion was present in a pixel. Subsequently some heuristic was applied to inpaint the regions with detected motion. Lindner and Kolb [LK09] took a somewhat different approach by loosening the requirement that the 4 measurements used for reconstruction need

to originate from the same pixel. Instead, the "detection" is done over the whole scene by estimating the optical flow between sub-frames. The application of optical flow to the raw data and the subsequent demodulation at different pixel positions require the following two points to be considered:

- Brightness constancy. Corresponding surface points in subsequent sub-frames should have the same brightness to be matched. This is not the case for the raw channels due to the internal phase shift between modulated and reference signal. Fortunately, in multi-tap sensors, the intensity (total amount of modulated light) can be obtained by adding up the measurements in different taps. Thus, the brightness constancy is given between the intensity of sub-frames:

$$y^i = \sum_{k=0}^{Q-1} \mathcal{T}_k^i. \tag{3.2}$$

  Note that recent ToF devices implement the two taps phase images subtraction directly in hardware, making the different tap measurements impossible to obtain.

- Pixel Homogeneity. The application of the demodulation at different pixel locations requires a homogeneous sensor behaviour over all locations. Otherwise artifacts will be observed, which usually cancel out by using the same pixel for all four measurements. Again, this is not the case for the raw channels due to pixel gain differences and a radial light attenuation toward the image border. To circumvent this, Lindner and Kolb [LK09] proposed a raw value calibration based on work by Stürmer et al. [SPH08].

Once the flow is known, it can be used to correct the raw image before applying the standard reconstruction formulas. The strength and weakness of this method are strongly coupled with the flow method used. It is important to obtain the correct flow, especially at occlusion boundaries, such that discontinuity-preserving flow methods should be preserved. Lindner and Kolb [LK09] reported a rate of 10 frames per second using the GPU implemented version TV-L1 flow proposed by Zach et al. [ZPB07] on a 2009 machine.

The following section describes a faster approach, proposed by Lefloch et al. [LHK13], based on the work of Lindner and Kolb [LK09].

3.2 The Fast Flow-based Correction Approach

Similar to Lindner and Kolb [LK09], a state-of-the art optical flow method is used to solve the image warping problem. The optical flow methods yield 2 displacement maps $\mathcal{F}_x$ and $\mathcal{F}_y$. It is a multi-level iterative minimisation process that leads to gradient consistencies and flow smoothness. The method computes optical flow on the normalised intermediate raw intensity phases $y^i = \mathcal{T}_0^i + \mathcal{T}_1^i$ as Equation 3.2. Since

channels $\mathcal{T}_0^i$ and $\mathcal{T}_1^i$ are measured with two different taps that have specific amplification characteristics, the normalisation is important to ensure reliable optical flow estimation.

This method needs to estimate two optical flows $\{\mathcal{F}_x^{0\to2},\ \mathcal{F}_y^{0\to2}\}$ and $\{\mathcal{F}_x^{1\to3},\ \mathcal{F}_y^{1\to3}\}$ between two different sets of normalised raw intensities $\{\widehat{y}^0, \widehat{y}^2\}$ and $\{\widehat{y}^1, \widehat{y}^3\}$, respectively; see Figure 3.2.1. This contrasts with the method proposed by Lindner and Kolb [LK09] that requires three optical flows. However, to achieve the final phase warping which expresses all raw images $\{\mathcal{T}_0^j, \mathcal{T}_1^j\}, j = 1, 2, 3$ into the same temporal reference as the acquisition time of the pair of frames $\{\mathcal{T}_0^0, \mathcal{T}_1^0\}$, a 3^{rd} optical flow needs to be derived from the two computed ones, i.e. from $\{\mathcal{F}_x^{0\to2},\ \mathcal{F}_y^{0\to2}\}$ and $\{\mathcal{F}_x^{1\to3},\ \mathcal{F}_y^{1\to3}\}$. Note that $\mathcal{F}_x^{i\to j} = -\mathcal{F}_x^{i\leftarrow j}$. To solve this final step, the optical flow $\{\mathcal{F}_x^{0\to2},\ \mathcal{F}_y^{0\to2}\}$ is assumed to be robustly computed and initialised by the following. Let $\boldsymbol{u}_p^0 = (x_p^0\ \ y_p^0)^\top$ be the 2-D coordinates of a pixel p in the channel pair $\{\mathcal{T}_0^0, \mathcal{T}_1^0\}$ and $\{\widehat{x}_p^i, \widehat{y}_p^i\}, i = 1, 2, 3$ the corrected 2-D coordinates of the same pixel in $\{\mathcal{T}_0^i, \mathcal{T}_1^i\}$. The correction of a pixel p is achieved through the application of a temporal polynomial function $\{T_p^x(t), T_p^y(t)\}, t \in [0, 3]$ which describes the final 2-D trajectory of p during the complete acquisition. This function is fitted to pixel coordinates $\{\overline{x}_p^i, \overline{y}_p^i\}$ that are corrected using interpolation between both computed optical flows. We define those coordinates as:

$$\begin{aligned}
\overline{x}_p^2 &= x_p^0 + \mathcal{F}_x^{0\to2}(\boldsymbol{u}_p^0),\\
\overline{x}_p^1 &= \frac{1}{2}\cdot\Big((x_p^0 + \frac{\mathcal{F}_x^{0\to2}(\boldsymbol{u}_p^0)}{2}) + (\overline{x}_p^2 - \frac{\mathcal{F}_x^{1\to3}(\boldsymbol{u}_p^0)}{2})\Big),\\
\overline{x}_p^3 &= \overline{x}_p^1 + \mathcal{F}_x^{1\to3}(\boldsymbol{u}_p^0),
\end{aligned} \tag{3.3}$$

note that the y-axis coordinate is computed using the same reasoning. From Equation 3.3, the temporal function $T_p^x(t) = a_x t^2 + b_x t + x_p^0$ is fitted by solving the following system of equations:

$$\begin{bmatrix} 1^2 & 1 & 1\\ 2^2 & 2 & 1\\ 3^2 & 3 & 1 \end{bmatrix} \begin{bmatrix} a\\ b\\ x_p^0 \end{bmatrix} = \begin{bmatrix} \overline{x}_p^1\\ \overline{x}_p^2\\ \overline{x}_p^3 \end{bmatrix}$$

This fitting function makes our correction more stable and compensate possible noise in the optical flow estimation.

The corrected coordinates are then used for the final warping of each raw image pair and are defined as:

$$\begin{aligned}
\widehat{x}_p^i &= T_p^x(i),\\
\widehat{y}_p^i &= T_p^y(i)
\end{aligned} \tag{3.4}$$

where $i \in \{0, 1, 2, 3\}$.

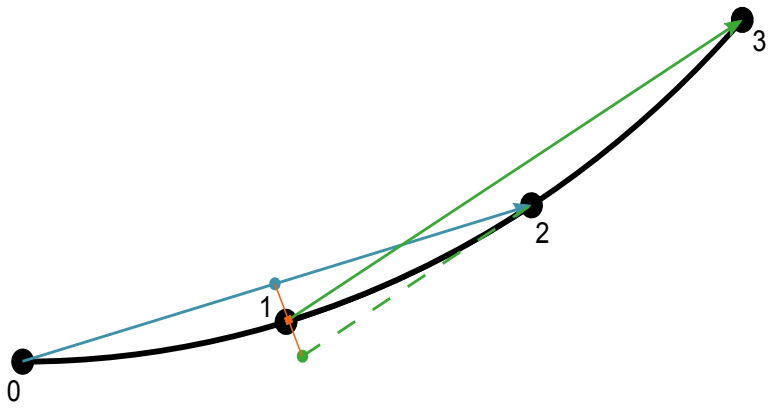


Figure 3.2.1: Representation of a constant-speed trajectory based on rotational motion.

To illustrate Equation 3.3, a simple circular trajectory at constant speed is shown in Figure 3.2.1. The four dots (0, 1, 2, 3) represent the sampling position of the trajectory. Blue and green arrows represent the flow $\{\mathcal{F}_x^{0\to2},\ \mathcal{F}_y^{0\to2}\}$ and $\{\mathcal{F}_x^{1\to3},\ \mathcal{F}_y^{1\to3}\}$, respectively. The orange square is the estimation of the position at $t = 1$ using both flow vectors (see the second equality of Equation 3.3).

Note that computing only one motion flow (for better performance) leads to an unsatisfactory quality of the motion correction in the case of complex motion. For this, the optical flow was only computed once using one pair of images $\{\widehat{y^0}, \widehat{y^2}\}$ leading to $\{\mathcal{F}_x^{0\to2},\ \mathcal{F}_y^{0\to2}\}$, and a linear relation was assumed to yield $\{\mathcal{F}_x^{1\to3},\ \mathcal{F}_y^{1\to3}\}$.

3.3 Results

This method was tested in a variety of scenes. To evaluate the robustness of the approach, two different data sets were generated and processed (see Figure 3.3.1). The middle column shows the polar depth images calculated by a ToF simulator [KK09] for the corresponding model at a specific camera position (4 phases sampled without any motion). The closer the distance, the darker the grey colour map is. Note that the ToF simulator was configured to generate depth data without any presence of noise to only evaluate the quality of the motion blur correction of the proposed method. Both models (left column) are approximately 3 meters away from the camera with a plane wall at 4 meters. For ground-truth purposes, a distance error is shown in the right column of the figure between the point cloud generated by the polar depth image and the original mesh. Since no motion is present in this data, the principal distance error is minimal.

Note the high distance errors on edges between background and foreground objects. The reason for this effect is common for all ToF cameras and is known as depth inhomogeneities or *flying pixels* (see previous chap-

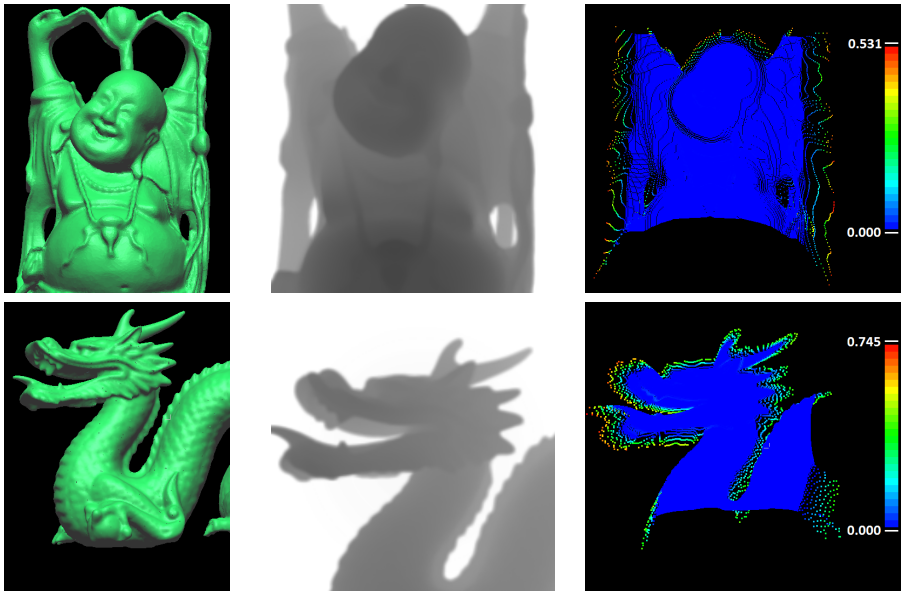


Figure 3.3.1: Two simulated depth data sets used for evaluation (the **Buddha** scene and the **Dragon** scene). The left column shows the rendered model; middle column represents the polar depth image generated by a ToF simulator camera in a static environment; and the right column shows the corresponding mesh-to-pointcloud distance.

ter Section 2.3.2). Like the error due to motion artifacts, this specific ToF sensor error is scene-dependent and occurs only on depth edges between foreground and background objects. In this experiment, the error is due to phase information mixing between the model and the wall leading to an error greater than 50 cm. It explains why high standard deviation errors are present in the following evaluations. Several methods have been already proposed to process these outliers (Section 2.4.2) but are not considered in the evaluation.

The evaluation of the static simulated data leads to mean distance errors of 1.0 cm ($\pm$ 5.3 cm) and 2.4 cm ($\pm$ 8.4 cm) for the **Buddha** and **Dragon** scenes, respectively.

Figure 3.3.2 shows the results applied to both simulated data sets. Each of the scenes was generated with a specific motion. The depth information of the **Buddha** scene was computed using a lateral camera motion during the complete phase acquisition. A camera drift of 1 cm was used for each of the phase sampling, which approximately leads to a 2 m.s^{-1} motion speed for a real camera setup (regarding a common *acquisition time* of 5 ms per raw phase); in contrast to the **Dragon** scene which was generated using a *yaw* rotation motion. A rotation angle drift of 1 degree was applied during the sampling of each phase and leads to a 200 $deg.s^{-1}$ angular velocity for a real camera setup. The depth data of the dynamic simulated scenes leads to a mean

	Error from ground-truth (cm) $(\mu \pm \sigma)$		
	Static	Dynamic	Corrected
Buddha	1.0 ± 5.3	5.3 ± 7.5	2.1 ± 5.2
Dragon	2.4 ± 8.4	5.6 ± 10.1	3.1 ± 8.6
	Error from static acquisition (cm) $(\mu \pm \sigma)$		
	Static	Dynamic	Corrected
Buddha	0 ± 0	1.4 ± 1.0	0.8 ± 0.7
Dragon	0 ± 0	1.4 ± 1.0	1.0 ± 0.9

Table 3.3.1: Statistics evaluation of the different scenes.

distance error of 5.3 cm ($\pm$ 7.5 cm) for the **Buddha** scene and 5.6 cm ($\pm$ 10.1 cm) for the **Dragon** scene. Whereas our raw phase warping method leads to mean distance errors of 2.1 cm ($\pm$ 5.2 cm) and 3.1 cm ($\pm$ 8.6 cm). Note the errors are significantly reduced (see the last two columns of Figure 3.3.2).

Table 3.3.1 provides a complete statistic of all our evaluations for simulated scenes. An additional piece of information is shown in this table (on the last column) which describes the mean and standard deviation distance errors of the dynamic acquisition (input depth in the presence of motion blur + the same depth corrected) in comparison to the static acquisition. This better highlights the robustness of the method since errors due to *flying pixels* are significantly reduced.

Finally, the robustness of the method on live sensor data is demonstrated using a moving *Person* scene. This last scene describes a meaningful application where a person is moving his hand rapidly in front of the depth camera. The PMD CamCube 3.0 is used which provides depth data at a rate of 30 Hz and the frame shown in Figure 3.3.3 corresponds to a rotational movement. Note how correct the moving region (i.e. hand) is reconstructed in both intensity and depth images.

All these results were obtained on a PC equipped with an *Intel* 8-core CPU and an *NVidia GeForce GTX* 480 GPU. For real time purpose, the entire correction was designed using the GPU CUDA development Toolkit. Regarding optical flow computation, a GPU implementation of a standard image processing library[1] was used.

[1] **OpenCV:** `http://opencv.willowgarage.com/wiki/`

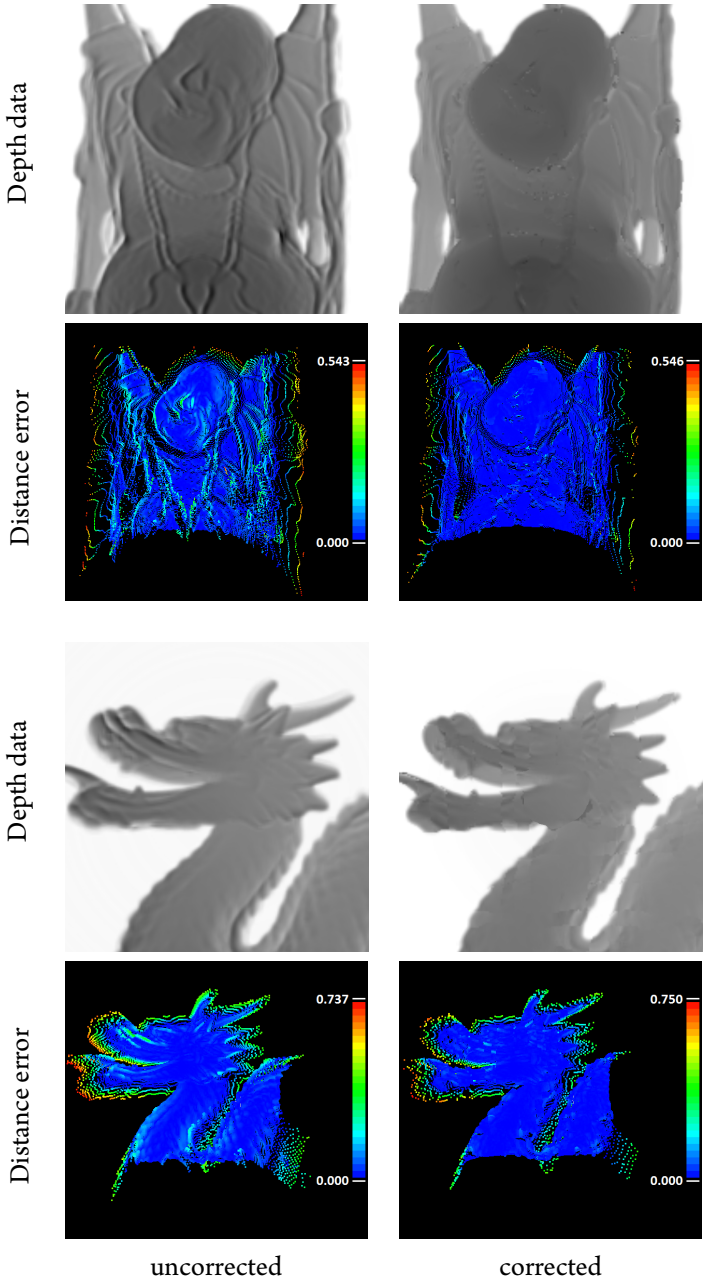


Figure 3.3.2: Evaluation of the proposed motion artifacts correction. The **Buddha** scene (2 first rows) and the **Dragon** scene (2 last rows) were generated with a lateral camera motion and a *yaw* camera motion, respectively. The odd rows represent the polar depth image computed in a dynamic environment and the even rows correspond to mesh-to-pointcloud distance.

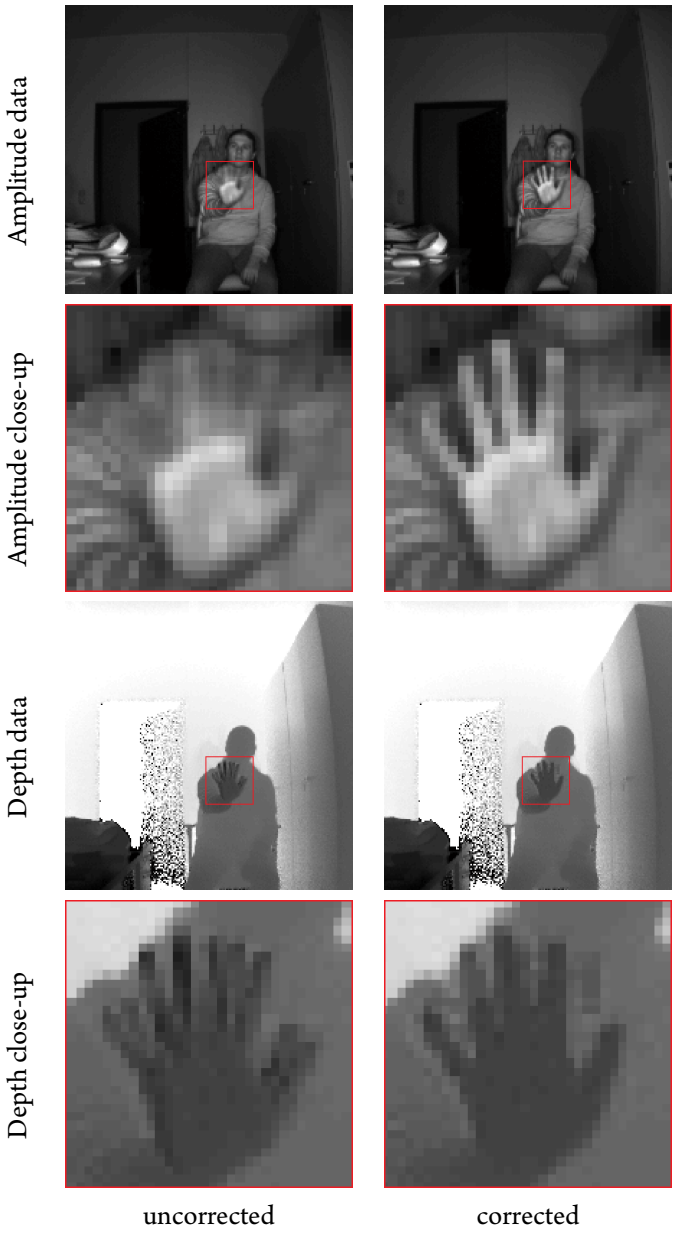


Figure 3.3.3: Motion artifacts compensation on the *Person* scene where the user rotates his right hand rapidly. The even rows are a close-up of the hand being mostly the dynamic region. The moving hand is well reconstructed in both intensity and depth ToF camera images.

3.4 Discussion

This chapter demonstrates the benefit of compensating motion artifacts. The robustness of this method was shown in the case of fast motion. This approach improved the quality of the phase-based ToF measurements. Such a system could be used to improve tracking or recognition of object of interest. For example, a hand gesture application would benefit from such a data correction system. However, the heavy computation of Optical Flow can still cause some performance issues.

Shortly after, Högg et al. [HLK13] proposed an approach to reduce the performance issue of optical flow computation by first segmenting region of images where motion occurs and, afterwards, to correct motion artifact using block-matching motion estimation method. This method reduces the total computation by at least half and provide even better results in terms of total distance error.

The presented algorithm can also be used to improve the accuracy of depth measurements which are directly given as input to 3-D reconstruction pipelines. The following chapters are focusing on online 3-D reconstruction applications.

There's something that ***3-D*** *gives to the picture that takes you into another land and you stay there and it's a good place to be...*

Martin Scorcese (∗1942)

4

Introduction to Online 3-D Reconstruction Methods using Range Data

Online 3-D reconstruction applications have attracted a lot of attention since the last half decade because of the availability of consumer range cameras. This chapter will introduce most of the state-of-the-art methods that enable high quality 3-D reconstruction in real time using range cameras focusing on the Point-Based Fusion approach (PBF) originally introduced by Keller et al. [KLL+13]. Note that methods that are designed for a single monocular camera will not be reviewed here since they are usually based on intensity-feature tracking, leading to very sparse reconstruction. Even if they share common principles with dense 3-D reconstruction methods, they belong to another branch of research named Visual Simultaneous Localization And Mapping (Visual SLAM) that mainly focuses on precise camera localisation

for huge environments. However, if this topic is any of interests, the reader is referred to the recent work of Engel et al. [EKC18] which is one of the best open-source approaches solving the complex Visual SLAM problem.

Furthermore, this chapter is meant to focus only on *real time* methods for reconstructing static scenes which was a basic constraint that drives this work. For a short description on methods solving 3-D reconstruction of non-rigid objects, refers to Section 4.1.2.

4.1 Overview

Given a stream of depth maps captured in real time by a commodity range sensor, an efficient scene representation is required to reconstruct in real time arbitrary scenes at different scales. Various interactive applications could benefit from obtaining reconstructions of arbitrary scenes in real time. A non-exhaustive list of such applications includes augmented reality (AR) where rendered graphic objects are realistically projected into the real world, autonomous guidance of robots, virtual reality where the motion of the person is directly mapped to the virtual world and direct feedback to the user once he is getting close to obstacles such as walls or tables, etc.

Online 3-D reconstruction systems use an incremental method to solve this challenging problem in real time. First, an alignment of consecutive input depth maps is done by estimating the camera ego-motion. Second, the overlapping data is fused into a single 3-D model/representation that is refined over time. The fusion is necessary for two reasons: one aspect is that the total amount of input data is big since such systems operate in real time; the other aspect lies in the intrinsic principle of range cameras which provide noisy data. The fusion phase allows a better handling of noise (using measurement redundancies) to deliver high quality reconstruction (super-resolution).

4.1.1 Related Works

There is a long history of research on 3-D reconstruction that started over more than three decades ago. However, systems that can densely reconstruct environments at a real time rate are available since less than one decade. Online methods require an incremental fusion of many overlapping depth maps into a single model representation that is continuously refined. To do so, methods track the ego-motion of the camera in order to align all depth maps to the same coordinate system. Figure 4.1.1 shows the common pipeline used by such methods.

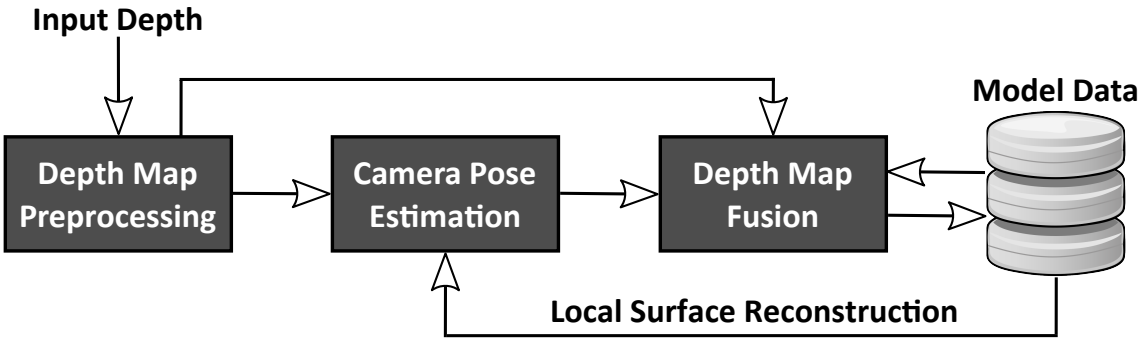


Figure 4.1.1: Main system pipeline of 3-D reconstruction methods.

Volumetric Fusion

With the availability of consumer range cameras (such as Kinect^{SL} the Xbox 360 version of the Kinect camera), online 3-D reconstruction methods providing high quality results arise (Newcombe et al. [NDI+11] and Izadi [IKH+11]). Their methods use a volumetric data structure to store samples of a continuous 3-D function (or implicit surface) [CL96]. Depth maps are first converted into signed distance functions and convexly averaged into a regular voxel grid. The extraction of the final surface is directly computed by the zero-level crossing of the implicit surface using raycasting or marching cubes polygonisation [LC87]. Both methods demonstrate real time high-quality reconstruction using a complete GPU implementations.

The major drawback of approaches based on the volumetric grid fusion is the memory overhead imposed by using such a regular voxel grid. Indeed, since depth data provides only the depth information of the surface of objects, accumulating this information to a volumetric grid will result in a huge amount of unusable empty voxels. The volumetric grid requires an initialisation of the volume space with its voxel resolution in a way that the bigger the volume space to reconstruct, the larger the voxel dimensions will be (lower resolution). In summary, these approaches fail to reconstruct large scale scenes without compromising quality.

To overcome this major problem, several strategies have been proposed. [RV12] and [WJK+12] avoid the volume space limitation by simply adopting a *moving volume* strategy. The volumetric grid is kept as the original Kinect-Fusion approach, but allows the volumetric data to move in the grid as soon as the distance between the camera centre and the origin of the volumetric grid is above a certain threshold. Once the moving volume operation is finished, implicit surface that is outside the volumetric grid is extracted as a dense point cloud and stream out on the CPU. Another strategy which avoids the waste of memory due to empty voxels has been proposed by Zeng et al. [ZZZL13] where a GPU-based octree is implemented (9- to 10-level octree) extending the original KinectFusion to larger reconstruction (an office reconstruction of $8\,\mathrm{m}\times$

$8\,\mathrm{m} \times 2\,\mathrm{m}$ is demonstrated) at the cost of computational complexity and pointer overhead. Whereas Chen et al. [CBI13] use an efficient $\mathbb{N}^3$ hierarchical and sparse data structure that enables interactive reconstruction of large volumes (e.g. volume of $8\mathrm{m}^3$ with $8\mathrm{mm}^3$ voxel resolution) and decouple the physical volume from the working set (similar to the moving volume approach). Even if the complexity is reduced in comparison to the GPU-based octree approach, this method still suffers from computational overheads due to the GPU-hierarchical data structure.

Niessner et al. [NZIS13] adopt a different strategy to avoid the problem of storage of empty voxels being still the most efficient approach. They use a voxel-based hashing data structure with custom hashing function. In this way, they avoid the need for a regular or hierarchical grid data structure leading to a very efficient accessing method of voxel data with large scale capabilities. Voxels far away from the camera centre are streamed out to CPU memory and conversely, voxels that get closer to camera centre are streamed back to GPU memory. This bidirectional streaming of voxels blocks occurs every frame at the pipeline starting point.

Height-map

The height-map representation is a compact and simple data structure which enables scalability and was first introduced by Gallup et al. [GPF10]. It is more suitable for modelling large planar regions such as buildings with floors and walls, since these appear as clear discontinuities in the height-map. More complex scenes can be supported such as balconies or arches by using multi-layered height-maps. However, this 2.5-D representation failed to represent more complex environments efficiently.

Point Fusion

These methods reconstruct their environment by simply merging overlapping data points. Points are defined as surface element (or surfel) that is basically similar to oriented discs. These methods are natural since they use a similar representation as the one directly given by the input depth data. Another positive point is that they enable an adaptive resolution compared to the traditional volumetric methods. The resolution of discs (disc radius) is directly linked with the perspective principle of the depth camera, meaning that the closer the camera is to an object the more points with smaller radius size the camera will provide for this object. Conversely, an object far from the camera will contribute to few surfels with big radii values. This contrasts with volumetric methods where noisy far objects are still reconstruct with a huge number of voxels.

Point fusion was first introduced by Weise et al. [WWLVG09] demonstrating high quality reconstruction of small-scale objects with sensor drift correction. Earlier, Rusinkiewicz et al. [RHHL02] introduced a point-based method to demonstrate in-hand online 3D scanning of small objects. However, an offline volumetric approach (from [CL96]) was used to improve their reconstruction quality indicating than point

based method were still far behind from volumetric approaches in term of reconstruction quality. Henry et al. [HKH+12] has introduced a point-based method for large scale reconstruction (e.g. entire indoor buildings) with camera drift correction (using loop closure detection and bundle adjustment). However, their method suffers drastically from overhead computations and their frame-rate was limited to 3 Hz. This is a huge trade-off between scale versus efficiency.

The point-based fusion method gains recently more interests due to the work proposed by Keller et al. [KLL+13] that demonstrates high quality reconstruction of large scale environments using a simple unsorted vector of augmented points without the need for any spatial data structure. This method will be explained in detail in Section 4.1.6. While achieving efficient performance and scalability for large scenes, the reconstruction quality is not exactly as good as the quality achieved by volumetric methods. See Chapter 5, for a new method that drastically reduces this gap.

Salas-Moereno et al. [SMGKD14] improved the original PBF approach from Keller et al. [KLL+13] and reduced the total amount of surfels by detecting large planar regions. This method is specially designed for the reconstructions of an office composed of several large planar sections.

4.1.2 Beyond "Basic" 3-D Reconstructions

Recently, different approaches focussed on high quality 3-D reconstruction of non-rigid objects. Even this challenging problem is out of the scope of this dissertation, a short discussion and related works on this recent topic will be given. To achieve high quality reconstruction of non-rigid objects, most of the proposed approaches are using either a skeleton or a template model. This model is then used to directly warp and fuse it to the pose reference of the input frame. Zollhöfer et al. [ZNI+14] use a reference model of the non-rigid object at a static pose. Whereas Newcombe et al. [NFS15] uses a volumetric model that is continuously refined with new input depth. Both methods use a reference model that is fitted to the model pose of the new incoming data. There are two major drawbacks for both these approaches. First, due to the use of a model reference, they cannot properly handle major changes in shape and topology. Secondly, these systems match correspondences between the current pose of the model reference and the new incoming pose by assuming small frame-to-frame motion.

Similarly, Innman et al. [IZN+16] introduced very recently an approach that creates the model reference directly from the scanning process and does not require any static acquisition [ZNI+14]. A volumetric representation is chosen to encode the surface geometry as well as the non-rigid space deformation. To maintain a reliable alignment, this method uses a volumetric regularisation based on the "As Rigid As Possible" (ARAP) surface modelling [SA07] coupled with a sparse colour feature tracking. They demonstrate better real time quality reconstruction than the method proposed by Newcombe et al. [NFS15].

Dou et al. [DKD+16] achieves outstanding result using multiple RGBD-live views which differs from previous approaches. They introduced a new real time pipeline for live performance capture that generates temporally coherent high-quality reconstruction. The huge advantages of this approach are that no prior assumption is made regarding the capture scene and it does not require the use of a skeleton or template model. This is by far the real time method that produces the highest reconstruction quality of non-rigid object at this time, however, it requires multiple sensors and thus, multiple GPUs to handle the extreme processing demands.

4.1.3 Estimation of Surface Attributes

Since range cameras provide as input a noisy depth map, different pre-processing steps are required to properly solve the online 3-D reconstruction problem. A 2-D pixel is denoted as $\boldsymbol{u} = (x, y)^\top \in \mathbb{R}^2$. $\mathcal{D}^t \in \mathbb{R}^{w \times h \times 1}$ is the raw depth map at time frame t, w and h being the width and height of the map respectively. For the following, w and h will be omitted to simplify the map notations. First, a complete 3-D information is extracted from the depth map using the intrinsic matrix $\boldsymbol{K}$ of the camera. From the resulting 3-D point-cloud (or vertex map $\mathcal{V}^t \in \mathbb{R}^3$), surface normals $\mathcal{N}^t \in \mathbb{R}^3$ can be easily extracted. Normals are crucial surface attributes for 3-D reconstruction methods. They are mainly used to build reliable correspondences between input and model data and enable realistic rendering using the phong illumination for example. Refer to the appendix Section A.1 for more details on the estimation of surface attributes from an input depth and to Chapter 5 for additional surface attributes extraction that leads to better 3-D reconstruction methods if used appropriately.

4.1.4 Camera Pose Estimation

To solve the complex problem of 3-D reconstruction, all input range data must be expressed in the same coordinate system (known as world coordinates, usually initialised by the first camera coordinates of the input sequence). Thus, for each input range, the system tracks the position and orientation of the camera. Note that this thesis will only describe the estimation of the 6-DoF rigid homogeneous transformation $\boldsymbol{T}_{4\times4}$.

Most real time 3-D reconstruction systems solve the camera tracking problem using the well-known ICP algorithm [BM92, CM92] that was originally designed to register two arbitrary sets of point clouds (source and target). However, due to the data structure of depth images (organised as a grid), these systems use a variation of the original ICP algorithm called the perspective ICP. Clearly, if the camera motion between two consecutive frames is small, the intensive computation of correspondence finding can be reduced to a simple projection of the input point expressed in the model coordinates. Analogously, the correspondence between the source vertex map $\mathcal{V}^t$ (transformed into the model frame $\mathcal{V}^{t\rightarrow(t-1)}$) and the model vertex map

$\mathcal{V}_{\mathcal{M}}$ is located at similar 2-D image coordinates. Any source point $\mathcal{V}^t(\boldsymbol{u})$ finds its model correspondence $\mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)$ by a simple projection given by the intrinsic matrix $\boldsymbol{K}$ where $\boldsymbol{u}^* = \boldsymbol{K}\ \boldsymbol{T}^{t\to(t-1)}\ \mathcal{V}^t(\boldsymbol{u})$.

The ICP output is the 3-D rigid transformation that transforms the source points in such a way that the total error between the correspondences set is minimal. In [BM92, CM92], the point-to-point error metric was chosen to introduce the concept of the ICP algorithm. This error metric is defined as the Euclidean distance between each pair of points (built by a transformed source point and its corresponding target point):

$$e_{\text{point}}(\boldsymbol{u}) = \left\|\boldsymbol{T}^{t\to(t-1)}\mathcal{V}^t(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)\right\|^2. \tag{4.1}$$

The same year, Chen et al. [CM92] plugged to the ICP algorithm another error metric called the point-to-plane error metric. It is defined as the distance between each pair of points projected to the tangent plane described by the target point:

$$e_{\text{plane}}(\boldsymbol{u}) = \left\langle \boldsymbol{T}_l^{t\to(t-1)}\mathcal{V}^t(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)\,,\mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^*)\right\rangle^2. \tag{4.2}$$

Other hybrid error metrics can be found in the literature such as the one firstly introduced by Kerl et al. [KSC15] which consists of coupling point-to-point error metric with colour error metric (having both different scales). Concerning the colour based ICP algorithm, most of the methods have been focusing only on building the set of correspondences based on colour (or intensity) information but do not purely integrate colour on the objective function. The first colour based correspondences search was introduced by Johnson and Kang [JK99].

Recent methods use the approach introduced by Newcombe et al. [NDI+11] to estimate the camera ego motion. Newcombe et al. [NDI+11] demonstrates that a frame-to-model based registration is more accurate than the direct frame-to-frame based registration. The frame-to-frame based registration, simply finds the best transformation between two input frames (that are both noisy), whereas the frame-to-model based registration is computing the best transformation between the current reconstructed model and the new input frame (see Section 4.1.5 for more details on the model reconstruction). To speedup the process of retrieving the best transformation between two consecutive frames, Newcombe et al. [NDI+11] uses a coarse-to-fine approach where different scales of each required maps is computed ($\mathcal{V}_l^t$ and $\mathcal{N}_l^t$ where $l \in \{0,1,2\}$ and $l = 0$ denotes the original input resolution). This approach is known as the hierarchical ICP and is implemented in many KinectFusion-like methods [KLL+13, NZIS13, LWK15, LKS+17].

Several variants of the ICP are available in the literature but these methods are generally decomposed into two main steps, the correspondence search and an error minimisation. For a detailed explanation of the ICP algorithm, please refer to the appendix Section A.2

4.1.5 Depth Map Fusion

Online 3-D reconstruction systems fuse the new incoming input depth map to their respective global representation. The global fusion of all depth maps seen from different viewpoints acts as a de-noising filter. Another reason of applying a global fusion is simply due to memory efficiency; low-cost range cameras usually operate (at least) at 30 Hz and giving, for the best of them, an image resolution of 640×480 (VGA) leading to over 9 million points per second. It is a huge amount of data and if such systems want to maintain interactive processing, they require to have a global fusion phase.

Even if different representations imply different approaches on how to find correspondences between input point and model data, all techniques use a simple convex averaging that is converging to the mean measurement. For each singular particle of the global model representation, a weight (also known as confidence counter) is stored. Practically, this weight encodes a function of how many times this singular particle is seen from the camera. The bigger the weight the higher the precision of the local surface reconstruction. Using the convex averaging approach for fusion, the bigger the weight, the lower the influence of the new input point on the merged result. It is also common to give different weights for different input data. For clarity, let's assume that the current model data $\mathcal{F}_{\mathcal{M}}^{t-1}(\boldsymbol{p})$ is merged with the new incoming measurement $\mathcal{F}^{t}(\boldsymbol{p})$ both having a respective weight of $\mathcal{W}_{\mathcal{M}}^{t-1}(\boldsymbol{p})$ and $\mathcal{W}^{t}(\boldsymbol{p})$. Then, the convex averaging that occurs during one iteration of the depth fusion is defined as:

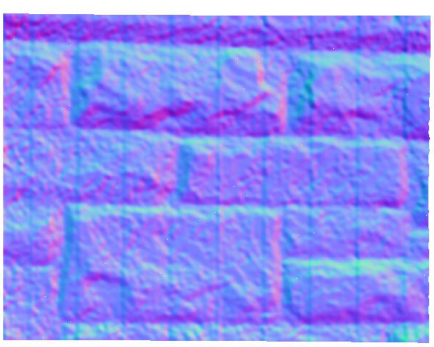
$$\begin{aligned} \mathcal{F}_{\mathcal{M}}^{t}(\boldsymbol{p}) &= \frac{\mathcal{W}_{\mathcal{M}}^{t-1}(\boldsymbol{p})\,\mathcal{F}_{\mathcal{M}}^{t-1}(\boldsymbol{p}) + \mathcal{W}^{t}(\boldsymbol{p})\,\mathcal{F}^{t}(\boldsymbol{p})}{\mathcal{W}_{\mathcal{M}}^{t-1}(\boldsymbol{p}) + \mathcal{W}^{t}(\boldsymbol{p})} \\ \mathcal{W}_{\mathcal{M}}^{t}(\boldsymbol{p}) &= \mathcal{W}_{\mathcal{M}}^{t-1}(\boldsymbol{p}) + \mathcal{W}^{t}(\boldsymbol{p}). \end{aligned} \tag{4.3}$$

Figure 4.1.2 shows the advantage of applying the convex averaging during the depth fusion stage over the single input data using the **StoneWall** dataset from [ZK13]. The first row shows both normal maps (input and model from the same viewpoint). Note how the surface normal map of the global model is much smoother and provide a lot of fine structures compared to the single noisy normal map given by the input depth.

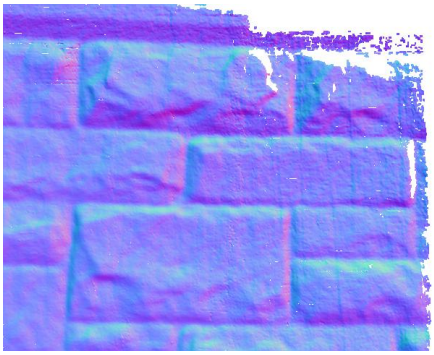
4.1.6 The Point-Based Fusion Approach

PBF approaches use as model an unsorted vector of oriented points and are implementing the following steps:

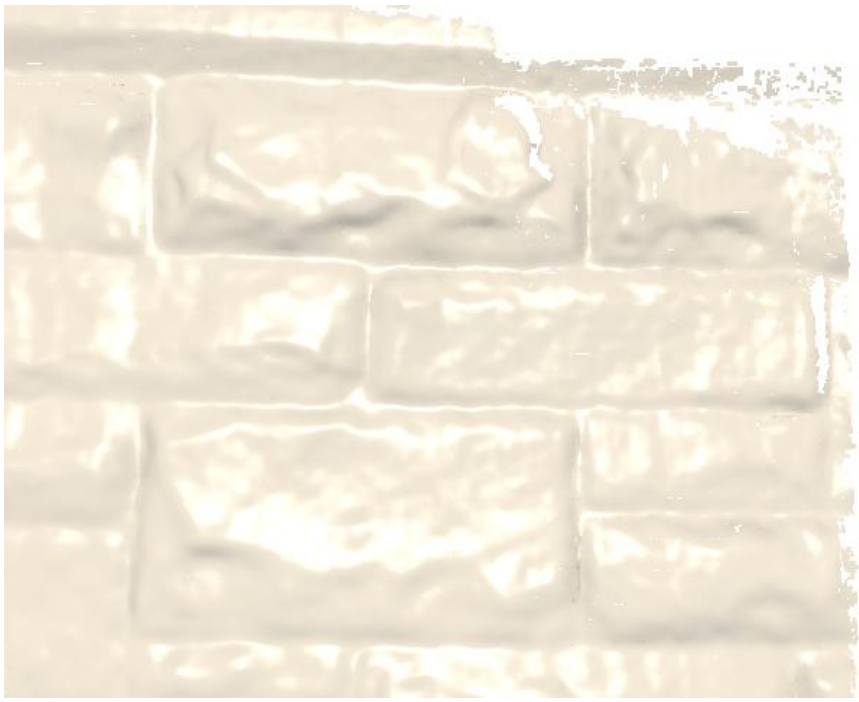
Depth Map Pre-processing From the input vertex and normal maps, a radius map ($\mathcal{R}^{t} \in \mathbb{R}$) is computed as proposed by Weise et al. [WWLVG09]. Large radii given by points seen from an oblique view are



Input normal map



Model normal map



Phong illumination

Figure 4.1.2: Depth map fusion in action using the **StoneWall** dataset at frame 473 (provided by the approach of Zhou et al. [ZK13]). The missing model data (region top-right) is due to the small amount of merged data (confidence counter is not yet high enough to be rendered).

clamped (exceeding 75°). The radius is simply a function of the z-distance of the point and the orientation of its surface normal. It is the back-projected length of the half-pixel diagonal weighted by the angle between

the local surface normal and the viewing direction of the camera.

Depth Map Fusion Given a valid camera pose, input points are fused into the *global model*. The global

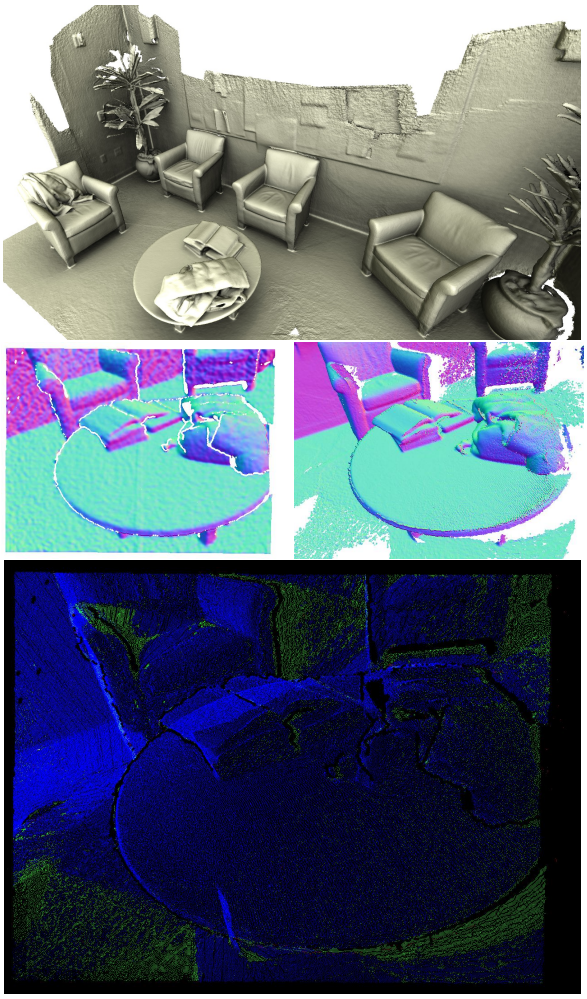


Figure 4.1.3: Visualisation of the *index map* originally proposed by Keller et al. [KLL+13]. The first row represents an overview of the **lounge** data set given by [ZK13]. The second row represents both input and model normal maps ($\mathcal{N}^t$ and $\mathcal{N}_{\mathcal{M}}$) at frame $t = 1441$. The last row is the *index map* where each coloured pixel represents an index in the vertex buffer for the corresponding model points. Blue colour represents indices of *stable* model points, whereas green and red colours represent indices of *unstable* or removal model points respectively. A cyan colour means that both *stable* and *unstable* model points are projected into the image plane on the same pixel.

model is an unsorted vertex buffer of 3-D points with associated attributes. Points evolve from *unstable* to *stable* status based on their confidence. Data fusion first *projectively associates* each point in the input depth

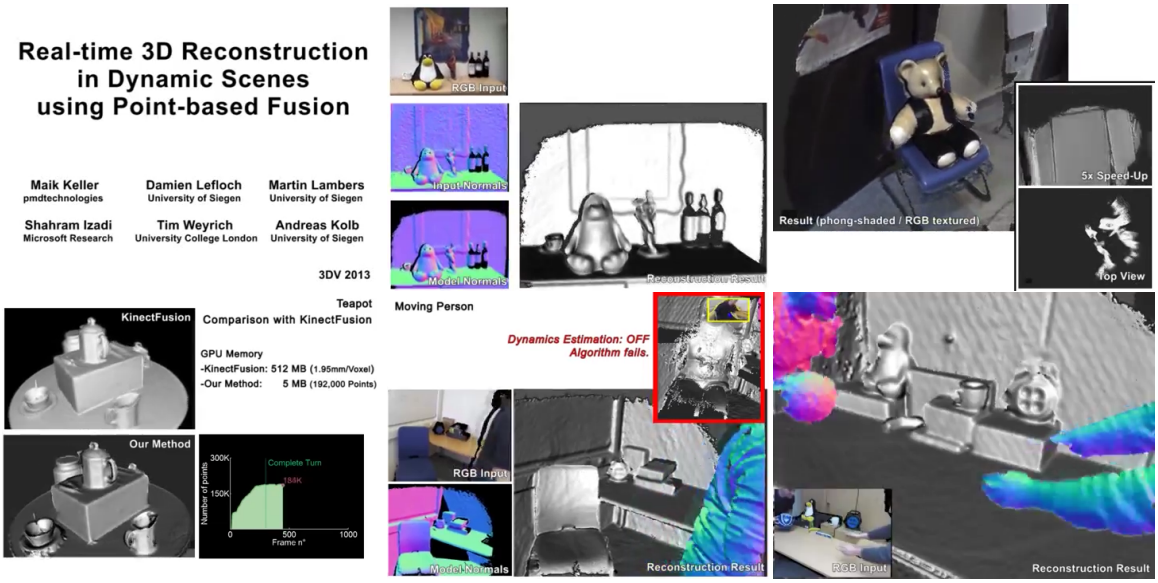


Figure 4.1.4: Still images extracted from the video results of the original point-based fusion presented by Keller et al. [KLL+13].

map with the set of points in the global model, by rendering the model as an *index map*. Figure 4.1.3 shows an example of such index map for the **lounge** data set given by Zhou and Koltun [ZK13] at frame 1441. Note that this *index map* is slightly different than the one proposed originally by the PBF approach [KLL+13]. It is a *deep index map* where different layers of indices are rendered depending on the intrinsic nature of the corresponding model points (see the following chapter Section 5.7). If corresponding points are found, the most reliable point is merged with the new incoming point using a weighted average. If no reliable correspondence pair exists, the new input point is added to the global model as an unstable point. The global model is cleaned up over time to remove outliers due to visibility and temporal constraints.

The reader is invited to check the following link[1] presenting the original PBF method [KLL+13]. This video (see Figure 4.1.4 for a thumbnail) shows the advantages of using the PBF approach which maintains high quality reconstruction even at a larger scale, and continuously adds new local surface reconstruction as long as GPU memory is available. It provides adaptive resolution (further objects are modelled with fewer surfels with bigger radius size than the ones given by closer objects). The adaptive resolution is intrinsically handled due to the perspective principle of depth cameras.

[1]Video results of Keller et al. [KLL+13]: https://www.youtube.com/watch?v=2BdwMdh5M7Q&t=11s.

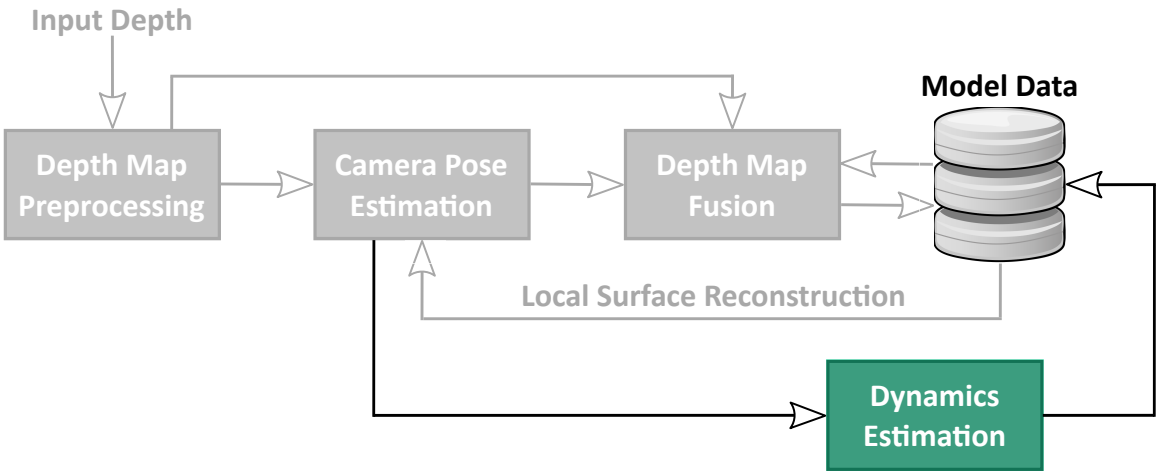


Figure 4.2.1: Updated 3-D reconstruction pipeline with dynamic environment support.

4.2 Dynamic Environments

Most of the online 3-D reconstruction approaches have limited support for dynamic objects. Due to the continuous convex averaging during the data fusion stage, dynamic objects do not gain enough confidence to strongly contribute to the global representation. However, this limited support of dynamics could cause camera tracking failure on specific scenes which was a focus of this work.

This thesis extends the original KinectFusion approach [NDI+11] by automatically segmenting dynamic subjects in the scene, in order to clear-up the global representation rapidly and support robust camera tracking. Initially, moving objects are classified as outliers during the ICP correspondences search. Given these initial dynamic areas, a multi-scale region growing procedure is applied to properly segment moving objects. Moving regions are excluded from the camera pose estimate, and their corresponding points in the global model are reset to *unstable* status, leading to a natural propagation of scene changes into our depth map fusion.

Figure 4.2.1 shows the updated 3-D reconstruction pipeline that properly handles dynamic environments. The *washed* colour refers to the original pipeline modules. The dynamic estimation is first initialised by the output of the camera pose estimation module and later on used to update the model representation accordingly.

4.2.1 Segmentation

Izadi et al. [IKH+11] observe that failure of data association during the ICP phase is a strong indication that these input points may belong to dynamic objects. We build our dynamic segmentation upon that observation and retrieve this information via the ICP status map $\mathcal{S}$ that encodes a status flag for each input depth samples directly resulting from the ICP correspondences search. Four different status flags are given:

`no_input`:	$\mathcal{V}^t(\boldsymbol{u})$ is invalid or missing.
`no_cand`:	No stable model points $\mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)$ in the proximity of $\mathcal{V}^t(\boldsymbol{u})$.
`no_corr`:	Stable model points in proximity of, but no valid ICP correspondence for $\mathcal{V}^t(\boldsymbol{u})$ (the model points did not pass both conditions composed of a Euclidean distance and a normal angle divergence).
`corr`:	Otherwise ICP found a correspondence.

Input points marked as `no_corr` are used as an initialisation of the proposed segmentation method based on a hierarchical region growing algorithm. The use of region growing is justified by the necessity to segments complete moving objects as dynamic even if only some parts of them move (like a static person that only moves his arms). This high-level view on dynamics drastically improves the limited handling in previous approaches as the one proposed in [IKH+11]. As output, the proposed algorithm segments the current input frame into two classes (static and dynamic points) stored in a *dynamics map* $\mathcal{X}^t$.

The current system renders two sets of augmented points: the global model points representing the high resolution reconstruction of the background (static) environment; and the set of input points marked as dynamic representing the segmented moving object of $\mathcal{X}$ (see Figure 4.2.2-bottom-centre).

Hierarchical Region Growing The goal is essentially to find connected components in $\mathcal{V}^t$. To do so, a region growing based on the similarity of point attributes is performed. More precisely, data consistency of point position ($\mathcal{V}^t$) and local surface normal ($\mathcal{N}^t$) are compared using a 4-connected component neighbourhood. Seed points are extracted and marked as dynamic in $\mathcal{X}^t$ and points, whose position and normal are within given thresholds, are iteratively added. For more details refer to Algorithm 1.

Since the speed of region growing is directly linked to the input data resolution and the size of the objects to grow, the algorithm starts with a down-sampled map $\mathcal{X}_2^t$, and repeatedly up-sample until reaching $\mathcal{X}_0^t = \mathcal{X}^t$. Note that the pyramidal hierarchy of the input maps is already built during the camera pose estimation phase (maps $\mathcal{V}_l^t$ and $\mathcal{N}_l^t$ with $l \in [0, 1, 2]$).

The robustness against camera noise and occlusions is improved applying a morphological erosion with a circle-shape structured element of 1-pixel size radius at the coarsest pyramid level $\mathcal{X}_2^t$. The erosion is

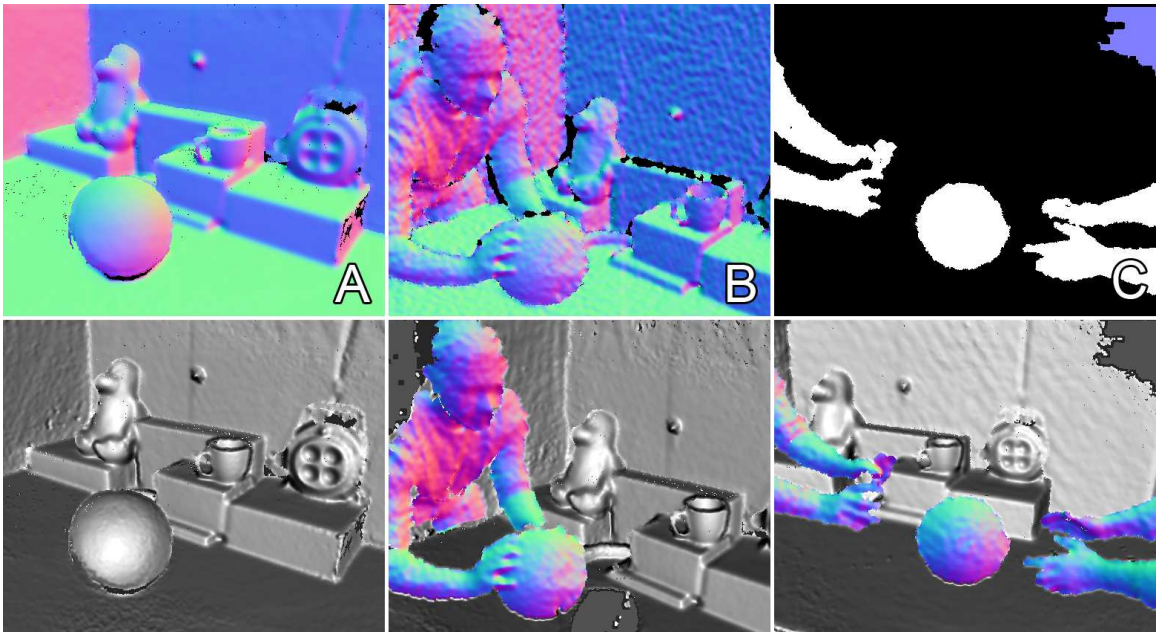


Figure 4.2.2: Segmentation of moving objects. A: global model coloured with surface normals; B: raw input data of the previously static ball being picked up; C: segmentation of dynamic parts; Bottom row: reconstructed result (model points + dynamic parts).

applied after the initialization of the dynamic map $\mathcal{X}_2^t$. This ensures that $\mathcal{X}_2^t$ covers only the inner region of dynamic objects (see Figure 4.2.2-C for an example of the dynamic object segmentation). The drawback of this method is that it only detects dynamic objects in regions where global model points are present. Indeed, model points must be in proximity of the input point in order to be flagged as `no_corr` by the ICP phase. Another drawback of this method is that it may classify fine structure objects (e.g., a pen) as dynamic if it lies in front of a dynamic region. In fact, during the creation of the next level dynamic segmentation, the coarse level is assumed to be fully correct which is a strong assumption that can be wrong if a fine structure was not visible at this resolution. However, this problem is rare and could be fully avoided by introducing a consistency check when creating the next level of the segmentation map. Figure 4.2.2-C shows an example of dynamic segmentation where region growing does not apply (blue-purple colour) due to the lack of model points in this region.

4.2.2 Dynamic-aware Model Updates

Previously, the proposed method demonstrates that dynamic objects can be fully segmented and used as a direct feedback to the user during the rendering stage. In addition, the segmentation of moving objects can

Algorithm 1: Dynamics segmentation via a hierarchical region growing approach (L refers to the highest level of the pyramid).

Input: $\mathcal{S}^t, \mathcal{V}^t_{0\dots L}, \mathcal{N}^t_{0\dots L}, \delta_{\text{dist}}, \delta_{\text{norm}}$
Output: $\mathcal{X}^i_0$

// Initialise Segmentation in lowest resolution
foreach *pixel* $\mathbf{u}$ *in level* L ***inparallel*** **do**
  $\boldsymbol{u}' \leftarrow 2^L\ (\boldsymbol{u})$
  $\mathcal{X}^t_L(\boldsymbol{u}) \leftarrow \mathcal{S}^t(\boldsymbol{u}') == \texttt{no\_corr}$
$\mathcal{X}^t_L \leftarrow \text{morphErosion}(\mathcal{X}^t_L)$
// Multi-scale region growing approach
for $l = L$ *to* 0 **do**
  repeat
    foreach *pixel* $\mathbf{u}$ *in scale level* l ***inparallel*** **do**
      // If dynamic
      if $\mathcal{X}^t_l(\mathbf{u}) \neq 0$ **then**
        // 4-Neighbour connectivity
        foreach *pixel* $\mathbf{u}_n \in \mathcal{N}(\mathbf{u})$ **do**
          if $\mathcal{X}^t_l(\mathbf{u}_n) = 0$ **then**
            if $\|\mathcal{V}^t_l(\mathbf{u}_n) - \mathcal{V}^t_l(\mathbf{u})\| \leq \delta_{depth} \wedge \angle(\mathcal{N}^t_l(\mathbf{u}_n), \mathcal{N}^t_l(\mathbf{u})) \leq \delta_{norm}$ **then**
              $\mathcal{X}^t_l(\boldsymbol{u}_n) \leftarrow \mathcal{X}^t_l(\boldsymbol{u})$
  until *No region to grow*
  if $l \geq 1$ **then**
    $\mathcal{X}^t{}_{l-1} \leftarrow \text{buildNextPyrLevel}(\mathcal{X}^t_l, l-1)$

also be used to update the current model rapidly. For example, if a user wants to reconstruct several objects on a desk and if he is later changing the configuration of some objects on the desk, then, previous approaches (such as [IKH+11]) will only handle this case via the convex averaging. Meaning that a certain amount of time is required for the convex averaging to update the "ghost" region where the moved object was previously located. Having a high-level notion of dynamics enables to better handle these scenario and directly remove the moving objects of the model.

During the depth map fusion stage, a model point $\mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)$ (with a certain confidence value $\mathcal{W}_{\mathcal{M}}(\boldsymbol{u}^*)$) that is averaged with a dynamic input point $\mathcal{V}^t(\boldsymbol{u})$ is demoted to an unstable point using the following rule:

$$\text{if} \quad \mathcal{X}^t(\boldsymbol{u}) \ \wedge \ \mathcal{W}_{\mathcal{M}}(\boldsymbol{u}^*) \geq c_{\text{stable}} + 1 \quad \text{then} \quad \mathcal{W}_{\mathcal{M}}(\boldsymbol{u}^*) \leftarrow 1 \tag{4.4}$$

In this way, the model is immediately updated due to the state change from static to dynamic. The offset

of $+1$ in Eq. (4.4) is only ensuring that any dynamic point that sufficiently gained enough confidence (potentially because it is again static) can be rendered as stable global model for at least one iteration; otherwise, an object that has once been classified as dynamic would never be able to be stable again, as it would always be inconsistent with the model.

To highlight the benefit of immediate clearing up the global model from complete segments of moving objects, the *Moving Person* scene is shown. In this scene, Figure 4.2.3, the person first sits in front of the camera and is reconstructed before moving out of view. Since the moving person occupies much of the field of view, leaving only few reliable points for the ICP algorithm, the camera tracking fails with previous approaches. However, the presented system fully segments the moving person (Figure 4.2.3-A) and removes it from the model (Figure 4.2.3-B) leading to a robust camera tracker in dynamic environments.

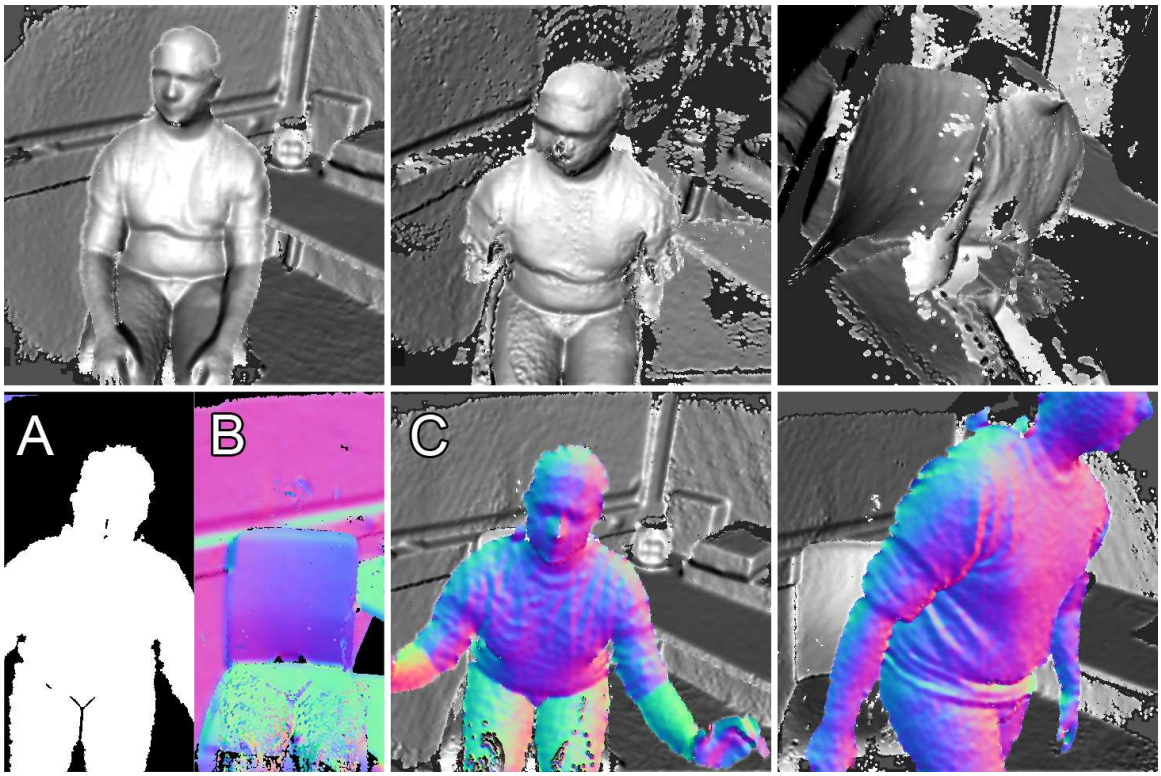


Figure 4.2.3: The *Moving Person* scene. A person sits on a chair, is reconstructed, and then moves. Dynamic parts occupy much of the field-of-view and cause ICP errors with previous approaches (top row). Segmenting the dynamics (A) and ignoring them during pose estimation (B) allows increased robustness (bottom row).

4.3 Discussion

This chapter focused on giving a complete introduction of online 3-D reconstruction systems. Several model representation methods have been proposed since the first release of the original Kinect-Fusion [NDI+11], however, they all share common modules. The method proposed by Keller et al. [KLL+13] is a good candidate to solve the challenging problem of online 3-D reconstruction due to its simplicity, its adaptive resolution and its scalability. Additionally, this chapter demonstrates the importance of segmenting dynamic objects to achieve a robust ego motion estimation.

The two following chapters (Chapter 5 and Chapter 6) will present the remaining contributions of this thesis to the online 3-D reconstruction topic.

*It's a brilliant surface in that sunlight. The horizon seems quite close to you because the **curvature** is so much more pronounced than here on earth. It's an interesting place to be. I recommend it.*

Neil Armstrong (∗1930 – †2012)

5

Curvature-aware Point-Based Fusion

In this chapter, a method that drastically improves the full 3-D reconstruction pipeline using an additional surface characteristic will be presented. This surface characteristic, i.e. curvature, stored as a point attribute, helps to build robust correspondences during the camera tracking phases

leading to better camera tracking on low-depth feature scenes and reducing the overall camera drift. Furthermore, this chapter will show that the curvature can be used to improve the rendering quality of point based approaches to be more on par with the quality given by volumetric approaches.

5.1 Introduction and Related Works

Despite various improvements in real time performance [WWLVG09], scalability [RV12, WJK+12, KLL+13, CBI13, NZIS13], a key challenge to online 3-D reconstruction remains instabilities in the reconstructed camera trajectory due to imprecision in the range scan alignments of the underlying ICP algorithm [BM92, CM92], which are particularly severe where the acquired object misses sufficiently salient geometric features to latch on to [GIRL03]. These errors accumulate over time, leading to distortions across larger scales in the final reconstruction.

While such drift may partially be mitigated through global offline relaxation [Pul99, RHHL02, ZK13], the need for a global post-process defeats many of the benefits of an online acquisition system. However, recent works [WLSM+15, SSP18] are able to reduce drift in real time by continuously correcting the current reconstruction through loop closure principle.

Various strategies have hence been proposed to minimize registration error already during the online registration stage, including colour feature-based pre-alignment [HKH+12], colour-based weighting of ICP pairs [GRB94, Wei97], stronger weighting of edge features [ZK15], and so on.

This chapter aims at the minimisation of registration errors, by focusing on surface curvature as a reliable feature that is detectable on range scans alone and hence does not depend on accurate multi-sensor alignment. Unlike previous work that took curvature or related measures into consideration, however, curvature is treated as an independent quantity that is consistently incorporated into every stage of the real time reconstruction pipeline, including densely curvature-weighted ICP, range image fusion, local surface reconstruction, and rendering, while maintaining real time rates even for very large scenes.

This work comprises the following features and contributions:

- first online reconstruction design to systematically incorporate curvature as an independent surface attribute into the end-to-end reconstruction pipeline; key innovations are:
  - an ICP variant that considers curvature for both dense correspondence finding and weighting for increased stability,
  - a method to efficiently blend curvatures in the fusion stage,

and, with respect to the underlying PBF framework [KLL+13] (previously presented in Section 4.1.6),

- fast and high-quality, curvature-aware local surface reconstruction directly from an *index map* [KLL+13],
- extension of the index-map approach to mitigate the impact of point collisions and to significantly speed up operations on the model point cloud.

Using multiple benchmark sequences, and in direct comparison to other state-of-the-art online acquisition systems, this approach is shown to significantly reduce drift, both when analysing individual pipeline stages in isolation, as well as seen across the reconstruction pipeline as a whole. All data sets, camera poses, ground-truth geometries, and reconstructions of this method are provided under the following link[1].

5.2 Online Surface Reconstruction

As seen previously in Chapter 4, the design of the current method follows the established overall structure for online reconstruction systems shown in Figure 4.1.1. This general structure is equally shared by the first in-hand scanners [RHHL02, WWLVG09], Newcombe et al.'s KinectFusion [NDI+11], and various later improved systems for online 3-D reconstruction from range images ([NZIS13, KLL+13, WLSM+15, ZK15], amongst others), with differences in algorithmic details and in data representations underlying the individual pipeline stages.

In this section, a brief overview over the system structure is provided while motivating its design decisions in the context of previous work.

Previous work showed continual improvement in accuracy, through algorithmic improvements, but also through improved camera technology and processing speed. One problem, however, remains common to such systems: drift in the recovered camera trajectory, due to geometry-dependent instabilities in the camera pose estimation. This problem is addressed by systematically incorporating curvature as an additional surface attribute into the reconstruction pipeline.

Benefits of including curvature may not be immediately obvious, as one might argue that curvature was simply a function of surface shape, which is already being reconstructed. Also, derivatives of (noisy) real world measurements are generally considered amplifying noise, which would render curvature a potentially unreliable quantity. As it will be shown, however, consistently incorporating curvature throughout the end-to-end reconstruction pipeline leads to significant reduction of drift. The remainder of this section outlines the respective extensions of this method to the online reconstruction pipeline.

Other enhancements orthogonal to the proposed approach, such as incorporation of sensor uncertainty [JU04, SYS07, MAB08, MHFdS+12], use of additional data sources beyond range images [GRB94, Wei97, KSC13,

[1] Full data sets of the method proposed by Lefloch et al. [LKS+17]: `http://www.cg.informatik.uni-siegen.de/3d-reconstruction/low-feature-benchmark`

KM07], simplifying assumptions on structures in the scene [SMGKD14], or non-rigid alignment [BR07, ZMK13], could generally be of additional use but are outside the scope of this thesis.

Depth Map Pre-processing In contrast to Keller et al. [KLL+13] and others KinectFusion like approaches (see Chapter 4 for an introduction of these approaches), a second-order surface property is used (called principal curvatures) that is directly derived from the range image and additionally stored with position and normal information. See Section 5.3 for more details.

Camera Pose Estimation is at the core of what makes hand-held online scanning possible. Incoming depth maps are continuously registered with the partial reconstruction of the object, the *model* acquired so far, to determine the camera's relative position to all previous observations. Any potential drift will occur at this stage, through inaccuracies in the depth map alignment. This chapter will show, however, improvements of the other pipeline stages indirectly reduce drift as well.

Previous work has analysed convergence rates and robustness of ICP, exploring alternative pairing strategies and error metrics [GRB94, Pul99, RL01, GIRL03]. Godin et al. [GRB94] introduce the closest-compatible point strategy that takes surface properties beyond simple point proximity into account during data association. They focus on surface colour but stress generality of the approach; Pulli [Pul99] demonstrates the benefits of considering compatibility of normals. Others report improved convergence when considering compatibility of image intensity and their gradients for pairing [Wei97, SG14].

Extending this strategy to incorporate compatibility of local curvature improves results even further. The system maintains in its model representation a continuously updated account of surface curvatures extracted from the input depth maps.

Beyond this simple compatibility criterion, (implicitly) paying attention to high-curvature regions has proved valuable in previous work: Gelfand et al. [GIRL03] show that normal-space sampling [RL01], i.e., sub-sampling of the surface so that the corresponding normal directions are distributed as evenly as possible, creates point pairs that lead to a much-improved numerical condition of the ICP minimisation.

While neither Gelfand et al. [GIRL03] nor Rusinkiewicz and Levoy [RL01] look at curvature itself, regions of higher curvature generally correspond to a larger spread of normal directions. Therefore, putting more emphasis on high-curvature regions should similarly improve condition. Zhou and Koltun [ZK15] recently presented a system that extracts contour cues from range images to stabilise alignment and thus to reduce drift. While generally characterised by high principal curvature, these contours, however, are treated as a discrete feature that can be present or not, resulting in a bi-level weighting scheme. In contrast, this chapter describes a system that uses continuous weights and is still able to exploit curved features that would not be classified as contour.

A purely feature-based approach has been presented by Johnson and Hebert [JH97], who compute local

spin images across depth maps, using their signatures to match corresponding features in different depth maps. As any feature-based approach [LS09], this has merits if the geometry exhibits a sufficiently dense set of unique surface characteristics. In contrast, this method is based on ICP, which does not rely on unique local features and still converges even in few presence of high-curvature regions, as long as sufficient large-scale characteristics of the surface shapes exist.

Local Surface Reconstruction Data Association requires identifying individual (corresponding) points on the surface of the so-far accumulated model. While early works explored various strategies to construct correspondences between incoming and partially reconstructed model surfaces, the ICP community eventually identified "projective" pairing of incoming and model points as leading to far superior convergence times [RL01]. For each point on the incoming range map, this involves casting a ray along the depth sensor's lines of sight onto the model, and taking the intersection point as a candidate for pairing. Regardless of the underlying model representation, such ray-surface intersections require *local surface reconstruction* in the vicinity of that ray, e.g., ray-surface intersection if the model is explicitly represented as a triangle mesh, ray-casting of an (implicit) volumetric model representation [NDI+11], or some form of "point-based rendering" of surface information into the camera plane [RHHL02, WWLVG09, KLL+13, WLSM+15]. In general, these operations borrow from surface rendering in computer graphics. However, the quantities being sampled (or rendered) depend on the attributes required for data association, typically comprising position, normal, and sometimes colour.

In this work, the local surface reconstruction is expanded upon the one proposed by Keller et al. [KLL+13], which in its original formulation leads to a piecewise-linear local reconstruction, not unlike the approximations by other previous works [WWLVG09]. In contrast, full curvature information is considered when determining local ray-surface intersections, and the following sections will show how the resulting higher-quality surface reconstruction noticeably contributes to the overall drift reduction.

Depth Map Fusion While early online reconstruction systems display more or less raw input data during the online phase ([RHHL02, WWLVG09]), leaving data fusion into a single model to a post-process of global alignment [Pul99] and volumetric fusion [CL96], Newcombe et al. [NDI+11] showed that a real time implementation of Curless and Levoy's volumetric fusion approach [CL96] is possible.

As seen previously in Chapter 4, volumetric fusion approaches require continual conversion between range-map and volumetric representations, and operates with a fixed spatial resolution. Keller et al. [KLL+13] present a purely point-based framework that allows for real time fusion including adaptive resolution, without the need for frequent data conversion.

This method follows the PBF framework, as it allows for the most natural extension to support and analyse the use of curvature throughout the reconstruction pipeline. Similar to their depth map fusion that accumu-

lates normal information independent from positional information, another independent information channel is introduced to accumulate curvature and a method to efficiently blend curvature information during fusion is presented (Section 5.6).

In order to maintain real time rates, the intermediate, screen-space, index map representation is extended. While [KLL+13] uses index maps for data association only, the *deep index map* supports incremental screen-space updates during fusion, enabling online rendering directly off that representation.

Rendering Many previous systems either offer lower-quality visual feedback, sufficient to guide the user toward regions where sensor data is missing [RHHL02], or perform comparatively expensive ray-casting on a volumetric representation [NDI+11, NZIS13]. Weise et al. [WWLVG09] use a local surface reconstruction approach for data association that is equally suitable for (point-based) rendering. Section 5.5 describes a local surface reconstruction approach that works directly on the internal model representation and can equally be used for high-fidelity, curvature-aware, rendering. A simple Phong illumination model is used, coupled with a fast approximation of ambient occlusion known as Screen-Space Ambient Occlusion (SSAO) [Mit07] for added realism in the visual feedback.

5.3 Depth Map Pre-processing

During the preprocessing stage, point attribute maps are extracted from the range image data, following and extending conventions used by Newcombe et al. [NDI+11] and Keller et al. [KLL+13].

5.3.1 Point-Based Fusion Surface Attributes

After outlier removal, depth map values $\mathcal{D}^t(\boldsymbol{u})$ are transformed into 3-D positions in camera coordinates, using the inverse intrinsic camera matrix $\boldsymbol{K}^{-1}$, and stored in a *vertex map* $\mathcal{V}^t(\boldsymbol{u})$, with t the input frame index and pixel coordinates $\boldsymbol{u} = (x, y)^\top$ within the camera image. The *normal map* $\mathcal{N}^t$ is extracted from bilateral filtered depths, a *point-radius map* $\mathcal{R}^t$ is obtained from local point neighbourhood sizes, and, new in this system, a *curvature map* $\mathcal{K}^t$ is derived from $\mathcal{N}^t$ (see Section 5.3.2). Furthermore, following again previous work, a confidence value $\mathcal{W}^t(\boldsymbol{u})$ is assigned for each input frame pixel $\boldsymbol{u}$. This value accounts for the radially decreasing quality of range image values (with fall-offs specific to each Kinect model) and for the reduction of depth quality due to motion blur. The latter is estimated from the relative transformation $\boldsymbol{T}^{t\to(t-1)}$ between adjacent camera poses at times $t-1$ and t (see also Section 5.4.2). For more details on the estimation of surface attributes, refer to the appendix Section A.1.

5.3.2 Curvature Estimation

The curvature map encodes directions and values of principal curvature at each surface point: $\mathcal{K}^t = \{\hat{\boldsymbol{e}}_1, \kappa_1, \kappa_2\}$ stores the first principal direction $\hat{\boldsymbol{e}}_1$ and both curvature values κ_1, κ_2; the second principal direction $\hat{\boldsymbol{e}}_2$ is implicitly given as $\hat{\boldsymbol{e}}_2 = \hat{\boldsymbol{n}} \times \hat{\boldsymbol{e}}_1$.

While surface curvature is well defined on G^2-continuous surfaces, various competing approximations exist for discretised surface representations [MSR07, NA13], and many of them are applicable to point-sampled geometry. Several approaches were tested for their robustness in different application scenario, i.e., on Kinect depth maps.

The eigen decomposition proposed by Pauly et al. [PGK02] not only yields normal estimates, but also a notion of curvature. Even though a rather robust estimate, the approach, however, is not scale-invariant if applied to projectively unevenly sampled geometry.

An alternative class of approximations performs a local surface fit and uses the curvature of the fitted surface as an estimate for the input vertex. Goldfeather and Interrante's adjacent-normal cubic approximation method [GI04], amongst the most robust curvature estimators in literature, uses a polynomial surface fit (typically order 3) that takes into account the normal information of adjacent vertices in its formulation; the method, however, involves a larger (7×7) linear-least squares fit that, even when performed on the GPU, does not meet the real time constraint.

This system uses the chord-and-normal-vectors (CAN) approach of Zhang et al. [ZLC08], which is comparatively robust also in the case of projectively unevenly sampled geometry and still computationally efficient. The CAN estimation initially fits circles to the current oriented vertex $(\mathcal{V}^t(\boldsymbol{u}), \mathcal{N}^t(\boldsymbol{u}))$ and each oriented vertex $(\mathcal{V}^t(\boldsymbol{u}'), \mathcal{N}^t(\boldsymbol{u}'))$ in the selected neighbourhood [ZLC08]. A principal curvature compatible with the fitted circles is then determined as an approximate, minimum least-squares solution that only requires solving a 3×3 linear system and is thus predestined for a GPU implementation. The unknowns of the 3×3 linear system is a combination of trigonometric operation of both principal curvatures values (κ_1 and κ_1) with the angle between the first unit vector of the reference frame (tangent plane described by $\mathcal{N}^t(\boldsymbol{u})$) with the first principal curvature direction. A more robust curvature estimator by Chen et al. [CZ09] would still require solving a 6×6 system, which for current GPU models would no longer be possible in real time.

Figure 5.3.1 compares the quality of Zhang et al. [ZLC08] with Goldfeather and Interrante [GI04] for both an input depth map and an accumulated model. It can be seen, that Zhang's method delivers stable results that are only slightly worse than Goldfeather and Interrante's, mainly for the second main curvature κ_2. Note that under ideal conditions, $\kappa_2 = 0$ along straight edges and $\kappa_2 \neq 0$ for parabolic and saddle points should be expected.

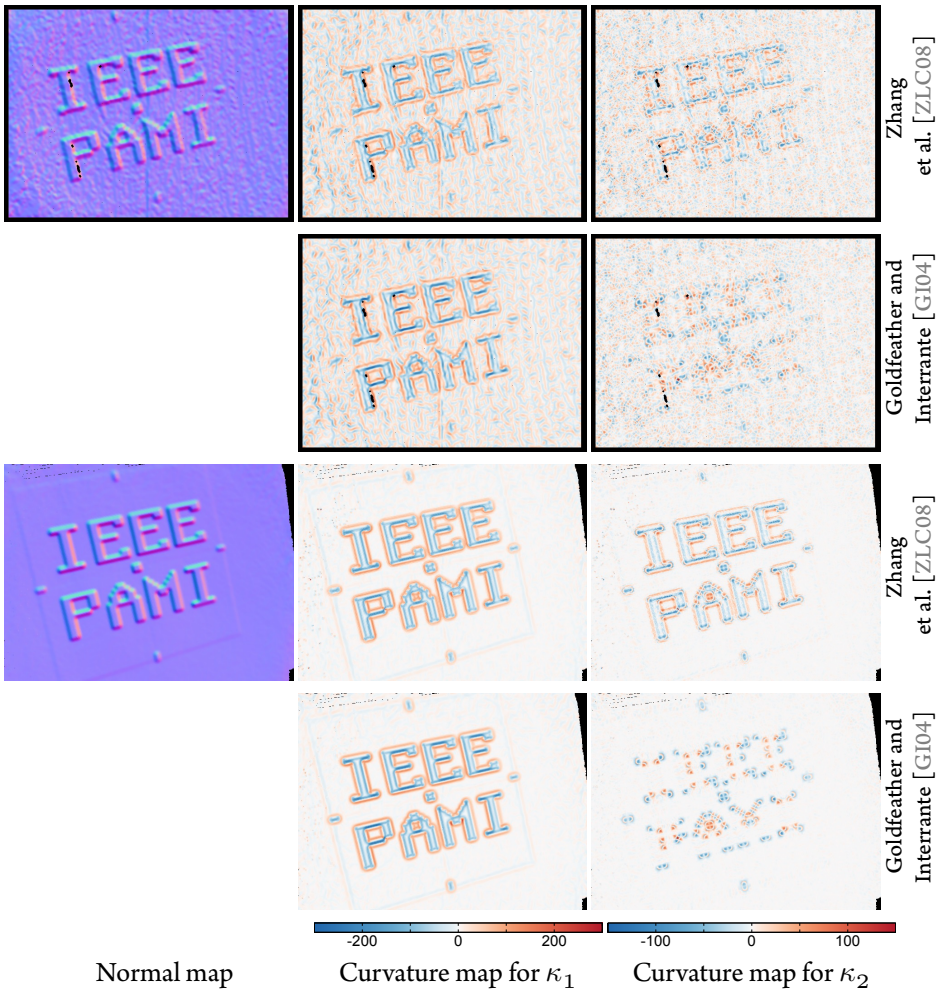


Figure 5.3.1: Example curvature estimation for frame 361 of the **legoPAMI**$_{SL}$ data set [LKS+17], using the methods by Zhang et al. [ZLC08], and Goldfeather and Interrante [GI04], respectively. *first two rows:* using the input frame given by the range camera; *last two rows:* using the current reconstructed surface model. Curvature maps on the *middle* and *right* show κ_1, and κ_2, respectively; corresponding input normal maps convey noise level.

5.4 Camera Pose Estimation

Similar to the pose estimation proposed by Newcombe et al. [NDI+11], this method uses a hierarchical model-to-frame variant of the ICP registration [BM92] and is based on the point-to-plane error metric [Pul99].

This common iterative framework alternates between *data association* (i.e., establishing correspondences between frame t's input points $\boldsymbol{p}_i^t = \mathcal{V}^t(\boldsymbol{u}_i)$ and corresponding points $\boldsymbol{p}_{\mathcal{M}}^* = \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)$ of the model ac-

quired until frame $t-1$), and *minimisation* of an error term $E(\boldsymbol{T}^{t\to(t-1)})$ that expresses the level of mismatch within point pairs under the estimated relative transformation $\boldsymbol{T}^{t\to(t-1)}$.

The main enhancements of this framework are twofold: the correspondence-finding stage additionally considers curvature (on both the input map and the model); furthermore, introducing a curvature-dependent weighting scheme into the error term $E(\boldsymbol{T}^{t\to(t-1)})$, which significantly increases the robustness of the convergence, and thus minimises drift. In the following, a description of these extensions is given in detail.

5.4.1 Data Association

At the beginning of each iteration l, and given the latest estimate of the relative transformation $\boldsymbol{T}_{(l)}^{t\to(t-1)}$ with $T_{(0)}^{k\to(k-1)} := [\mathbf{I}_{3\times3}|\mathbf{0}]$, selection and matching are performed simultaneously, starting with the full set of input points.

Each input point $\boldsymbol{p}_i^t = \mathcal{V}^t(\boldsymbol{u}_i)$, including its geometric entities $\{\hat{\boldsymbol{n}}_i^t, \hat{\boldsymbol{e}}_{1,i}^t, \hat{\boldsymbol{e}}_{2,i}^t, \kappa_{1,i}^t, \kappa_{2,i}^t\}$, is transformed into the model reference $\boldsymbol{p}_j^{t-1} = \boldsymbol{T}_{(l)}^{t\to(t-1)}\boldsymbol{p}_i^t$ (and analogously for vectors $\hat{\boldsymbol{n}}_i^t, \hat{\boldsymbol{e}}_{1,i}^t$ and $\hat{\boldsymbol{e}}_{2,i}^t$). Then, the set of neighbouring model points $\mathcal{H}(\boldsymbol{p}_j^{t-1})$ is built from a 5×5 pixel window of a local surface reconstruction around the projection of $\boldsymbol{p}_j^{(t-1)}$ under $\boldsymbol{T}_{(l)}^{t\to(t-1)}$. Following general practice in point correspondence search [GRB94], points in $\mathcal{H}(\boldsymbol{p}_j^{(t-1)})$ whose position and normal significantly differ from $\boldsymbol{p}_j^{(t-1)}$ are discarded. More precisely, potential correspondences are rejected if $\|\boldsymbol{T}_{(l)}^{t\to(t-1)}\mathcal{V}^t(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)\| \geq \theta_{\text{dist}}$ or $\angle(\boldsymbol{R}_{(l)}^{t\to(t-1)}\mathcal{N}^t(\boldsymbol{u}), \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^*)) \geq \theta_{\text{angle}}$, like Newcombe et al. [NDI+11].

The method is looking for the model point $\boldsymbol{p}_{\mathcal{M}}^* \in \mathcal{H}(\boldsymbol{p}_j^{(t-1)})$ that best matches position, *Darboux frame* $\{\hat{\boldsymbol{e}}_1, \hat{\boldsymbol{e}}_2, \hat{\boldsymbol{n}}\}$, and the respective curvature values κ_1, κ_2, by minimizing a weighted sum of dissimilarity measures of positions (D_p), normals (D_n), and curvature (D_c),

$$\boldsymbol{p}_{\mathcal{M}}^* = \underset{\boldsymbol{p}_{\mathcal{M}}\in\mathcal{H}(\boldsymbol{p}_j^{(t-1)})}{\arg\min} \lambda_p D_p + \lambda_n D_n + \lambda_c D_c\,, \tag{5.1}$$

with

$$D_p = \frac{\left\|\boldsymbol{p}_{\mathcal{M}} - \boldsymbol{p}_j^{(t-1)}\right\|_2}{R}\,, \quad D_n = 1 - \langle \hat{\boldsymbol{n}}_{\mathcal{M}}, \hat{\boldsymbol{n}}_j^{(t-1)}\rangle\,, \quad \text{and}$$
$$D_c = \begin{cases} \frac{\left|\kappa_{1,\mathcal{M}} - \kappa_{1,i}^t\right| + \left|\kappa_{2,\mathcal{M}} - \kappa_{2,i}^t\right|}{\kappa_{\mathcal{M}}^{\max}}\,, & \text{if } \left|\kappa_{1,i}^t - \kappa_{2,i}^t\right| < \theta_\kappa\,, \\ \frac{\left\|\mathcal{Q}_{\mathcal{M}} - \mathcal{Q}_j^{(t-1)}\right\|_2}{\kappa_{\mathcal{M}}^{\max}}\,, & \text{otherwise}\,, \end{cases} \tag{5.2}$$

where R is the maximum radius of the neighbourhood search $\mathcal{H}(\boldsymbol{p}_j^{(t-1)})$, $\kappa_{\mathcal{M}}^{\max} = \max\{|\kappa_{1,\mathcal{M}}|, |\kappa_{2,\mathcal{M}}|\}$

and $\mathcal{Q} = \boldsymbol{C}\,\mathrm{diag}(\kappa_1, \kappa_2, 0)\,\boldsymbol{C}^\top$ with $\boldsymbol{C} = (\hat{\boldsymbol{e}}_1, \hat{\boldsymbol{e}}_2, \hat{\boldsymbol{n}})$ representing the transformation of the model's Darboux frame into the tangent space, scaled by the principal curvatures. The division of D_p and D_c, by R and $\kappa_{\mathcal{M}}^{\max}$, respectively, normalises the components, leading to a relative error measure; all of the following results use $\lambda_{\{p,n,c\}} = \frac{1}{3}$.

As the principal curvature directions are formally undefined for $\kappa_1 = \kappa_2$ (e.g., for planes or spheres) and numerically unstable for $\kappa_1 \approx \kappa_2$ (particularly in the presence of noise), D_c determines curvature dissimilarity independent from curvature directions when κ_1 and κ_2 are similar, i.e., within a threshold θ_κ; we used $\theta_\kappa = 15\,\mathrm{m}^{-1}$ for all experiments.

Furthermore, in the case of $\kappa_1 \neq \kappa_2$, the Darboux frame is unique up to inversion around $\hat{\boldsymbol{n}}$, thus we rotate $\mathcal{Q}_j^{(t-1)}$ by π around $\hat{\boldsymbol{n}}$ if $\langle \hat{\boldsymbol{e}}_{1,\mathcal{M}}, \hat{\boldsymbol{e}}_{1,j}^{(t-1)} \rangle < 0$ to ensure compatible alignment before applying Equation 5.2. $\left\| \mathcal{Q}_{\mathcal{M}} - \mathcal{Q}_j^{(t-1)} \right\|_2$ is evaluated via SVD, exploiting the matrix 2-norm equality $\|\boldsymbol{A}\|_2 = \sigma_{\max}(\boldsymbol{A})$.

5.4.2 Minimisation

The point-to-plane error metric can be expressed as

$$E(\boldsymbol{T}^{t\to(t-1)}) = \sum_{\boldsymbol{u}\in\mathcal{S}} \left\langle \boldsymbol{T}_l^{t\to(t-1)} \mathcal{V}^t(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)\,, \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^*) \right\rangle^2, \tag{5.3}$$

with $\mathcal{S}$ the subset of all input map points for which a valid correspondence has been found, and with $\boldsymbol{u}^*$ being, again, each $\boldsymbol{p}_{\mathcal{M}}^*$'s projection into the previous frame. Some previous ICP works extend this least-squares minimisation by a set of per-correspondence weights $w(\boldsymbol{u}^*)$, yielding

$$E(\boldsymbol{T}^{t\to(t-1)}) = \sum_{\boldsymbol{u}\in\mathcal{S}} w(\boldsymbol{u}^*) \left\langle \boldsymbol{T}_l^{t\to(t-1)} \mathcal{V}^t(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)\,, \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^*) \right\rangle^2. \tag{5.4}$$

Zhou and Koltun [ZK15] choose $w(\boldsymbol{u}^*)$ depending on whether a point is a contour point. In contrast, instead of using bi-level weights only, the present method defines $w(\boldsymbol{u}^*)$ as a continuous function of the curvature information, thus leading to an adaptive, curvature-related weight.

The following curvature weight scheme is implemented which is based on the maximum absolute principal curvature $\kappa_{\mathcal{M}}^{\max}$:

$$w(\boldsymbol{u}^*) = \frac{1}{[\boldsymbol{p}_{\mathcal{M}}^*]_z^2} \left(w'_M(\boldsymbol{u}^*) + \exp\left(-\frac{1}{2} \left[\frac{\lambda}{\kappa_{\mathcal{M}}^{*,\max}(\boldsymbol{u}^*)} \right]^2 \right) \right), \tag{5.5}$$

with $w'_M(\boldsymbol{u}^*) = c_{\mathcal{M}}(\boldsymbol{u}^*)/256$ derived from the model point's confidence counter $c_{\mathcal{M}}$ (see [KLL+13]).

The denominator $[\boldsymbol{p}^*_{\mathcal{M}}]^2_z$ regularizes against noise (correlated with distance) since curvature computation is not reliable enough on data with low signal-to-noise ratio and at far distance, with $[\boldsymbol{p}^*_{\mathcal{M}}]_z$ the z-Cartesian distance of the model point in meters. λ is a control parameter of the curvature-based weight $w(\boldsymbol{u}^*)$ and regulates its influence; for larger λ, the weight increasingly depends on the point's depth and confidence counter only.

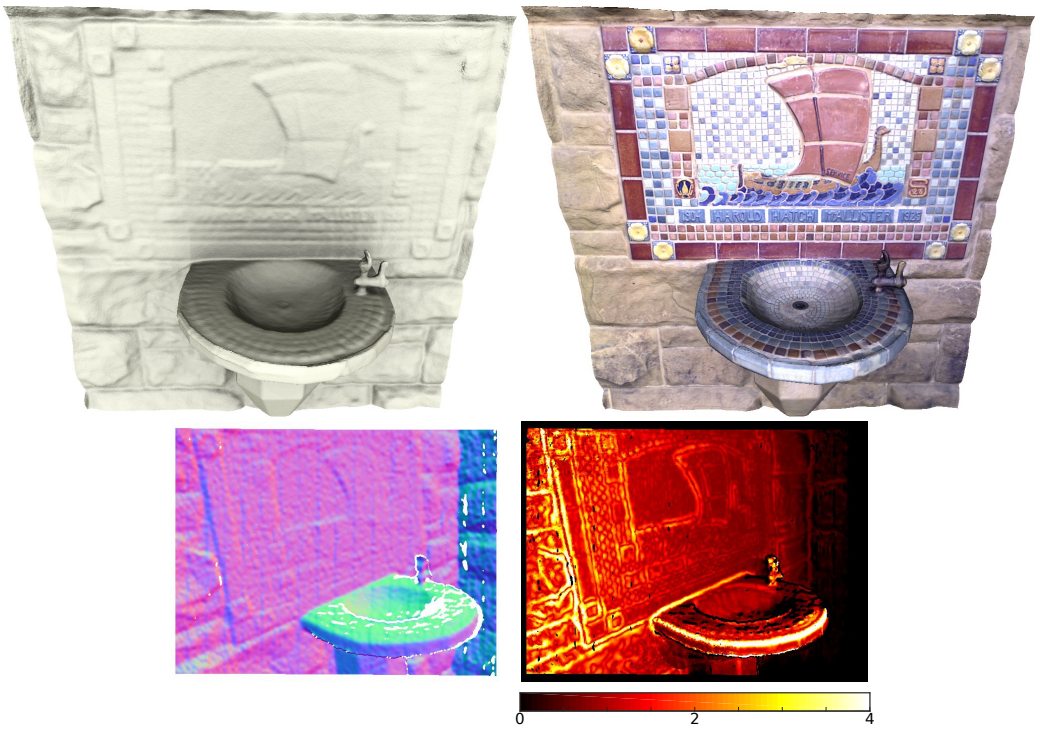


Figure 5.4.1: ICP weight map through curvature estimation. The first row shows the **fountain** data set reconstructed by Zhou and Koltun [ZK14]. The last row shows the normal map $\mathcal{N}^t$ and the ICP weight map (Equation 5.5) used for the minimization at input frame $t = 478$.

All weights will contribute differently for the ICP minimisation increasing the point-to-plane error metric for pair of correspondence points with high curvature. This type of curvature-aware minimisation greatly improves the camera tracking module [LKS+17]. Figure 5.4.1 shows the computation of this weight for each pair of correspondences for the last iteration of the ICP algorithm. Note, how well the weight is given to small curvature structures, even if the input data is quite noisy referred to the normal map. Both maps were generated by the current approach from the **Fountain** data set provided by Zhou and Koltun [ZK14] who were focusing on the computation of high quality colour texture information. For details on the ICP principle, refer to the appendix Section A.2.

5.5 Local Surface Reconstruction

A key ingredient toward improved tracking robustness is paying particular attention to a high accuracy of the local, real time surface reconstruction , which is essential for data association and, finally, pose error minimisation. To that end, the viewing ray is intersected with the second-order surface patches defined by the model point's orientation and curvature (see Section 5.5.1) and apply an elliptically-weighted blending scheme for all patch intersections resulting in the finally reconstructed surface point (see Section 5.5.2). Algorithm 2 summarizes the local surface reconstruction procedure used in the current framework.

Algorithm 2: Model maps generation using the updated index map (see Section 5.5.2).

Input: $\mathcal{M}$ (model), $\mathcal{I}^t$ (index map), ε_d (surface thickness)
Output: $\mathcal{V}_\mathcal{M}$ (model vertex map), $\mathcal{N}_\mathcal{M}$ (model normal map), $\mathcal{K}_\mathcal{M}$ (model curvature map)

foreach *pixel* **u** *in model map* **inparallel do**
  $\boldsymbol{r}$ = generate ray for $\boldsymbol{u}$
  $\mathcal{P}$ = stable points in the vicinity of $\boldsymbol{r}$ using index map $\mathcal{I}^t$
  $z_{\text{front}} = -\inf$

  // Identify intersection points of closed surface
  $\mathcal{L} \leftarrow \emptyset$ // $\mathcal{L}_i.\boldsymbol{v}$: vertex, $\mathcal{L}_i.\hat{\boldsymbol{n}}$: normal, $\mathcal{L}_i.w$: weight
  foreach $\mathbf{q} \in \mathcal{P}$ **do**
    $(\boldsymbol{v}_q, \hat{\boldsymbol{n}}_q, \mathcal{K}_q) = \boldsymbol{r} \bigcap$ surface patch at $\boldsymbol{q}$ // see Section 5.5.1
    if $z_{front} - \varepsilon_d < \mathbf{v}_q.z$ **then**
      $w_q = \exp(-\frac{1}{2}\left(\left|\boldsymbol{q} - \boldsymbol{v}_q\right| / \mathcal{R}^t(\boldsymbol{u})\right)^2)$ // blend weight
      $\mathcal{L}$.append$(\boldsymbol{v}_q, \hat{\boldsymbol{n}}_q, w_q)$
    // Identify closest stable intersection point
    if $z_{front} < \mathbf{v}_q.z$ **then**
      $z_{\text{front}} = \boldsymbol{v}_q.z$
      $\mathcal{K}_\mathcal{M}(\boldsymbol{u}) \leftarrow \mathcal{K}_\mathcal{M}(\boldsymbol{q})$

  // Model map output w.r.t. points on closest surface
  $\mathcal{V}_\mathcal{M}(\boldsymbol{u}) \leftarrow \boldsymbol{0}$ // initialise model map vertex position
  $\mathcal{N}_\mathcal{M}(\boldsymbol{u}) \leftarrow \mathbf{0}$ // initialise model map normal
  $w_{\text{valid}} = 0$ // initialise sum of blend weights
  foreach $(\mathbf{v}, \hat{\mathbf{n}}, w) \in \mathcal{L}$ **do**
    if $z_{front} - \varepsilon_d < \mathbf{v}.z$ **then**
      $w_{\text{valid}} \leftarrow w_{\text{valid}} + w$
      $\mathcal{V}_\mathcal{M}(\boldsymbol{u}) \leftarrow \mathcal{V}_\mathcal{M}(\boldsymbol{u}) + w\boldsymbol{v}$
      $\mathcal{N}_\mathcal{M}(\boldsymbol{u}) \leftarrow \mathcal{N}_\mathcal{M}(\boldsymbol{u}) + w\hat{\boldsymbol{n}}$
  // Normalise result
  $\mathcal{V}_\mathcal{M}(\boldsymbol{u}) \leftarrow \mathcal{V}_\mathcal{M}(\boldsymbol{u}) / w_{\text{valid}}$
  $\mathcal{N}_\mathcal{M}(\boldsymbol{u}) \leftarrow \text{normalize}(\mathcal{N}_\mathcal{M}(\boldsymbol{u}))$
  return

5.5.1 Quadratic Surface Patch Intersection

The main idea to incorporate the curvature information into the local surface reconstruction is to replace the standard ray-plane intersection used by previous works [WWLVG09, KLL+13] by an intersection with a higher-order surface with the same curvature as the one stored in the model point under consideration.

Knowing the Darboux frame $\{\hat{\boldsymbol{e}}_1, \hat{\boldsymbol{e}}_2, \hat{\boldsymbol{n}}\}$ and the curvature amplitudes $\{\kappa_1, \kappa_2\}$ for a given model point $\boldsymbol{p}$, we define an explicit quadratic surface parameterised over the $\hat{\boldsymbol{e}}_1$-$\hat{\boldsymbol{e}}_2$ tangent plane in local coordinates as:

$$F(x, y) = \tfrac{1}{2}\kappa_1 x^2 + \tfrac{1}{2}\kappa_2 y^2. \tag{5.6}$$

The local coordinate of the quadratic surface is represented by the Darboux frame centred at the current point position. The intersection is computed in the local coordinate frame, therefore the ray $\boldsymbol{r}(\alpha) = \boldsymbol{q} + \alpha\mathbf{d}$ is transformed into local coordinates of the Darboux frame $\boldsymbol{r}'(\alpha) = (q'_x, q'_y, q'_z)^\top + \alpha(d'_x, d'_y, d'_z)^\top$. The intersection of $\boldsymbol{r}(\alpha)$ with the quadratic surface patch from Equation 5.6 can be computed as follows:

$$\begin{cases} \tfrac{1}{2}\kappa_1 x^2 & + & \tfrac{1}{2}\kappa_2 y^2 & - & z & = & 0 \\ & & \boldsymbol{q}' & + & \alpha\mathbf{d}' & = & (x, y, t)^T \end{cases}$$

$$\Leftrightarrow \frac{1}{2}\kappa_1(\alpha d'_x + q'_x)^2 + \frac{1}{2}\kappa_2(\alpha d'_y + q'_y)^2 - \alpha d'_z + q'_z = 0,$$

$$\Leftrightarrow \frac{1}{2}\alpha^2(\kappa_1 d'^2_x + \kappa_2 d'^2_y) + \alpha(\kappa_1 d'_x q'_x + \kappa_2 d'_y q'_y - d'_z) + \frac{1}{2}(\kappa_1 q'^2_x + \kappa_2 q'^2_y) - q'_z = 0.$$

Leading to the following quadratic equation:

$$\begin{aligned} &\frac{1}{2}\alpha^2 A + \alpha B + C \;=\; 0\,, \\ \text{where}\quad &A = \kappa_1 d'^2_x + \kappa_2 d'^2_y, \quad B = \kappa_1 d'_x q'_x + \kappa_2 d'_y q'_y - d'_z\,, \\ \text{and}\quad &C = \frac{1}{2}(\kappa_1 q'^2_x + \kappa_2 q'^2_y) - q'_z\,. \end{aligned}$$

The resulting intersection point $\boldsymbol{s}' = (s'_x, s'_y, s'_z)^\top$ is expressed in the local Darboux frame coordinates.

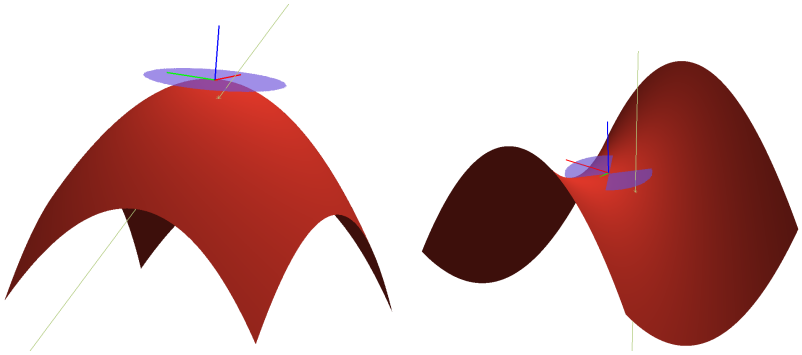


Figure 5.5.1: Synthetic visualisation of ray intersection with two different types of quadratic surface patches (spherical $[\kappa_1 = -300, \kappa_2 = -300]$ and saddle $[\kappa_1 = -300, \kappa_2 = 150]$). The blue disc represents the current surfel data with its associate Darboux frame, the red surface refers to the quadratic surface patch according to Equation 5.6.

Additionally, the associated surface normal is given by:

$$\hat{\boldsymbol{n}}_q = \frac{\boldsymbol{n}}{\|\boldsymbol{n}\|} \quad \text{with} \quad \boldsymbol{n} = \begin{pmatrix} \kappa_1 s'_x \\ \kappa_2 s'_y \\ -1 \end{pmatrix}, \tag{5.7}$$

using Equation A.1.

The intersection point and its normal are finally back-transformed into camera coordinates.

Figure 5.5.1 shows two different types of quadratic surface patches (spherical and saddle) and their respective ray intersection. Solving this intersection equation usually leads to two solutions. The closest one to the related model point $\boldsymbol{p}$ is chosen as the most appropriate intersection. The weight for this intersection point is calculated using a (non-normalised) Gaussian distribution, scaled by the model point's radius $\mathcal{R}^t(\boldsymbol{u})$ (see Alg. 2).

The resulting intersection point $\boldsymbol{s}' = (s'_x, s'_y, s'_z)^\top$ is expressed in the local Darboux frame coordinates. Additionally, the associated surface normal is corrected using Equation 5.7. The intersection point and its normal are finally back-transformed to camera coordinates.

5.5.2 Blending of Quadratic Surface Intersection Points

Since a set of local surface intersections for each model-map pixel $\boldsymbol{u}$ is computed, a fast weighted average of all intersection points can be deduced belonging to the first surface shell in the current point-based model representation. As model points and their respective quadratic surface patches are not perfectly aligned in

depth, due to noise and quantisation, points whose depth values fall into a *depth tolerance threshold* ε_d are blended in order to identify all model points contributing to the model map at $\boldsymbol{u}$.

The current approach shows strong similarity to point-based rendering techniques, especially to differential point rendering [KV01], as well as elliptical weighted average (EWA) splatting [ZPBG02], of which even curvature-rendering variants exist [BSK04]. A key difference to previous works in that field is the order in which surface samples are collected, which eliminates the costly need for a dedicated normalisation render step: rather than accumulating splat contributions for all pixels by each model point individually before dividing each pixel by the sum of relative weights accumulated so far, the enhanced index map (see Section 5.7) provides direct access to all model points contributing[2] to each pixel, allowing for local (and trivially parallel) evaluation of each surface intersection, rather than employing the costly distribute-and-gather process of traditional EWA splatting.

For each pixel of the current input frame, the proposed parallel implementation identifies the intersection point at distance z_{front} closest to the camera, selects all intersection points lying within the given surface thickness ε_d, and blends the intersection points including their normals in order to get the final model map entries $\mathcal{V}_{\mathcal{M}}(\boldsymbol{u}), \mathcal{N}_{\mathcal{M}}(\boldsymbol{u})$. As blending curvature information is computationally more complex (see Section 5.6), the final curvature information is taken from the closest intersection point. Algorithm 2 describes this procedure in detail.

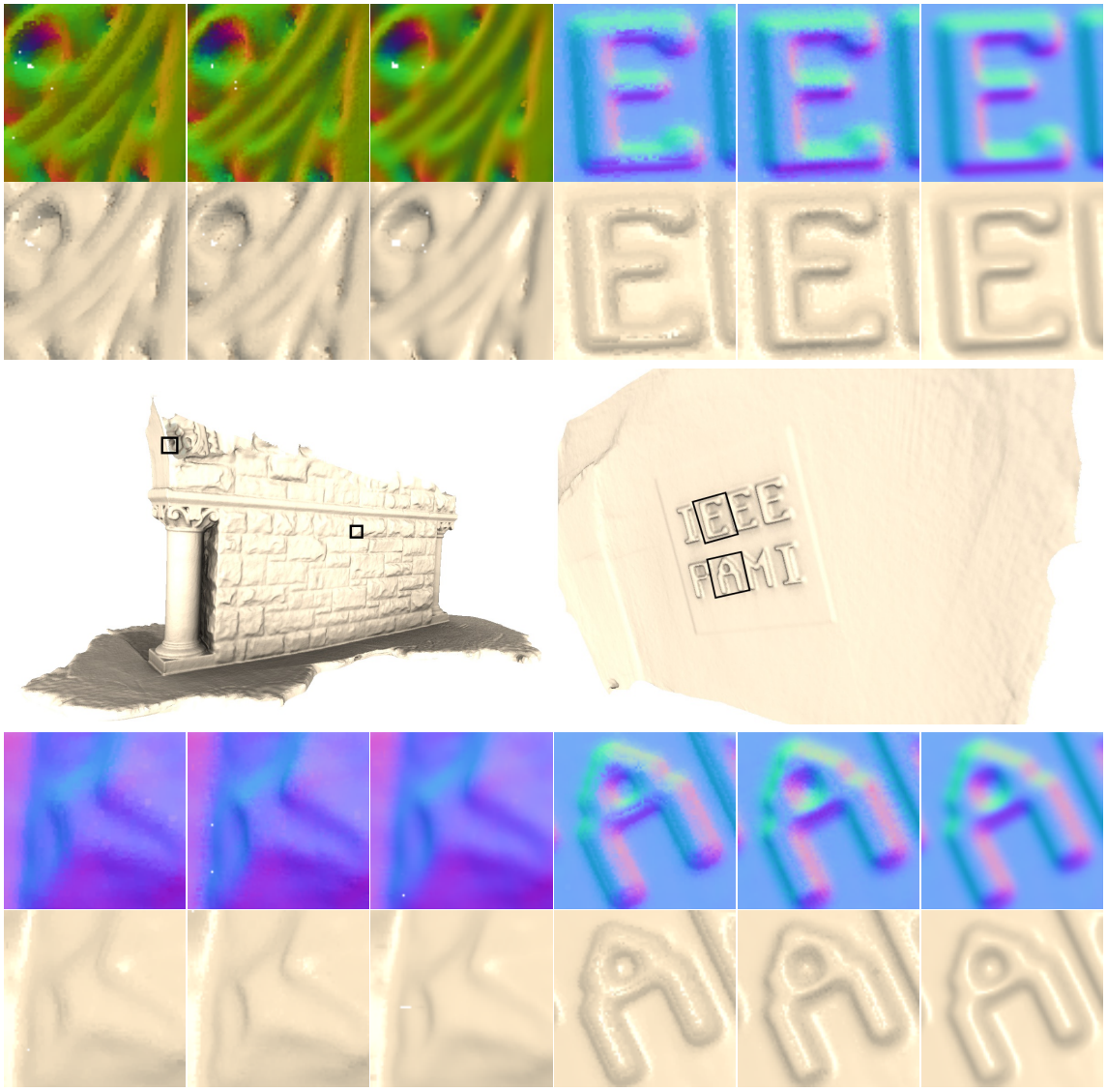
Figure 5.5.2 shows a comparison of the model map quality for two different representations for two sequences, applying the splatting from Keller et al. [KLL+13] (left column), replacing the ray-plane intersection in the splatting with the intersection scheme for quadratic surfaces described in Section 5.5.1 (middle column), and the blending scheme for intersections with the quadratic surface explained in Section 5.5.2 (right column). It can be seen, that the model map quality already increases when curvature information is used, but the blending further mitigates discontinuities at splat boundaries. For all the presented experiments, the depth tolerance ε_d is set to 5 mm.

5.6 Depth Map Fusion

Conceptually, point-based data fusion follows Keller et al. [KLL+13] by accumulating geometric point attributes independently, thus, avoiding costly re-computation of normals and curvature from a local neighbourhood of points.

A simple convex combination is used to accumulate an input point's position $\boldsymbol{p}_i$ into the position $\boldsymbol{p}_{\mathcal{M}}$ of

[2] Screen-space size of model points is artificially bound, as in a real time acquisition setting, such cases would mean that the current camera data is of much higher quality than the coarse model points in question. This allows to collect all points contributing to a pixel within a fixed-size window.



[KLL+13] Quadr. ∩ [LKS+17] [KLL+13] Quadr. ∩ [LKS+17]

Figure 5.5.2: Comparing model map quality for two different scenes. The first sub-column refers to the simple splatting proposed in [KLL+13]), the second to the quadratic surface intersection based on curvature information (see Section 5.5.1) and the third one is the proposed blending scheme using the quadratic surface intersection (see Section 5.5.2).

its associated model point:

$$\boldsymbol{p}_{\mathcal{M}} \leftarrow \frac{c_i \boldsymbol{p}_i + c_{\mathcal{M}} \boldsymbol{p}_{\mathcal{M}}}{c_i + c_{\mathcal{M}}}, \quad c_{\mathcal{M}} \leftarrow c_i + c_{\mathcal{M}}, \tag{5.8}$$

where c_i and $c_{\mathcal{M}}$ are the input weight of the point, and the confidence counter of the model point (the accumulated weight of the input points), respectively.

Darboux frames are fused using a fractional rotation (slerp) of the model Darboux frame towards the input frame using the unique 3-D rotation matrix $\boldsymbol{R}(\hat{\boldsymbol{a}}, \phi)$ described by a rotation axis $\hat{\boldsymbol{a}}$ and an angle ϕ, transforming between the frame $\boldsymbol{C}_{\mathcal{M}} = (\hat{\boldsymbol{e}}_{1,\mathcal{M}}, \hat{\boldsymbol{e}}_{2,\mathcal{M}}, \hat{\boldsymbol{n}}_{\mathcal{M}})$ and $\boldsymbol{C}_i = (\hat{\boldsymbol{e}}_{1,i}, \hat{\boldsymbol{e}}_{2,i}, \hat{\boldsymbol{n}}_i)$:

$$\begin{aligned} &\boldsymbol{C}_{\mathcal{M}} \leftarrow \boldsymbol{R}(\hat{\boldsymbol{a}}, \alpha\phi)\boldsymbol{C}_{\mathcal{M}} \\ &\text{with} \quad \boldsymbol{R}(\hat{\boldsymbol{a}}, \phi) = \boldsymbol{C}_i \boldsymbol{C}_{\mathcal{M}}{}^{\top} \quad \text{and} \quad \alpha = \frac{c_i}{c_i + c_{\mathcal{M}}}\,. \end{aligned} \tag{5.9}$$

As described in Section 5.4.1, the Darboux frame is unique up to inversion around $\hat{\boldsymbol{n}}$, thus input frames are suitably inverted in order to ensure fractional rotation along the shorter path. Finally, the curvature amplitudes are accumulated analogously to Equation 5.8.

5.7 Deep Index Map

The previous sections primarily focused on quality improvements that lead to drift reduction. Equally important, however, is real time processing that operates at the camera's native frame rate. This is not the least as there is a direct relationship between throughput of range images and reduction of drift: aligning more range maps yields more data per surface area and thus less measurement uncertainty; it also implies shorter baselines between consecutive frames, which in many scenarios translates to higher stability of the range-map alignment, that is, camera pose estimation.

In order to efficiently handle models of up to several million points including their associated attributes, the entire reconstruction pipeline is implemented on the GPU using CUDA. Such an implementation has to support various operations at once, including efficient spatial addressing for *data association*, *local surface reconstruction*, *point attribute manipulation* during fusion, and efficient removal of *outliers* and *invalid model points* due to moving objects in the scene.

Keller et al. [KLL+13] enable efficient spatial access to the unordered point cloud by introducing a simple screen-space data structure, the *index map*, in the data association and fusion stages. Rather than rendering a dense surface reconstruction from the camera's perspective to determine all camera viewing ray-surface intersections, only pixel-sized points are rendered that encode vertex indices rather than colours in the output map. Thus, model points that project close to a given camera pixel $\boldsymbol{u}^*$ can easily be identified by looking up the index map in the vicinity of $\boldsymbol{u}^*$. The remainder describes the extensions of the current method to their index map that improve performance at all pipeline stages.

5.7.1 Point Collisions

Depending on viewing distance and model resolution, multiple model points may map to the same pixel in the index map. In order to more reliably determine a suitable match, Keller et al. reduce the risk of collisions by 4×4-upsampling their index map.

In different experiments, however, we often found that much of the 16-fold increased pixel space of the upsampled index map remains unused while collisions are still frequent. The effect is similar to what is observed in memory cache design: even with uniformly random-distributed data, collisions are extremely likely.

Borrowing from cache design, which addresses this problem through the *set associative cache* that trades cache address space for additional storage to resolve collisions, the amount of upsampling is reduced and (partially) resolve collisions by storing multiple point indices per pixel.

Concretely, the *deep index map* stores indices for up to two stable and one unstable points at each pixel position, which allows to reduce the upsampling to 2×2 while still losing fewer points to collisions than Keller et al.'s approach at 4×4. Furthermore, by allocating separate capacity for storing stable and unstable points we effectively eliminate situations where unstable points occlude stable model points, a case that can hamper depth map fusion, for instance in the presence of dense outliers due to a misregistered camera frame or moving object.

Note that parts of the model seen from a far distance, or under an oblique angle, create more collisions than can be held by the deep index map. In practice, however, such cases tend to occur in spatial regions where the camera data is deemed too unreliable to be fused into the model.

5.7.2 Screen-Space Updates

Any point update in the fusion stage, be it merging of an input point (which could have a point change its position in screen-space) or removal due to point expiration or free-space violation, triggers removal of one of the original model points. In Keller et al.'s implementation this removal required a copy-intensive model point cloud array compaction in every frame.

In contrast, we eliminate the need for a costly point copy operations by first marking all required deletions within the deep index map itself, in a dedicated per-pixel "removal index" field. New, incoming points are either merged with existing model points or held in a queue for later insertion. Once all removals are known, the index buffer's removal index plane is sorted (in parallel) to obtain a list of freed positions in the model point list, which is where new model points from the insertion queue are inserted. Further insertion points are appended to the model list, unused freed positions remain tagged as free for future use.

Furthermore, by logging newly created vertex indices in the deep index map, the map is kept fully up to date during fusion, which allows the final rendering stage to retrieve the latest model version directly from

that map, without the need to one more time iterate over millions of points for rendering.

In summary, the proposed deep index map improves and speeds up critical operations throughout the pipeline, which helps maintain real time rates with the curvature-enhanced online reconstruction pipeline.

5.8 Results

Since inception of the ICP algorithm [BM92, CM92], copious design variants have been implemented. Interplay of the different design decisions is non trivial, and the best holistic analysis of the ICP design space available [RL01] predates real time, dense ICP implementations. This section will present a meaningful set of comparisons and evaluations that expose the inherent benefits of the proposed approach.

This approach will be compared to the following state-of-the-art techniques in the context of KinectFusion-like surface reconstruction, due to a combination of their availability, their proven high quality, and their input-output compatibility.

Kell13: The point-based approach given by Keller et al. [KLL+13]. The authors' implementation has been used.

Nies13: Niessner et al. [NZIS13] voxel-based hashing technique. The authors' implementation has been used[3] with the finest grid resolution of 4 mm.

Sera15: Serafin and Grisetti introduce Dense Normal Based Point Cloud Registration (NICP) [SG14, SG15]. Like the proposed approach, their method builds upon derivatives for improved reconstruction. Their publicly available implementation has been used[4]. Note that a modified version of **Sera15** is also shown here where pre-bilateral filtered depth maps are used to compute the surface normals. These filtered depth maps are the same as the ones used by the proposed method.

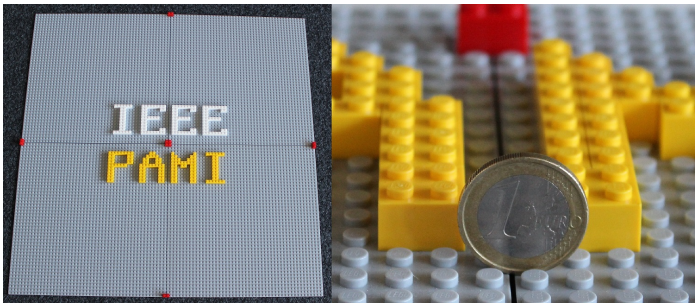
To have their computationally costly CPU implementation of NICP run in real time, Serafin and Grisetti suggest to work on images of one quarter of the original size. However, to provide a fairer quality comparison, NICP was run offline, at full resolution (640×480, or 512×424, respectively); consequently, the presented results would currently not be achievable within an online system.

Lefl17: the approach described in this chapter and proposed by Lefloch et al. [LKS+17] that uses curvature information throughout the overall reconstruction pipeline. For further analysis, some innovation of the proposed method were partially disable, to observe their effect on tracking quality (see Section 5.8.3.

Various other approaches for online fusion have been proposed, using conceptually orthogonal ways to address drift, including offline and online optimiation of camera poses and surface reconstruction [WWLVG09,

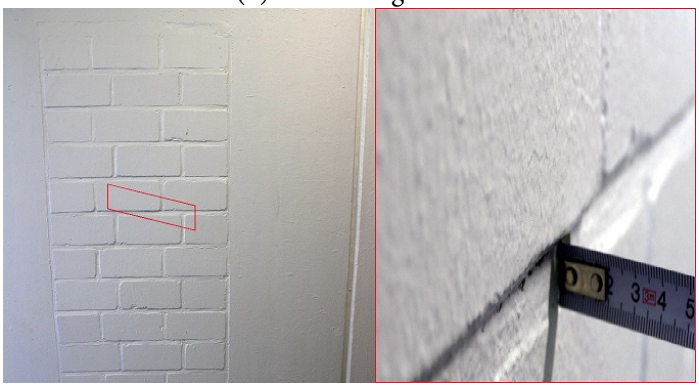
[3] Open-source code of [NZIS13]: `http://graphics.stanford.edu/~niessner/niessner2013hashing.html`
[4] Open-source code of [SG15]: `http://jacoposerafin.com/nicp/`

WLSM+15, FTF+15]. In this work, however, the merits of comprehensive use of curvature information throughout the reconstruction pipeline will be specifically shown.



(a) LEGO™ small scale

(b) LEGO™ big scale



(c) Brick wall

Figure 5.8.1: Objects used to create all sceneries. A 1€ coin ($= 2.325$ cm) is used to visualise the object scale.

The results will be presented using several reference data sets with and without geometric and/or camera pose ground-truth:

Lego-PAMI-TT: The letter sequence "IEEE PAMI" constructed out of LEGO™ pieces on a 80×80 cm^2

ground plate is acquired using a range camera. The camera is fixed at about 80 cm height above the scene, which resides on a precisely controllable turntable. The scene's controlled rotation—one 360° revolution in the course of 1601 frames—yields ground-truth (relative) camera poses where the first and the last positions coincide.

Two different scales of the "IEEE PAMI" scene have been used: **Lego-PAMI-TT**$^{\times 1}$ and **Lego-PAMI-TT**$^{\times 2}$ with a single and a double block width (uniform scale factor of 2), respectively (see Figure 5.8.1).

Geometry ground-truth with high precision is also provided using the LEGO™ Designer software (`http://ldd.lego.com/en-us/`). However, the LEGO™ knobs (cylindrical connectors) are at or below the depth resolution limit of current Kinect range cameras.

Lego-PAMI-Free: The two "IEEE PAMI" scenes are acquired with a free-hand uncontrolled camera motion at about $50 - 100$ cm distance from the target and acquiring some 1,100 frames. Here, ground-truth is available for geometry only, not for camera pose.

Stone-Wall: This data set by Zhou and Koltun [ZK13] comprises some 2,700 input frames of a wall with approx. $5.8\,\text{m} \times 2.8\,\text{m} \times 0.7\,\text{m}$ size, acquired with Asus Xtion Pro Live range camera and including a prominent loop closure (`www.stanford.edu/~qianyizh/projects/scenedata.html`).

Brick-Wall: This scene comprises of a nearly planar wall with very thin depth features ($\leq$ 4 mm) only present at the wall's brick interstices (see Figure 5.8.1). The wall was acquired using a hand guided Kinect camera at a distance of approx. $50 - 100$ cm, yielding some 800 depth frames. The scenery covers approximately 1.80×1.70m.

Neither ground-truth of camera poses nor geometry are available.

Racing-Car-R3: Wasenmüller et al.'s time-of-flight sequence [WMS16] comes with a high-quality ground-truth mesh that allows for direct evaluation of reconstruction error.

mit_76-417b: The dataset by Xiao et al. [XOT13] offers a long-distance structured-light sweep of a large-scale open office space, which is suitable to demonstrate performance for very large datasets, in terms of both memory efficiency and camera drift over time.

The three scenes **Lego-PAMI-TT**, **Lego-PAMI-Free**, and **Brick-Wall** have been acquired using the Kinect$^{\text{SL}}$ and the Kinect$^{\text{ToF}}$ cameras. Subscripts are used to distinguish between the two Kinect types, e.g., **Lego-PAMI-TT**$_{SL}$ and **Lego-PAMI-TT**$_{ToF}$. The Kinect$^{\text{SL}}$ and the Kinect$^{\text{ToF}}$ have quite different quality levels for depth, noise, and other error sources; see Sarbolandi et al. [SLK15] for a detailed discussion.

5.8.1 Qualitative Evaluation

We first demonstrate the robustness of the proposed method in a qualitative way by visually comparing its reconstruction results to state-of-the-art methods.

Figure 5.8.2: Comparison of reconstructions for the **Lego-PAMI-Free** sequences.

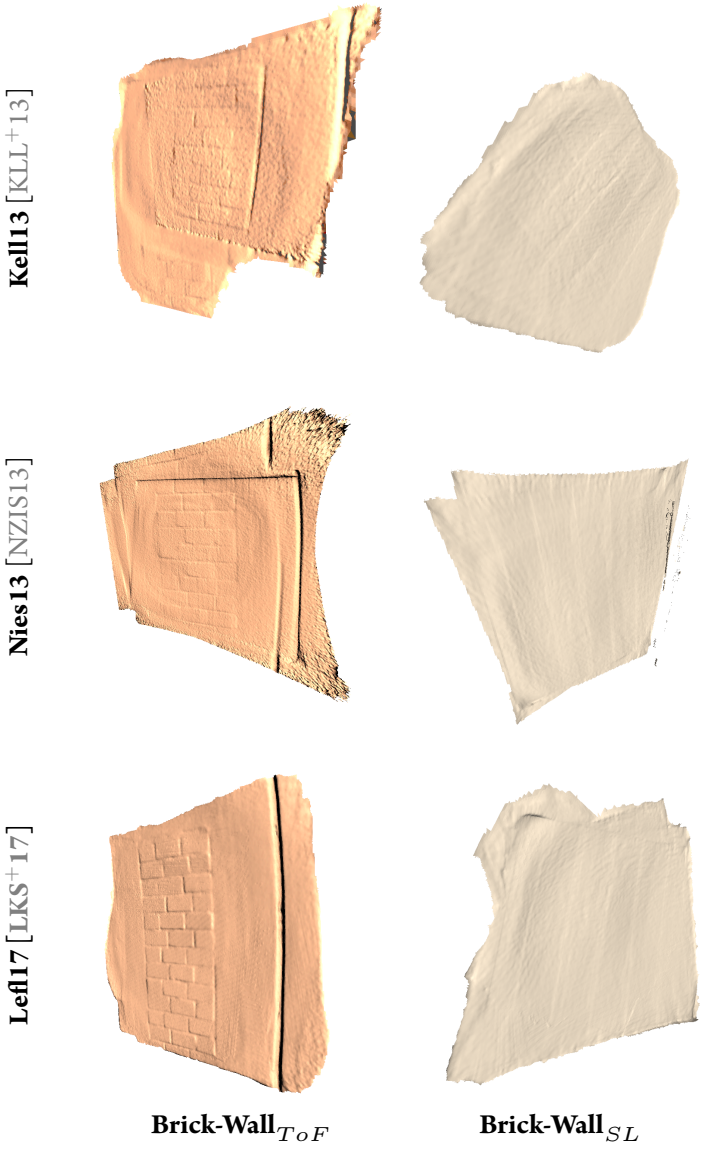


Figure 5.8.3: Comparison of reconstructions for the **Brick-Wall** sequences.

Figure 5.8.2 shows a reconstruction comparison using all **Lego-PAMI-Free** data sets. Even for the double sized LEGO™ scene (rows 1 and 2) some of the state-of-the-art methods, like **Nies13** have severe difficulties in tracking the camera motion. In general, the tracking is more robust for Kinect$^{\text{ToF}}$ data sets than for the Kinect$^{\text{SL}}$ ones, which is most likely due to its better quality in depth resolution and noise [SLK15]. Rows 3 and 4 in Figure 5.8.2 demonstrates the robustness of the curvature-enhanced tracking method. Virtually all

state-of-the-art methods completely fail to retrieve a valid camera motion and/or an appropriate reconstruction for the small scale scenery **Lego-PAMI-Free**$^{\times 1}_{SL/ToF}$, whereas the proposed method yields robust reconstruction even for the Kinect^{SL}with its lower depth quality.

Figure 5.8.3, top row, shows the reconstruction of the **Brick-Wall**$_{ToF}$. Note how well the proposed method is able to robustly reconstruct the subtle wall structure. Nevertheless, this method is still not able to properly reconstruct the Kinect^{SL}acquired scenery **Brick-Wall**$_{SL}$ (see Section 5.9).

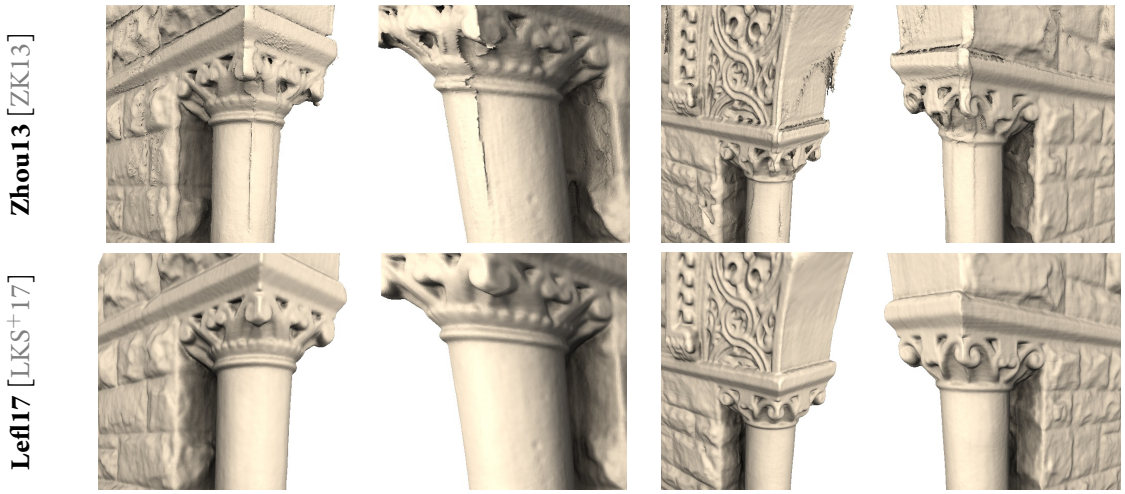


Figure 5.8.4: Comparison of reconstructions for the **Stone-Wall**$_{SL}$ sequence, reconstructed using the offline global optimiser from Zhou and Koltun [ZK13] (top row) and the proposed method (bottom row). The two first columns refer to the right pillar of the stonewall and the two last columns to the left pillar of the stonewall, where the acquisition starts and ends.

For completeness, Figure 5.8.4 compares the proposed approach against the offline global optimiser **Zhou13**. The proposed approach shows no apparent drift, i.e., when returning to the initial camera pose, and seamlessly completes the model. Be aware that the proposed method leads to an artifact-free reconstruction due to a valid ego motion. However, a one-to-one comparison of visual results is not possible since both methods use different model representation (*TSDF over a uniform voxel grid* and *point-based*).

5.8.2 Quantitative Ground-truth Evaluations

The following series of experiments considers scenarios in which a reliable ground-truth exists for comparison.

A) Camera Tracking Camera tracking accuracy is evaluated using the turntable dataset **Lego-PAMI-TT**, which provides a measure of the robustness of the compared methods. Due to the rigid acquisition setup, the camera pose estimated in the ICP should ideally result in an equidistantly sampled, perfect circle, with a constantly rotated optical axis. Therefore, reference poses are generated in order to evaluate the estimated camera poses as follows:

1. RANSAC fit of camera poses to a plane, removing the influence of outliers,
2. RANSAC fit of a circle in the plane to the projected camera centres, and
3. projection of the initial camera pose to the circle as starting point for regularly sample the circle.

Using these reference camera poses, the *camera centre error* is calculated as the Euclidean distance between the estimated ICP-pose and the corresponding reference point. In order to compute the *rotation angle error* the angular argument of the rotational transformation matrix is extracted between the initial pose 0 and the current pose at t and is compared against the ideal angular offset $\Delta p = (360/1600)^\circ$ between neighbouring frames: $\left|\, \text{angle}(\boldsymbol{R}_t \boldsymbol{R}_0{}^\top) - t\Delta p \,\right|$.

The reconstructed geometry is quantitatively evaluated for all **Lego-PAMI-TT** data sets by extracting the relevant scenery from reconstructed geometry, registering the reconstructed geometry to the ground-truth LEGO™ model and, finally, computing distance errors using CloudCompare [GM13].

Tab. 5.8.1 and Tab. 5.8.2 present plots of the camera centre and the camera rotation angle errors, images of the geometric reconstruction error, and error statistics (mean, standard deviation, min and max) for the **Lego-PAMI-TT**$_{SL/ToF}$ data sets. (For **Lego-PAMI-TT**$^{\times 1}_{ToF}$, **Sera15** was unable to lock onto the geometry, producing an invalid trajectory of an almost static camera position; these results are hence excluded.)

As expected, the worse signal-to-noise ratio for smaller geometric features of the small scale datasets **Lego-PAMI-TT**$^{\times 1}$ decreases the stability of the ICP-based camera tracker, leading to larger camera centre and rotation angle errors for all methods. Apparently, the proposed method is much more robust than the state-of-the-art methods, which have severe difficulties to retrieve the correct ego motion.

Comparing the small scale with the large-scale data set **Lego-PAMI-TT**$^{\times 2}_{ToF}$, two facts seem to be counter-intuitive. Firstly, the geometric error for the small scene is smaller than for the large scene, even though the tracking is less robust, and, secondly, the proposed method yields slightly larger camera centre and rotational angle errors for the large-scale data set **Lego-PAMI-TT**$^{\times 2}_{ToF}$ than state-the-art methods. The first aspect is explained by comparing a *single input frame* to the ground-truth geometry. This results in less geometric error for the small scale (mean=0.590, SD=0.557, min=0, max=5.943) than for the big scale (mean=0.745, SD=0.766, min=0, max=6.659). This is most likely due to different camera error effects such as multi-path, *flying pixel*, etc. [SLK15]. Furthermore, the averaging applied within any KinectFusion-like approach erases geometric errors due to erroneous tracking after some frames. We have no consistent explanation for the second aspect, i.e., the smaller rotation angle error in conjunction with an extreme camera centre error for

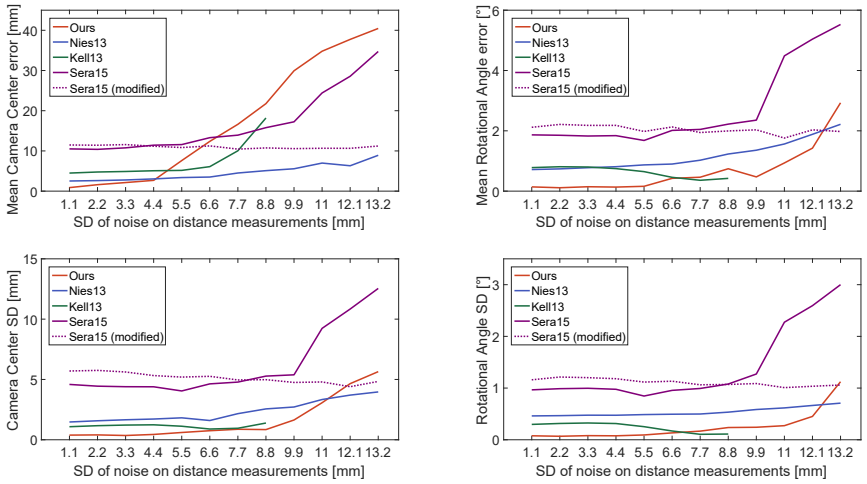


Figure 5.8.5: Mean error (top row) and SD (bottom row) for estimated camera centres (left) and rotational angles (right) with increasing noise for the **Lego-PAMI-TT**$^{\times 2}_{ToF}$ scene. Data points where the camera tracking completely failed are omitted. In order to make the results comparable, the results for a modified version of **Sera15** is included using bilaterally pre-filtered depth images.

Kell13. However, the proposed approach is the only one robust according to both, camera centre and rotation angle error.

B) Impact of Noise As the proposed method is based on second-order derivatives, and because derivatives are known to be sensitive to noise, a systematic study on the influence of noise on the camera tracking accuracy was conducted.

The following noise evaluation is based on the large-scale turntable data set **Lego-PAMI-TT**$^{\times 2}_{\text{ToF}}$. As in the previous section, circle fitting is used to create reference poses.

The estimated camera trajectories at different levels of (Gaussian) noise is evaluated. Starting from a noise level typical for ToF cameras, we successively increase the noise's standard deviation by integer factors, through addition of normal-distributed noise to the original (noisy) depth values; for the principal point of the Kinect$^{\text{ToF}}$, the SD for measured depth values at 80 cm distance (**Lego-PAMI-TT**$^{\times 2}_{\text{ToF}}$) is about 1.1 mm, see Sarbolandi et al. [SLK15].

A complete comparison between **Kell13** [KLL$^+$13], **Nies13** [NZIS13] and **Sera15** [SG14, SG15] is given. **Kell13**, **Nies13**, and the proposed method [LKS$^+$17] have in common that they use a bilateral pre-filter to mitigate noise. **Sera15** does not pre-filter and hence is more susceptible to noise. In order to ensure a meaningful comparison even toward higher noise levels, we additionally evaluate a version where bilaterally pre-filtered images are given to NICP (**Sera15 (modified)**).

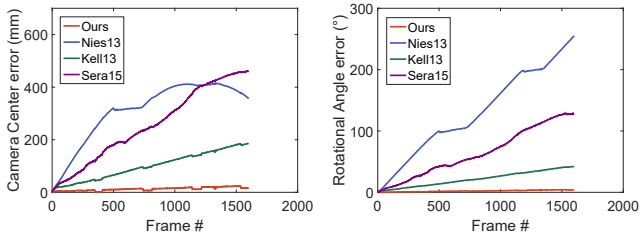
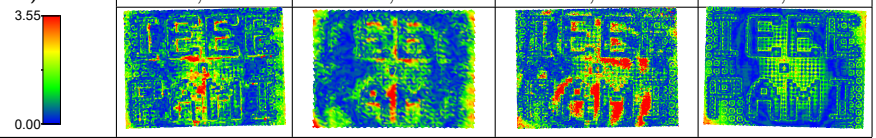
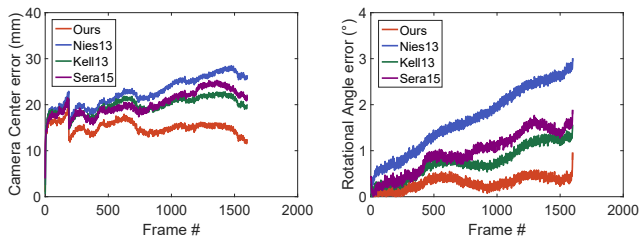
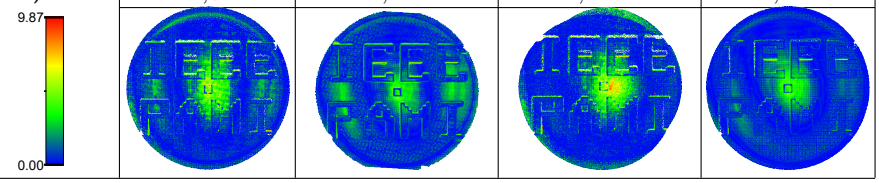
	Error Type	Methods			
		Kell13	**Nies13**	**Sera15**	**Lefl17**
Lego-PAMI-TT$_{SL}^{\times 1}$	Centre [mm]				
	Mean, SD	99.787, 52.101	313.268, 111.632	256.363, 138.939	**12.748**, **6.128**
	Min, Max	4.866, 186.679	5.323, 415.532	1.826, 461.583	**0.186**, **24.679**
	Rot. Angle (°)				
	Mean, SD	21.687, 11.984	130.331, 68.766	64.026, 39.040	**2.177**, **1.209**
	Min, Max	0.409, 42.313	0.197, 255.062	0.044, 129.079	**0.002**, **4.390**
					
	Reconst. [mm]				
	Mean, SD	0.803, 0.744	0.899, 0.863	0.979, 1.043	**0.685**, **0.602**
	Min, Max	**0.000**, 4.995	0.001, 4.728	**0.000**, 8.549	**0.000**, **3.547**
					
Lego-PAMI-TT$_{SL}^{\times 2}$	Centre [mm]				
	Mean, SD	19.883, 1.785	22.774, 3.182	20.274, 2.609	**15.065**, **1.310**
	Min, Max	**0.618**, 22.796	2.550, 28.480	3.963, 25.290	1.068, **19.212**
	Rot. Angle (°)				
	Mean, SD	0.762, 0.342	1.682, 0.704	0.971, 0.446	**0.305**, **0.138**
	Min, Max	**0.000**, 1.564	0.004, 3.010	0.004, 1.887	0.001, **0.959**
					
	Reconst. [mm]				
	Mean, SD	1.012, 1.049	1.061, 1.045	1.103, 3.237	**0.898**, **0.906**
	Min, Max	**0.000**, **9.867**	**0.000**, 10.244	**0.000**, 314.765	**0.000**, 11.233
					

Table 5.8.1: Comparison of the ego-motion robustness for three different methods based on small (top row) and large (bottom row) scales of **Lego-PAMI-TT$_{SL}$** data sets. **Bold numbers** refer to the smallest error for each given error statistic.

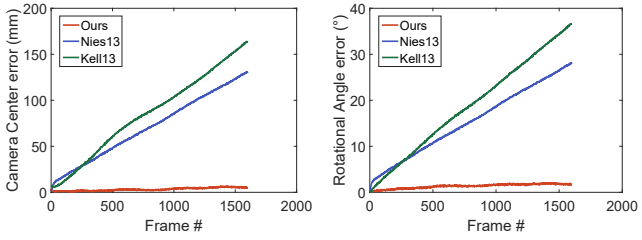
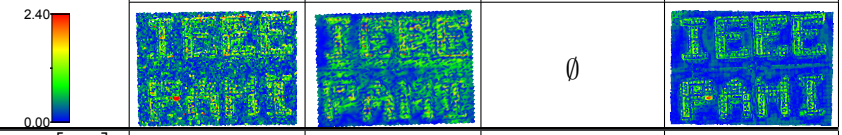
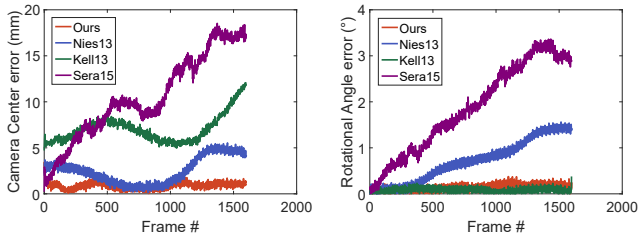
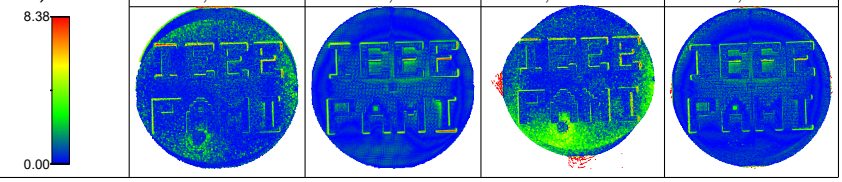
	Error Type	Methods			
		Kell13	**Nies13**	**Sera15**	**Lefl17**
Lego-PAMI-TT$_{\text{ToF}}^{\times 1}$	Centre [mm] Mean, SD Min, Max	 84.866, 45.723 5.608, 164.437	 70.532, 35.280 1.772, 131.159	 ∅	 **3.274**, **1.619** **0.532**, **6.911**
	Rot. Angle (°) Mean, SD Min, Max	 18.760, 10.378 **0.015**, 36.669	 15.397, 7.457 0.212, 28.145	 ∅	 **1.342**, **0.480** 0.086, **2.117**
					
	Reconst. [mm] Mean, SD Min, Max	 0.466, 0.429 **0.000**, 5.030	 0.445, 0.387 **0.000**, **2.402**	 ∅	 **0.398**, **0.383** **0.000**, 3.183
					
Lego-PAMI-TT$_{\text{ToF}}^{\times 2}$	Centre [mm] Mean, SD Min, Max	 7.069, 1.507 4.708, 12.080	 2.520, 1.471 0.130, 5.534	 10.504, 4.588 0.229, 18.524	 **0.877**, **0.386** **0.040**, **3.453**
	Rot. Angle (°) Mean, SD Min, Max	 **0.087**, **0.057** **0.000**, 0.378	 0.715, 0.461 **0.000**, 1.562	 1.866, 0.966 0.141, 3.364	 0.140, 0.077 **0.000**, **0.375**
					
	Reconstr. [mm] Mean, SD Min, Max	 0.725, **0.806** **0.000**, 11.710	 0.726, 0.874 **0.000**, **8.375**	 1.183, 19.065 **0.000**, 3173.000	 **0.637**, 0.807 **0.000**, 25.268
					

Table 5.8.2: Comparison of the ego-motion robustness for three different methods based on small (top row) and large (bottom row) scales of **Lego-PAMI-TT$_{\text{ToF}}$** data sets. **Bold numbers** refer to the smallest error for each given error statistic. Reconstruction runs where a method's camera tracking failed completely are left out from the comparison (and marked as ∅).

	Kell13	**Nies13**	**Lefl17**
Mean, SD	10.405, 12.757	25.371, 24.093	**9.250**, **9.743**
Min, Max	**0.000**, 155.180	**0.000**, 200.466	**0.000**, **153.895**

Table 5.8.3: Absolute distance error [mm] for the **Racing-Car-R3**$_{ToF}$ sequence [WMS16]. For every model point, the absolute distance error to the ground-truth mesh is calculated.

In all of these experiments, the bilateral filter with the same values: $\sigma_{\mathrm{D}} = 2.5, \sigma_{\mathrm{R}} = 0.03, r_{\text{filter}} = 5$. For **Kell13**, we further enabled the use of positions from the bilaterally filtered map in their fusion stage,which otherwise would use positions from unfiltered data.

Figure 5.8.5 shows the resulting mean error (top row) and SD (bottom row) for estimated camera centres (left) and rotational angles (right).

We generally observe that the proposed method reliably produces lowest error at first, while it suffers more than others as noise levels become very high. Depending on the error quantity considered, the cross-over point lies between five to twelve times of the natural noise level (SD 5.5–13.2 mm in our experiment); for a reference of scale, consider that the **Lego-PAMI-TT**$^{\times 2}_{ToF}$ data set consists of LEGO™ double blocks with a height of 19.2 mm, without knobs.

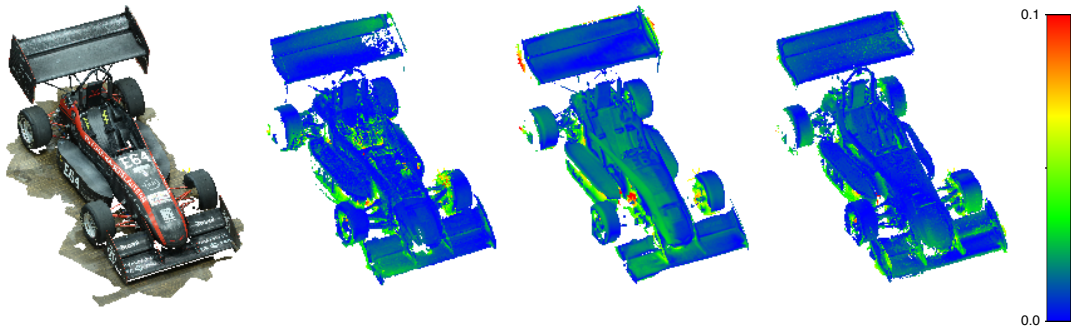
We hence argue that under realistic imaging conditions, the benefits of incorporation of curvature tend to prevail; only in particularly high-noise scenarios, alternative approaches should be preferred.

C) Surface Reconstruction Error The **Racing-Car-R3**$_{ToF}$ sequence by Wasenmüller et al. [WMS16] provides a high-quality ground-truth geometry for a complex object. The reconstructions computed with **Kell13**, **Nies13**, and the proposed approach are compared to the ground-truth geometry. The floor was removed manually so that only the relevant scenery remains, the reconstructed geometry was registered to the ground-truth mesh and the distance error was computed using CloudCompare [GM13].

For **Nies13**, the voxel size was changed from 4 mm (default) to 6 mm; with the default settings, their method stopped adding depth data and produced a corrupted output mesh. For **Kell13**, the removal of dynamic parts of the scene was disabled. All other parameters were kept unchanged (default values).

Figure 5.8.6 shows the absolute distance error [m] to the ground-truth mesh. See Tab. 5.8.3 for mean and standard-deviation of the corresponding reconstruction errors.

The proposed method generally provides significantly lower reconstruction error than **Nies13**, while offering a more complete reconstruction (featuring fewer holes) than **Kell13**, which otherwise shows competitive reconstruction error.



Ground-truth **Kell13** [KLL+13] **Nies13** [NZIS13] **Lefl17** [LKS+17]

Figure 5.8.6: Comparison of reconstructions for the **Racing-Car-R3**$_{ToF}$ sequence by Wasenmüller et al. [WMS16]. For every model point, the absolute distance error [m] to the ground-truth mesh is visualised using the CloudCompare tool [GM13].

5.8.3 Contributors to Robustness

An experiment was conducted in order to evaluate and justify the use of curvature in the individual stages of the proposed reconstruction pipeline. The following variants of the proposed algorithm are compared.

Base: The proposed reconstruction pipeline with deactivated curvature methods for ICP correspondences finding, ICP optimisation, and local surface reconstruction, yielding reconstructions equivalent to Keller et al. [KLL+13].

ICP Weight: **Base** plus activated curvature based ICP weighting scheme (see Section 5.4.2).

Correspondence Finding: **Base** plus activated curvature based ICP correspondences finding (see Section 5.4.1).

Local Reconstruction: **Base** plus activated enhanced local surface reconstruction using curvature information and blending (see Section 5.5).

The small-scale turntable data set **Lego-PAMI-TT**$^{\times 1}_{SL}$ is used, which consists of 1600 frames for a complete 360° turn (0.225° frame-to-frame rotation) and provides a camera pose reference. This data set is most challenging, as it has little depth variation. Furthermore, the frame-to-frame motion is increased by taking one every nth frame, $n \in \{5, 15, 25, 35, 45, 55, 65\}$, resulting in a total of seven experiments and five variants of the reconstruction pipeline.

Tab. 5.8.4 shows the camera centre error statistics for all methods and experiments. Results are only evaluated, if the camera tracking stage of the variant does not completely fail. More precisely, each of the methods that process the full frames of the turntable sequence **Lego-PAMI-TT**$^{\times 1}_{SL}$ give a radius of the fitted circle equals to 26 ± 0.5 cm. Thus, experiment trajectories are rejected from the evaluation if their radius is not close enough to this radius. As can be seen from the results, weighting has a strong impact

and for inter-frame rotation up to $45 \times 0.225^\circ = 10.125^\circ$ weighting alone has very similar results compared to the fully curvature equipped method. Furthermore, curvature-based correspondences finding as well as local surface reconstruction improve on the base algorithm, failing beyond an inter-frame rotation of $15 \times 0.225^\circ = 3.375^\circ$. Weighting alone exhibits strongly decreased robustness at an inter-frame rotation of $55 \times 0.225^\circ = 12.375^\circ$ and fails afterwards.

For completeness, the proposed approach starts degenerating beyond an inter-frame rotation up to $66 \times 0.225^\circ = 14.85^\circ$, thus applying all curvature-based components clearly leads to improved robustness of the overall reconstruction system compared to an isolated curvature component based application.

Drop	**Base** $\mu \pm \sigma$ Min-Max	**Base & ICP Weighting** $\mu \pm \sigma$ Min-Max	**Base & Corresp. Finding** $\mu \pm \sigma$ Min-Max	**Base & Local Reconstr.** $\mu \pm \sigma$ Min-Max	**Lefl17** $\mu \pm \sigma$ Min-Max
1 : 5	3.25 ± 0.59 $0.06 - 4.00$	0.62 ± 0.23 $0.09 - 0.95$	2.97 ± 0.55 $\mathbf{0.02} - 3.68$	2.97 ± 0.53 $0.05 - 3.65$	$\mathbf{0.45 \pm 0.13}$ $0.09 - \mathbf{0.72}$
1 : 15	2.78 ± 0.55 $0.07 - 3.31$	0.55 ± 0.17 $\mathbf{0.01} - 1.00$	2.43 ± 0.48 $0.06 - 2.91$	2.35 ± 0.47 $\mathbf{0.01} - 2.86$	$\mathbf{0.43 \pm 0.14}$ $0.12 - \mathbf{0.85}$
1 : 25	$\emptyset$	$0.55 \pm \mathbf{0.15}$ $0.05 - 0.83$	2.00 ± 0.48 $0.05 - 2.51$	1.91 ± 0.46 $\mathbf{0.02} - 2.42$	$\mathbf{0.46 \pm 0.15}$ $0.12 - \mathbf{0.76}$
1 : 35	$\emptyset$	0.57 ± 0.17 $0.12 - 0.83$	2.07 ± 0.50 $\mathbf{0.03} - 2.58$	$\emptyset$	$\mathbf{0.49 \pm 0.13}$ $0.16 - \mathbf{0.81}$
1 : 45	$\emptyset$	0.59 ± 0.21 $\mathbf{0.13} - 0.91$	$\emptyset$	$\emptyset$	$\mathbf{0.51 \pm 0.15}$ $0.20 - \mathbf{0.84}$
1 : 55	$\emptyset$	7.86 ± 1.36 $0.97 - 8.56$	$\emptyset$	$\emptyset$	$\mathbf{0.52 \pm 0.15}$ $\mathbf{0.20} - \mathbf{0.86}$
1 : 65	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\mathbf{0.56 \pm 0.18}$ $\mathbf{0.18} - \mathbf{0.90}$

Table 5.8.4: Camera centre error statistics in cm for the robustness experiment based on the **Lego-PAMI-TT**$^{\times 1}_{\text{SL}}$ and applied to **Kell13**, improved version of **Kell13** and the fully curvature enhanced pipeline [LKS+17]. $1 : n$ indicates that every nth frame is used for reconstruction. **Bold numbers** indicate the lowest error in its category and $\emptyset$ refers to a complete failure of the camera tracker.

5.8.4 Scalability

The proposed approach is compared to Keller et al. [KLL+13] and Niessner et al. [NZIS13] on the large scene **mit_76-417b**$_{\text{SL}}$ by Xiao et al. [XOT13].

For **Nies13**, the voxel size was changed from 4 mm to 10 mm. With 4 mm, **Nies13** stopped adding depth data in the beginning of the scene. For **Kell13**, the removal of dynamic parts of the scene is disabled.

Figure 5.8.7 shows the reconstructions for the **mit_76-417b**$_{SL}$ sequence by Xiao et al. [XOT13]. Using CloudCompare [GM13], the reconstructions are aligned and rendered using EDL (Eye-Dome Lighting).

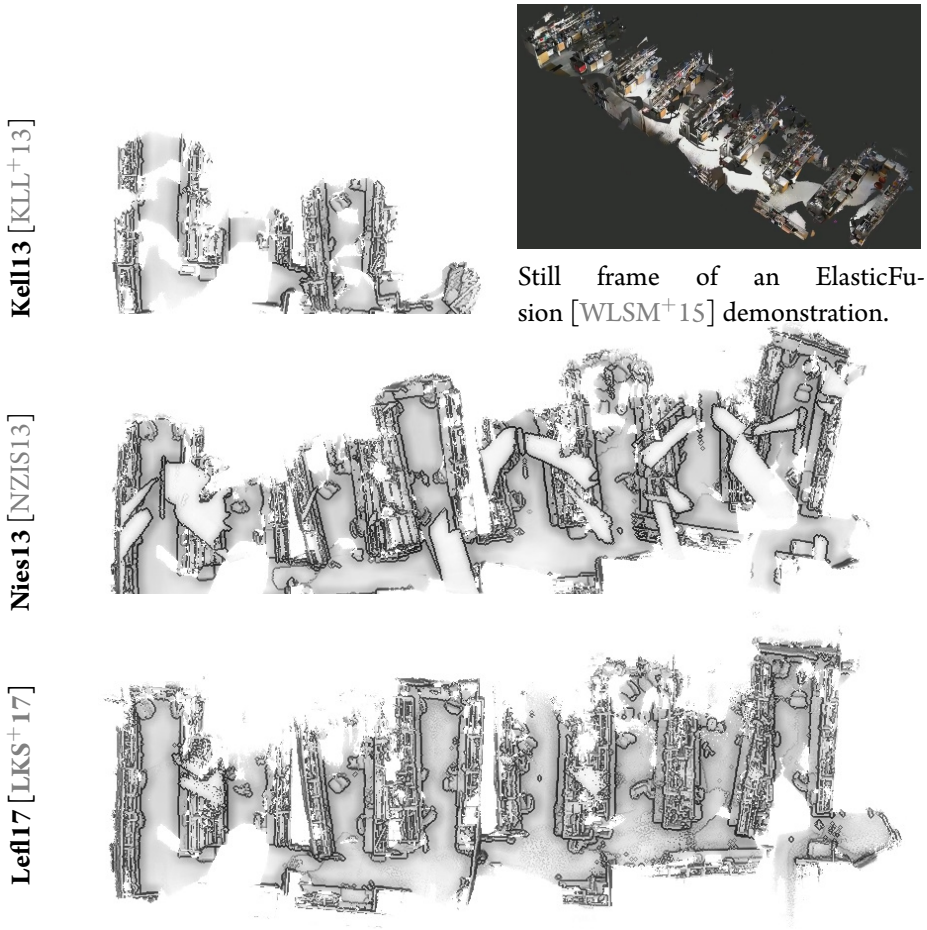


Figure 5.8.7: Comparison of reconstructions for the **mit_76-417b**$_{SL}$ sequence by Xiao et al. [XOT13].

Kell13 lost ICP tracking and could not recover. See the inset for a reconstruction with ElasticFusion, as shown in their video available at the following link[5].

On this challenging dataset that is prone to drift over large scales, the proposed method generally performs as well as, or better than, **Nies13**. As far as visible in the video, ElasticFusion, which employs non-rigid surface deformation, preserves rectilinearity of the office isles better; we note that their deformation scheme is a design decision orthogonal to the incorporation of curvature, and it is conceivable that both could be combined.

Lastly, note that the proposed reconstruction consists of $24{,}630{,}119$ surfels, while the **mit_76-417b**$_{SL}$ sequence consists of $11{,}158$ frames. This demonstrates the space-efficiency of the point-based-fusion scheme,

[5]Video results of ElasticFusion [WLSM+15]: `https://youtu.be/-dz_VauPjEU?t=3m52s`.

where each input frame on average contributes only 2,207 points to the final model.

5.8.5 Live Results

The reader is referred to the following link[6] presenting the advantages of using curvature information for online 3-D reconstruction application. The video (see Figure 5.8.8 for a thumbnail) shows the improvements of stability and robustness of the camera tracking module compared to state-of-the-art approaches.

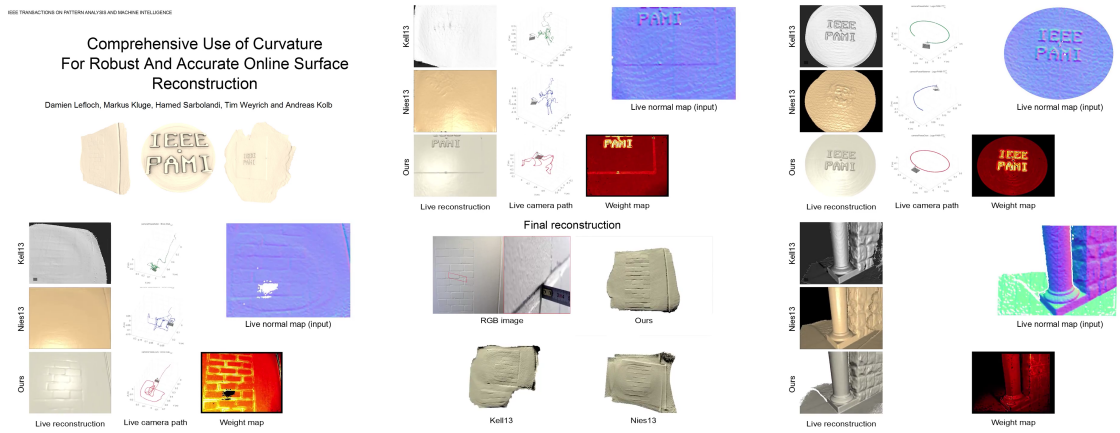


Figure 5.8.8: Still images extracted from the video results of this work [LKS+17].

5.9 Discussion

This chapter presents a novel real time, point-based reconstruction framework for robust surface extraction using curvature as an independent quantity. The proposed approach significantly reduces drift, thus improving camera tracking and reconstruction quality. Particularly, this method is able to robustly reconstruct scenes with very low depth-feature information, not possible with state-of-the-art methods. Finally, a new benchmark data set was built that provides ground-truth camera poses and geometry using both Kinect cameras, supporting further research in the field.

The proposed approach is still limited in not being able to successfully reconstruct scenes with few depth features, when the input depth maps have a high noise level. Fig. 5.8.3, bottom row, shows such a failure case for the **Brick-Wall**$_{SL}$ sequence, for which the brick structure is still discernible in the individual input frame of the Kinect^{SL} camera.

[6] Video results of Curvature-Based Fusion [LKS+17]: https://www.youtube.com/watch?v=o6-GN7DbynA.

Finally, this method does not solve explicitly loop closure, only reduce the overall camera drift, thus problems could still occur for large-scale scenarios.

*One person's **data** is another person's **noise**.*

K.C. Cole ($*1946$)

6

Anisotropic Point-Based Fusion

This chapter describes a new 3-D reconstruction framework that takes into account the intrinsic camera noise model to better fuse data into a single model. To do so, new information is stored into the global representation that describes the anisotropic behaviour of depth sensor noise. Since this information is using memory storage per model points, a new compression scheme has been proposed to reduce drastically the total memory footprint of the global model.

6.1 Introduction and Prior Work

As already seen in Chapter 4, many online 3-D reconstruction systems share a three-stage process, consisting of the following components:

1. **Depth Map Pre-processing:** The range map delivered by the Kinect camera is pre-processed, e.g., using bilateral filtering, and additional data such as normals are estimated for each range map pixel.
2. **Camera Pose Estimation:** Based on the current observation and the so far accumulated model, the camera pose is estimated using an ICP approach [BM92].
3. **Depth Map Fusion:** In this step the registered input range map is accumulated into the existing model representation.

One aspect that has insufficiently been addressed so far is the *anisotropic nature* of the input data. The spatial uncertainty of an individual pixel of the input range map is determined by two factors:

a) the *lateral pixel extend* which is given by the lateral resolution of the camera chip and the intrinsic parameters of the camera, i.e, focal length, principal point and lens distortion, in combination with the depth value, i.e. the distance from the camera, and

b) the *depth noise* of the sensor, which itself strongly depends on the underlying range measurement principle.

There are already some works on noise models for range devices, e.g., for ToF cameras such as the new generation of the Kinect camera. Falie et al. [FB07] present a noise model based on phenomenological considerations, which predicts a range error as a function of the amplitude and the distance value of a specific pixel. For an overview of denoising approaches for ToF cameras, refers to the survey of Lenzen et al. [LKS+13]] (see also Section 2.4). Often simple Gaussian noise models are assumed, e.g., in the context of motion capturing [GPKT10].

So far, the anisotropy of the input data has not been considered in the context of real time scene acquisition. However, Maier-Hein et al. [MHFdS+12] introduce a method in order to improve ICP-based registration of ToF range maps with respect to a given polygonal model in the context of medical applications.

This chapter presents a new real time framework, for efficient reconstruction of large-scale scenery incorporating the anisotropy of the input data. The proposed system uses an enhanced point-based representation similar to Keller et al. [KLL+13], but not bound to it, which is capable in handling anisotropy in the depth map fusion step. The main contributions of this chapter are

- a novel *symmetric anisotropic distance measure* that is applied to establish more robust correspondences between input and model points in the fusion step, and
- a novel *anisotropy-aware fusion technique* for accumulation of anisotropic input data into the model,

- a *data compression scheme* for point-based model representation implying an efficient storage of attributes per-point.

Furthermore, a solid evaluation of both, the data compression scheme and the anisotropic accumulation approach, and their impact on the reconstruction quality is presented.

6.2 Anisotropic Point-based Fusion

In contrast to the work presented in Chapter 5 where curvature information is stored as a point attribute, the following section introduces a new reconstruction framework that stores additional per-point properties: the symmetric, 3×3, anisotropic noise covariance matrix $\Sigma(\boldsymbol{u})$, represented as 6 floats per point.

6.2.1 Anisotropy

So far, real time reconstruction methods with range maps have ignored the anisotropic nature of the range data. The anisotropy results from the fact, that the reliability of a 3-D point in a range map is much higher in lateral direction than in axial direction, as the lateral uncertainty is only limited by the pixel size and, due to the perspective mapping, by the distance. The axial uncertainty is defined by the noise of the acquisition device, i.e. the Kinect camera, which, for example, increases for larger object-to-camera distances. Maier-Hein et al. [MHFdS+12] model the standard deviation as a function over distance. The model for the Kinect camera proposed by Nguyen et al. [NIL12] is used in order to compute the variance of the noise based on the z-distance.

Given a covariance matrix $\Sigma_{\boldsymbol{p}}$ for a point $\boldsymbol{p} \in \mathbb{R}^3$, the *Mahalanobis distance* of any other point $\boldsymbol{q} \in \mathbb{R}^3$ can be calculated based on the inverse of the covariance matrix $\Sigma_{\boldsymbol{p}}^{-1}$, which is also called *reliability matrix*:

$$d_{\boldsymbol{p},\Sigma}(\boldsymbol{q}) = \sqrt{(\boldsymbol{q}-\boldsymbol{p})^\top \Sigma_{\boldsymbol{p}}^{-1} (\boldsymbol{q}-\boldsymbol{p})} \; .$$

the symmetric 3×3 reliability matrix $\Sigma_{\boldsymbol{p}}^{-1}$ is directly stored leading to 6 additional values per point.

Similar to Maier-Hein et al. [MHFdS+12], the data association is built before data fusion using the following anisotropic model. Whereas [MHFdS+12] uses the Malahanobis distance based on the inverse of the sum of the covariance matrix $(\Sigma_{\boldsymbol{p}} + \Sigma_{\boldsymbol{q}})^{-1}$, the presented method minimises the sum of both Malahanobis distances $d_{\boldsymbol{p},\Sigma}(\boldsymbol{p}-\boldsymbol{q}) + d_{\boldsymbol{q},\Sigma}(\boldsymbol{p}-\boldsymbol{q})$ in order to choose the best associated corresponding pair for registration. The main reason of this approach is performance. As the reliability matrix $\Sigma_{\boldsymbol{p}}^{-1}$ is stored, applying Maier-Hein et al. [MHFdS+12] would require three additional matrix inversions per point-pair comparison. Note that several experiments have been done in order to compare this simple minimisation

to the one proposed in [MHFdS+12]. All experiments were leading to the same result, i.e, same point pair. This validates the choice to keep the proposed data association for a better efficiency.

6.2.2 Anisotropic Fusion

The accumulation of range data in the anisotropic case has to consider the non-uniformity of distance measurements given by the depth sensor. Similar to the geometric fusion, the anisotropic noise model should be refined over time. Therefore, the geometric and anisotropic fusion procedures have to be reformulated by convex combinations for accumulating of the point's mean and the accumulation of the reliability matrix.

Considering two different points $\boldsymbol{p}_i$ with covariance matrices $\Sigma_{\boldsymbol{p}_i},\ i = 1, 2$, and point $\boldsymbol{q}$ lying on the line segment between $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$, a meaningful definition of an *anisotropic split ratio* β of $\boldsymbol{q}$ with respect to $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ is given by

$$\begin{aligned}\boldsymbol{q} &= \frac{d_{\boldsymbol{p}_2,\Sigma_2}(\boldsymbol{q})}{d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{q}) + d_{\boldsymbol{p}_2,\Sigma_2}(\boldsymbol{q})}\,\boldsymbol{p}_1 + \frac{d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{q})}{d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{q}) + d_{\boldsymbol{p}_2,\Sigma_2}(\boldsymbol{q})}\,\boldsymbol{p}_2 \\ &= (1-\beta)\boldsymbol{p}_1 + \beta\boldsymbol{p}_2, \text{ with } \beta = \frac{d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{q})}{d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{q}) + d_{\boldsymbol{p}_2,\Sigma_2}(\boldsymbol{q})}\ . \end{aligned} \tag{6.1}$$

To create an analogy to PBF, the points $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{q}$ are referring to the model point, the corresponding input point and the resulting merged point respectively.

Since $\boldsymbol{q}$ is the resulting merged point, the anisotropic split ratio β needs to be reformulated as given in Eq. (6.1). Defining $\boldsymbol{q}$ as affine combination $\boldsymbol{q} = (1-\alpha)\boldsymbol{p}_1 + \alpha\boldsymbol{p}_2$ for some $\alpha \in [0, 1]$ and exploiting, that the Mahalanobis distance simply scales the isotropic distance values for a given direction, from Eq. (6.1) the anisotropic split ratio becomes

$$\begin{aligned}\beta &= \frac{d_{\boldsymbol{p}_1,\Sigma_1}((1-\alpha)\boldsymbol{p}_1 + \alpha\boldsymbol{p}_2)}{d_{\boldsymbol{p}_1,\Sigma_1}((1-\alpha)\boldsymbol{p}_1 + \alpha\boldsymbol{p}_2) + d_{\boldsymbol{p}_2,\Sigma_2}((1-\alpha)\boldsymbol{p}_1 + \alpha\boldsymbol{p}_2)} \\ &= \frac{\alpha d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{p}_2)}{(1-\alpha)d_{\boldsymbol{p}_2,\Sigma_2}(\boldsymbol{p}_1) + \alpha d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{p}_2)}\ . \end{aligned} \tag{6.2}$$

Inverting Eq. (6.2), the proper affine weight α has to be applied to achieve the desired anisotropic split ratio β

$$\alpha = \frac{\beta d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{p}_2)}{(1-\beta)d_{\boldsymbol{p}_2,\Sigma_2}(\boldsymbol{p}_1) + \beta d_{\boldsymbol{p}_1,\Sigma_1}(\boldsymbol{p}_2)}\ .$$

Additionally, the *anisotropic split ratio* β is used to accumulate the point normals.

Regarding the model accumulation of the covariance represented in the same coordinate system, the approach proposed by Kerl et al. [KSSC14] is applied. They define the covariance accumulation by adding the reliability, i.e. given an input and a model covariance matrices Σ_i^{in} and Σ_i^{mod} for a corresponding input and model point for frame i, respectively, the fused covariance matrix reads

$$(\widehat{\Sigma_i^{\text{mod}}})^{-1} = (\Sigma_i^{\text{mod}})^{-1} + (\Sigma_i^{\text{in}})^{-1} \ . \tag{6.3}$$

Note that in order to transform the covariance matrix $\Sigma_{i_{\text{mod}}}^{\text{mod}}$ to the same coordinate system of the input frame Σ_i^{mod}, the following transformation is required:

$$\Sigma_i^{\text{mod}} = (\boldsymbol{R}^\top{}_{i\to\text{WC}} \cdot \boldsymbol{R}_{i_{\text{mod}}\to\text{WC}})\Sigma_{i_{\text{mod}}}^{\text{mod}}(\boldsymbol{R}^\top{}_{i\to\text{WC}} \cdot \boldsymbol{R}_{i_{\text{mod}}\to\text{WC}})^\top \ ,$$

with $\boldsymbol{R}_{i_{\text{mod}}\to\text{WC}}$ and $\boldsymbol{R}_{i\to\text{WC}}$ referring to the rotational part of the transformations $\boldsymbol{T}_{i_{\text{mod}}\to\text{WC}}$ and $\boldsymbol{T}_{i\to\text{WC}}$ to pass from local to world coordinates (WC), respectively.

6.3 Implementation

Notation: In the following, the data type nomenclature given by [CDE+14] is adopted where `uint`b refers to a positive integer with b bits representing integers on $[0, 2^{b-1}]$ and `float`b is the floating-point representation with b bits in total describing sign, mantissa and exponent.

In order to store the symmetric reliability matrix $(\Sigma_i^{mod})^{-1}$ for each point inside the proposed model representation, an efficient reduction of memory footprint for the point properties is required to preserve the scalability of the overall acquisition system. Salas-Moreno et al. [SMGKD14] propose a point-based accumulation model which directly reduces the total number of points by efficiently encoding points belonging to the same planar surface using a new planar representation. The method was shown to be robust and efficient, but it is mainly designed for indoor scenes, which comprise many planar regions.

Since the storage cost of the point-based fusion framework must be reduced for any type of data set, e.g., see Fig. 6.3.1 a scene from Zhou et al. [ZK13], a direct compression of point properties is applied. A naive way to store all required point properties such as, position, normal, radius, confidence counter and timestamp, would require 9 `float32` scalars leading to a total of 288 bits per point.

To reduce the memory footprint of the surface normal property, the method proposed by Praun et al. [PH03] designed to compress unit vectors is used efficiently. This method first maps the unit sphere to a unit octahedron that is later on unfolded to the $z = 0$ plane. This method is known as one of the best approaches to compress unit vectors rapidly and robustly. Recently a survey of unit vector compression by Cigolle et al. [CDE+14] shows that the simple octahedron compression (non-numerically optimised) using 16 bits

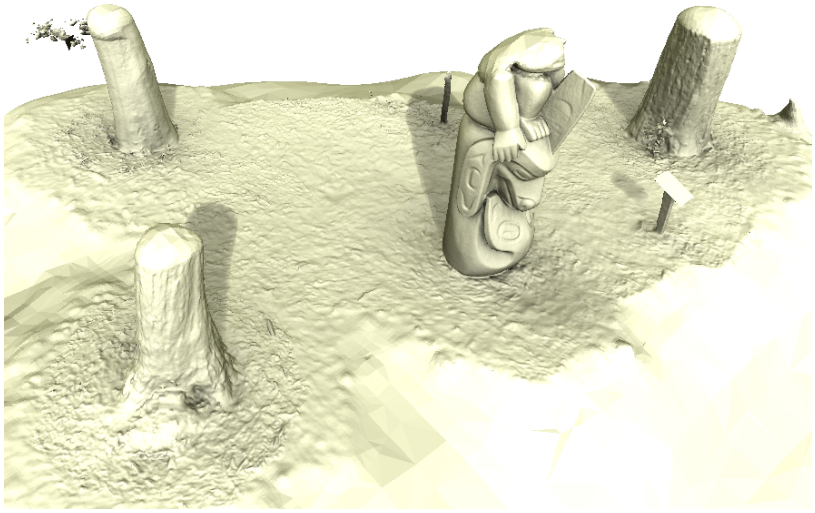


Figure 6.3.1: Advanced rendering of the extracted surface mesh given by the proposed point-based reconstruction framework (**Totempole** scene).

encoding (i.e. `enc16`) for both texture coordinates leads to a mean error angle of 0.37709° whereas the one using 32 bits (i.e. `enc32`) leads to a mean error of 0.00131°. At a first glance, a mean error of less than half a degree might appear negligible, it is shown in the following that the impact of the 8 bits encoding on the accumulation significantly coarsens the final reconstructed model. Fig. 6.3.2 gives a visual comparison of different encoding schemes applied to the **Totempole** data set.

The point position is also compressed by partially adopting the same method. First, all model points are expressed in their local coordinate referring to the camera position from which they were last observed. The original point based fusion method [KLL+13] represents the model points in world coordinates. Practically, once a fusion of an input point and model point occurs, the new average model point will be represented in the camera coordinate system of the current input frame i. This representation enables us to encode the vertices using their viewing direction and their polar distance. The same procedure can be used to encode the viewing ray as applied to the normal vector. Assuming that consumer depth cameras only provide range measurements up to a maximum radial distance of 10 meters with millimetre precision. Thus, the polar distance ρ expressed in meter can be stored on one `uint16` scalar applying the following encoding $\rho_e = \lfloor 6553.5 \times \rho \rceil$[1]. This vertex position encoding requires only $32 + 16 = 48$ bits per model point in contrast to the usual $3 \times 32 = 96$ bit storage.

The drawback of this method is that it requires to save all camera pose transformations $\boldsymbol{T}_{i \to \mathrm{WC}}$ in order

[1] $\lfloor * \rceil$ refers to the closest integer rounding operation.

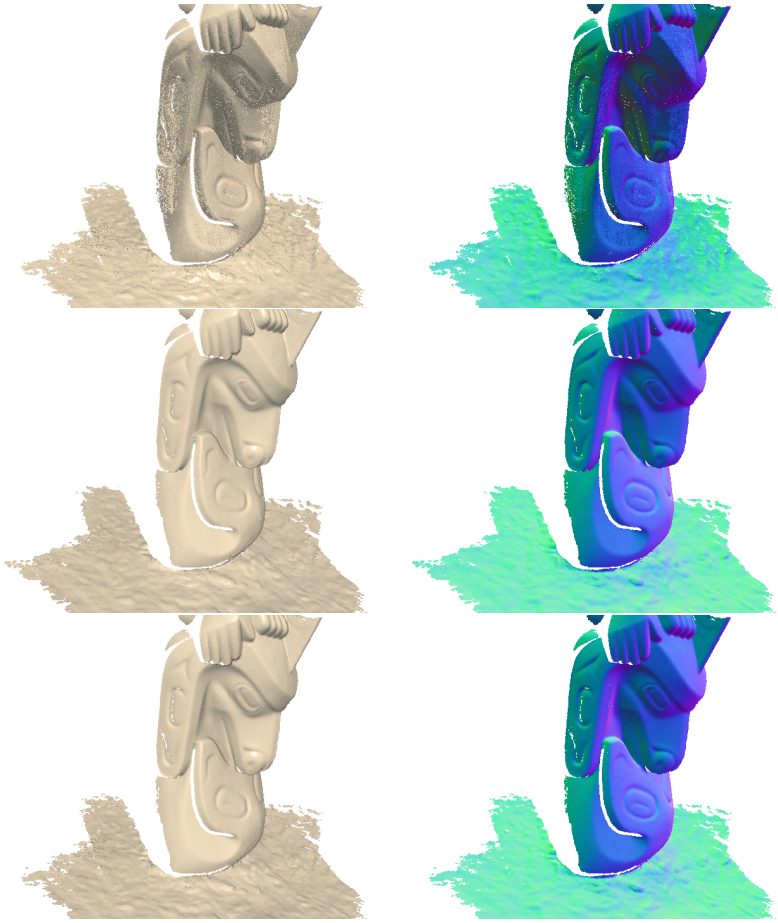


Figure 6.3.2: Comparison of three different compression schemes at frame 380 of the **Totempole** scene using the proposed SSAO surfel splatting. The compression `enc16` (top) leads to a coarser model compared to the original, uncompressed version (centre). However, `enc32` (bottom) shows negligible visual difference. The second column refers to the colour coded normal maps.

to transform all model points to a common world coordinates. Nevertheless, in Section 6.4 this additional data is shown to have little influence on the achieved compression ratio.

The remaining properties, i.e., radius, timestamp and confidence counter, are also encoded using a simple quantisation. For the timestamp t, a `uint16` scalar is chosen leading to a maximum frame id of 65535. Using a 30 Hz camera, it represents more than 30 minutes acquisition time, which is sufficient for most applications. The confidence counter is described as a `uint8` scalar since it is usually clamped with a maximum value of 255 in order to adapt to changes in the scene [NDI+11]. Similar to Weise et al. [WWLVG09], the radius

property is computed by using the following formula

$$\mathcal{R}(\boldsymbol{u}) = \delta_{\text{pix}} \times \frac{\max(s_x, s_y)}{f} \times \frac{\mathcal{D}(\boldsymbol{u})}{\mathcal{N}(\boldsymbol{u}) \cdot [0, 0, 1]^\top} , \tag{6.4}$$

where f, s_x and s_y are given by the intrinsic parameters of the camera. δ_{pix} represents the half of the pixel's diagonal $\frac{\sqrt{2}}{2}$. As seen previously, the z-distance cannot exceed 10 meters and a valid range measurement of depth camera usually occurs when the surface normal describes an oblique angle smaller than $80°$ with the camera direction [KLL+13]). Thus, a maximum radius size of an input point is defined by $r_{\text{max}} = \frac{5\sqrt{2}}{\cos(80°)} \times \frac{\max(s_x, s_y)}{f}$. Additionally, the intrinsic parameter's ratio $\frac{\max(s_x, s_y)}{f}$ can be considered to be in any case smaller than $\frac{1}{200}$ which leads to a maximum radius size of $r_{\text{max}} \approx 0.2$ meters, which is a quite conservative upper bound for real world applications. Thus, the radius is encoded as a `uint16` scalar giving $r_e = \lfloor 327.675 \times r \rceil$.

In summary, the proposed encoding results in a storage of 2 `float32`, 3 `uint16` and 1 `uint8` scalars (120 bits + **8 bits alignment cut-off**) for the set of point properties, in contrast to the naive storage of 9 `float32` scalars (288 bits), which leads to a compression ratio of $1 : 2.25$. This compression scheme leads to a negligible difference in contrast to the original point-based fusion method (see Section 6.4 for a detailed evaluation).

6.4 Results

In order to evaluate the proposed method, four different data sets are used. Two real world data sets are used to evaluate the proposed compression method without storing or processing the anisotropy. Two simulated data sets are used to obtain a quantitative comparison of the isotropic reconstruction with the novel anisotropic accumulation scheme (with compression enabled in both instances).

Totempole: This data set is provided by Zhou et al. [ZK13] and consists of 8853 RGB-D frames (≈ 5 minutes of acquisition time) given by a Kinect-like camera. Figure 6.3.1 shows the reconstructed scene given by the proposed framework. Note that for this data set only pseudo-ground-truth of camera pose and geometry is given, based on the approach by Zhou[2].

Office: This data set is provided by Kerl et al. [KSC13] and contains 2509 RGB-D frames (≈ 1.4 minutes of acquisition time) given by a Kinect-like camera, see Figure 6.4.1. The data set includes the camera path ground-truth acquired by an infrared tracking system and was designed for SLAM benchmark applications[3].

[2]Available at `http://qianyi.info/scenedata.html`

[3]Available at `http://vision.in.tum.de/data/datasets/rgbd-dataset`

Buddha: This data set is generated using a ToF simulator, which is an enhanced version of Keller and Kolb [KK09], applied to the Stanford Buddha model scaled to 3 meters height. It is composed of 237 depth frames disturbed with Gaussian noise on the computed polar distance using the formulation of Nguyen et al. [NIL12] for the Kinect^{SL} camera. This formulation relates the standard deviation of the z-distance noise to the measured distance via a second-degree polynomial and was modelled using images of planar regions located at different distances.

Statue: This second simulated data set is generated in the same way as the Buddha scene and consists of 286 frames.

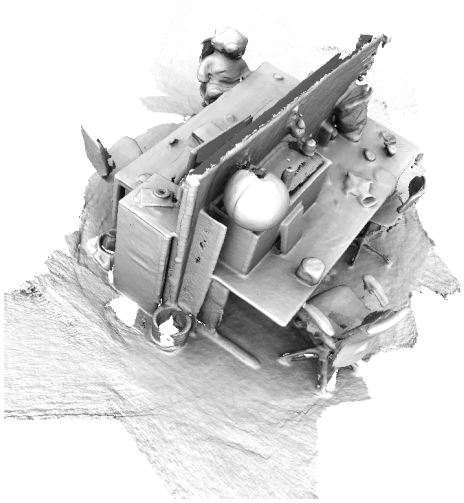


Figure 6.4.1: Overview of the **Office** scene data set from [KSC13].

6.4.1 Encoding Evaluation

In order to evaluate the proposed compression representation, two data sets are used given by the **Totempole** and the **Office** scenes. The following three representations are compared to each other:

naive storage: refers to the original PBF framework proposed by [KLL+13] (uncompressed model).

enc16: compresses normals and viewing rays in low-resolution, 16-bit representation.

enc32: compresses normals and viewing rays in high-resolution, 32-bit representation.

The **Totempole** data set is used to highlight the visual quality and the storage ratio. The proposed compression scheme retains the visual reconstruction quality if the `enc32` compression is used for unit vector representations; see Fig. 6.3.2. Concerning the storage gain, the final **Totempole** reconstructed model is composed of 7,822,519 oriented points. The naive storage method (9 `float32` scalars) will lead to a mem-

ory usage of 269 MB where the proposed compression scheme leads to a memory usage of 104.4 MB (+ **7.5 MB alignment cut-off**). However, this method requires the storage of all the camera poses (8853×12 `float32` scalars) which enlarges the memory footprint by 415 KB, i.e. by 0.4%.

The **Office** data set is used in order to quantitatively evaluate the compression scheme against the camera tracking and the reconstructed geometry quality. Fig. 6.4.2 shows the camera centre position errors computed by the ICP algorithm for the naive storage, the `enc16` and `enc32` encoding schemes. Whereas the `enc16` encoding leads to a higher error in terms of the camera pose estimation, the `enc32` encoding gives camera pose errors very close to the uncompressed method. Additionally, we evaluate the quality of the geometry model reconstructed by each compression scheme. Since no geometry ground-truth is given, a pseudo-ground-truth is generated by applying the reconstruction framework without compression using the ground-truth camera poses. This generated geometry pseudo-ground-truth is compared to three different reconstruction methods that all use the ICP algorithm to track the camera motion. Fig. 6.4.3 shows the Euclidean distance errors of the `enc16`, `enc32` and uncompressed storage. Note how negligible the difference is between the `enc32` and the naive storage. For a better view on the distance error statistics comparison, refer to Tab. 6.4.1.

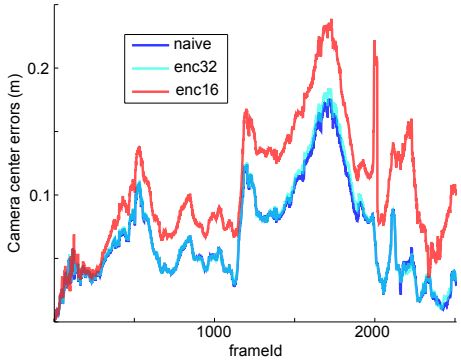


Figure 6.4.2: Camera position errors using the pose ground-truth with the naive representation and the two different compression schemes for the **Office** data set.

6.4.2 Anisotropic Fusion Evaluation

In order to evaluate the benefit of the anisotropic fusion, it is important to have proper ground-truth of the scene geometry. Therefore, simulated data sets are used, i.e., the **Buddha** and the **Statue** scenes. The proposed approach is applied on two different scenarios processing the full depth sequences with known ground-truth camera poses. Since only the anisotropic fusion is evaluated, the ground-truth camera poses

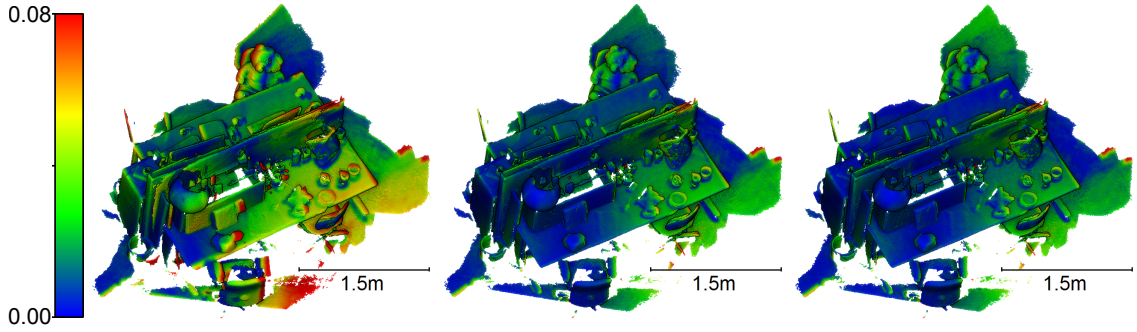


Figure 6.4.3: Colour-coding of the geometry distance errors of the **Office** scene for different compression schemes. The reconstruction using the `enc16` (left), the uncompressed (centre) and the `enc32` encoding (right). The images are generated using the CloudCompare software [GM13].

Methods	**enc16**	**naive**	**enc32**
Error Distances mean$\pm$std (mm)	24.0$\pm$18.3	12.2$\pm$10.31	13.0$\pm$10.6

Table 6.4.1: Distance error statistics for the **Office** scene experiment.

given by the ToF simulator is used in order to avoid any external error introduced by the ICP algorithm. First, the data is processed using a simple isotropic fusion as it is commonly done for KinectFusion-like approaches. Whereas the other scenario consists of processing the data sequence with anisotropic fusion. Both resulted point clouds are compared to the ground-truth mesh. For each point, the minimal distance error to all mesh faces is computed.

Fig. 6.4.5 (left) shows a close view of the point distance errors in the isotropic case, and (right) concerns the anisotropic fusion for the **Buddha** scene. One can clearly see that the anisotropic fusion noticeably reduces the overall point distance errors. Fig. 6.4.4 (left) shows the statistic of the errors depending on the confidence counter attribute, i.e. the number of point merges. For the isotropic case, the distance error of model points with a confidence counter greater than 30 is increasing. Fig. 6.4.6 visualises these points and their distance errors, which are mainly located around the lower part of the Buddha. Due to the specific camera path, this region of the scene has been observed by many frames with a comparably large range noise. Apparently, the isotropic accumulation has more difficulties with this strong anisotropy than the proposed anisotropic approach. The total mean distance errors is 1.67 ± 1.4249 mm for the isotropic fusion, whereas the anisotropic fusion leads to a total mean of 1.4856 ± 1.3452 mm.

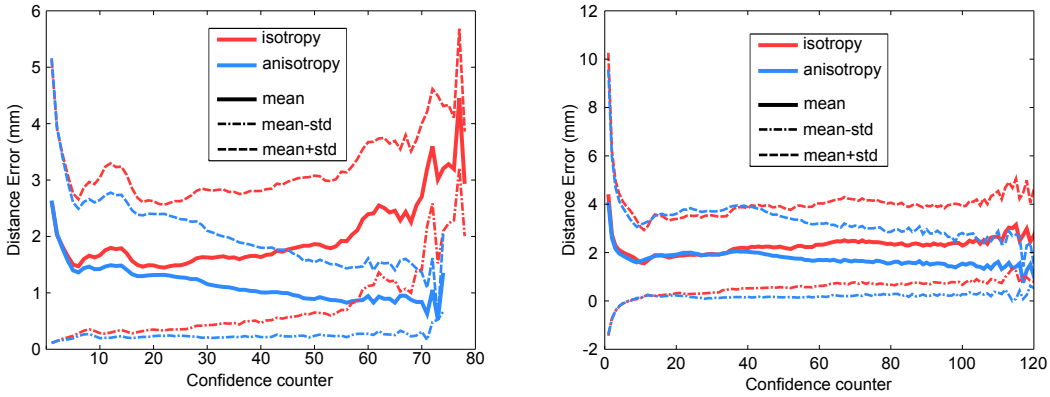


Figure 6.4.4: Comparison of distance error statistics of the **Buddha** scene (left) and the **Statue** (right) for the isotropic and anisotropic accumulation. The mean and the standard deviation are plotted. The confidence counter is related to the number of merges for the model points.

The simulated **Statue** scene confirms this observation, even though the increase of quality is less significant as for the **Buddha** scene. The error statistics in Fig. 6.4.4 (right) show a comparable error statistics for points up to 30 merges and again an improvement beyond 30 merges. The points with a confidence counter greater than 30 are shown in Fig. 6.4.7.

6.4.3 Performance

The efficiency of the method has been demonstrated by evaluating the performance of the different compression schemes and the anisotropy in isolation. Tab. 6.4.2 shows a detailed summary of the timings. Note how the compressed encoding is faster than the original method for the generation of model maps. This is easily explained by the fact that loading the compressed point attributes into a vertex buffer requires 4 floats, whereas the naive storage requires 9 floats per point. Furthermore, the anisotropy is not used during this processing which explains the similar timing with the one from the compression alone.

6.5 Conclusion

In summary, a new efficient point-based reconstruction framework was proposed that allows a better handling of anisotropic noise of range camera. A new compression scheme was introduced that allows large-scale reconstruction, reducing the final storage by half with the same performance. It is shown that this encoding does not disturb neither the camera tracking algorithm nor the quality of the reconstructed geometry. Furthermore, this chapter demonstrates that anisotropic fusion improves the overall quality of the

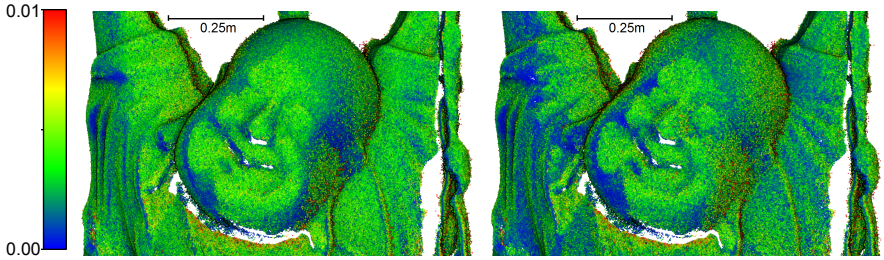


Figure 6.4.5: Colour coded error distances of the **Buddha** scene. The point distance errors to the ground-truth mesh for the isotropic fusion *(left)* and for the anisotropic fusion *(right)*, The images are generated using the CloudCompare software [GM13].

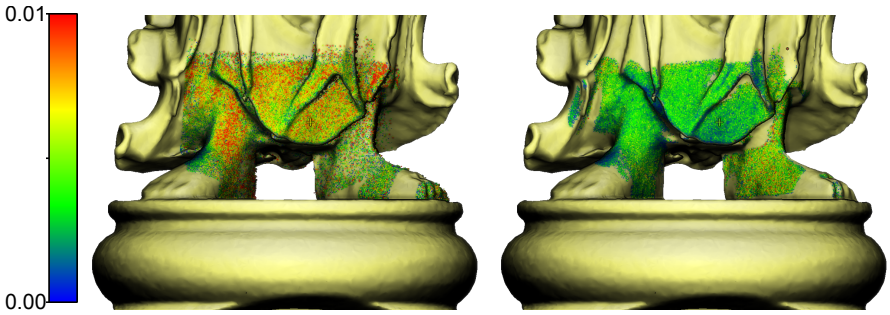


Figure 6.4.6: Colour coded error distances of the **Buddha** scene in a region with high anisotropy. Here only points that have a confidence counter greater than 30 are shown. The anisotropic accumulation *(right)* better handles this region with strong distance noise compared to the isotropic fusion *(left)*. The images are generated using the CloudCompare software [GM13].

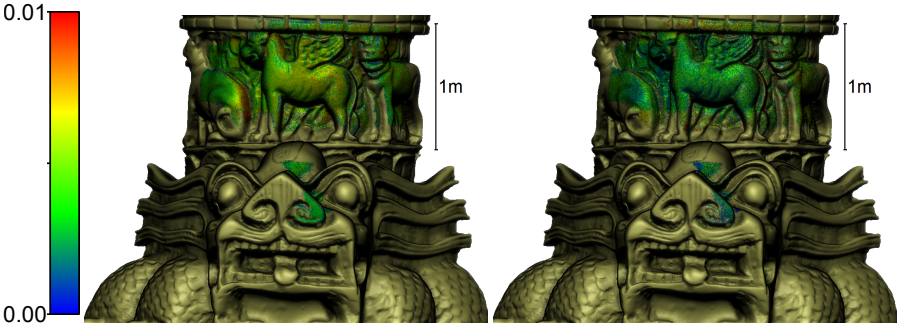


Figure 6.4.7: Colour coded error distances of the **Statue** scene in a region with high anisotropy. Here only points that have a confidence counter greater than 30 are shown. The anisotropic accumulation *(right)* better handles this region with strong distance noise compared to the isotropic fusion *(left)*. The images are generated using the CloudCompare software [GM13].

Methods (all times in msec)	**MapComp** min, max mean±std	**IdxMap** min, max mean±std	**Accum** min, max mean±std	**GenModelMaps** min, max mean±std
Naive	1.9, 6.1 3.1±0.5	0.7, 2.6 1.6±0.5	3.2, 6.7 5.1±0.3	2.1, 6.7 5.0±1.2
Encoding	1.9, 6.1 3.1±0.5	0.6, 2.6 1.4±0.4	3.4, 7.9 5.2±0.4	1.6, 4.4 3.4±0.5
Encoding + Anisotropy	1.9, 6.7 3.1±0.6	0.6, 2.6 1.8±0.4	3.5, 8.0 5.6±0.4	1.6, 4.4 3.4±0.5

Table 6.4.2: Timings given by three methods using the **Buddha** scene for four processing modules. **MapComp** (Depth map pre-processing), **IdxMap** (index map generation), **Accum** (Depth map fusion), **GenModelMaps** (Rendering). Red colours refer to the modules where the anisotropic information is used.

reconstruction.

In the planning stage of a book, don't plan the ending. It has to be earned by all that will go before it.

Rose Tremain (∗1943)

7 Conclusion

RANGE imaging systems have gained more attractions these recent years. Originally coming from the Kinect^{SL}, new generation of depth sensors has just entered the consumer market. Cleaning or security robots, autonomous cars or virtual and augmented reality devices use range image information to better drive their algorithm. Devices such as the recent Microsoft Hololens contains several camera sensors, including a ToF-based depth camera, to capture the environment and improve Human-Computer Interface. It is essentially a fully mobile wearable device (glasses-like) that mixes real world with synthetic holograms. Recent iPhones implement Face recognition application (FaceID) using a SL-based range camera, creating a high-resolution 3-D map of the user's face. ToF depth sensors are now mature enough to be directly implemented on Phablets operating at low frame rate. However, to reduce the power consumption is still a huge challenge for mobile devices implementing ToF-based technology since the modulated illumination should be strong enough to capture reliable depth information. In this thesis, a fast

method for improving the quality of ToF-based depth sensors has been presented as well as techniques that improves environment modelling taking into account sensor noise.

7.1 Summary

Chapter 2 presented the perspective camera model and an overview of current range imaging technologies, focusing on the SL and ToF principles. A comprehensive description of error sources of ToF sensors has been given, with still open problems that are not yet fully understood, such as the intensity-related error. The chapter concluded with the importance of denoising methods for range data and a comparison of common denoising 2-D convolution filters and their influence on the surface features of the input depth data.

In Chapter 3, a real time method for reducing motion artifacts of ToF depth data has been proposed. The method was evaluated with simulated data given by a realistic ToF camera simulation and shows the importance of such algorithm in real use case, such as hand gesture recognition.

Chapter 4 introduced online and dense real time 3-D reconstruction methods with a focus on the point-based fusion representation since it drives the main work of this thesis. Individual modules used in the 3-D reconstruction pipeline were described in detail. Advantages and drawbacks of these different representations have been discussed. The chapter concluded with a new method to handle the reconstruction of static environments with dynamic objects and showed how it drastically improves the camera ego motion estimation, compared to the original KinectFusion framework.

Both Chapter 5 and Chapter 6 presented new 3-D reconstruction methods, which better handle the strong noise of depth sensors. Chapter 5 described a 3-D reconstruction framework that computes complete surface curvature information to build better point correspondences used in the tracking and fusion processes. To reduce performance losses of point-based representation [KLL+13], in the case of reconstructing large-scale environments with millions of points, a *deep index map* was introduced to efficiently fuse model points with new incoming points and to clean up outdated model points. This chapter shows how noise is a real challenge for low feature scenes (such as brick walls) and also can influence drastically the quality of the reconstruction output. Additionally, the proposed method was compared with multiple recent state-of-the-art approaches and the chapter showed that integrating surface curvature information over time for the complete 3-D reconstruction leads to performance improvements for all main pipeline modules. Finally, in Chapter 6, another path to improve 3-D reconstruction methods was followed by considering the noise uncertainty of the depth sensor. A new compression scheme has been proposed to drastically reduce the memory usage of point-based fusion methods without decreasing the quality of the overall reconstruction pipeline. Using the compression scheme, new information has been accumulated to the model representation, such as the reliability matrix of individual measurements. In this way, better point correspondences are found, which is

a direct benefit for the tracking and merging modules. This chapter showed that using the anisotropic nature of the sensor noise led to a better overall quality of the reconstruction and avoid intrinsically to coarse the current reconstructed model by merging with less reliable data (noisier input data).

7.2 Outlook

A number of exciting research questions and engineering directions are still not fully solved, and a glimpse of those topics is now outlined.

The presented work on dense reconstruction does not tackle the challenging problem of the accumulation of sensor drift due to error on the camera ego motion estimation. In Chapter 5, the proposed method showed that improving the camera ego motion estimation could limit the drift on some special use cases (like the *stonewall* data set from Zhou and Koltun[ZK13]), but does not correct it for larger scenes. Drift in large-scale environments is an interesting topic for improvement. The point-based representation better suits the loop closure correction due to its intrinsic representation rather than resampling a dense voxel grid. As seen in Chapter 6, each model points can be expressed in camera coordinates (either the original camera coordinates where the point was firstly seen or conversely the newest camera coordinates where the point was lastly fused). Applying loop closure correction directly on the camera path itself, would result in correcting all model points. A post processing is required to properly close the 3-D model, but should be applied locally on faulty regions and does not require to correct the complete model representation composed of millions of points. Whelan et al. [WLSM+15] have proposed an approach, named ElasticFusion, which extends the point-based fusion method by applying online loop closure.

Chapter 5 focuses on the difficulty to properly reconstruct a low feature scene with noise in the input data. But the method did not tackle the noise uncertainty of the sensor. However, Chapter 6 incorporates the anisotropic nature of sensor noise to the camera tracking and fusion processes. It demonstrates that more precise reconstruction can be achieved using noise uncertainty on synthetic data. It is of big interest to combine both methods and evaluate the overall quality. Unfortunately this combination is not straightforward due to the required additional information per model point which increase the memory footprint usage; thus a new compression scheme needs to be introduced. Recently, Cao et al. [CKH18] show that assigning a probabilistic uncertainty model to each depth measurement, which then guides the scan alignment and depth fusion, allows high accuracy 3-D reconstruction of real data sequences. An additional focus on improving the overall quality of the reconstruction is to consider the colour information as a useful information. Zhou et al. [ZK14] proposed a method to reconstruct the environment with high-resolution texture colour, improving the final rendering quality of the reconstructed model.

Finally, dense reconstruction of very large scale environments such as the indoor of several buildings is still

not solved mainly due to the memory usage and the real-time performance constraint. To increase the usability of the voxel grid-based fusion, Nießner et al. [NZIS13] introduced a bi-directional CPU-GPU streaming combined with a hash voxel function to bypass the limited GPU memory. However, this method is still not suitable for high-resolution reconstruction of very large scenes. Approaches that understand better the semantics of the current environments are of a huge interest. With the availability of NVidia GPUs containing new TensorCores specially designed for convolutional neural network, using deep learning for real-time dense 3-D reconstruction is a valid direction. Recent works have been proposed to first create huge data set of 3D objects with semantic labels [DCS+17] to facilitate the training process of neural nets. Very recently, Dai et al. [DRB+18] use this data set to create a complete reconstruction from a sparse volumetric grid input.

A

Appendix

A.1 Additional details on Surface Attributes Estimation

A.1.1 Surface Position

Given the intrinsic camera matrix $\boldsymbol{K}$, the depth map $\mathcal{D}^t$ is transformed into a corresponding vertex map $\mathcal{V}^t$, by converting each depth sample $\mathcal{D}^t(\boldsymbol{u})$ into a vertex position $\mathcal{V}^t(\boldsymbol{u}) = \mathcal{D}^t(\boldsymbol{u})\boldsymbol{K}^{-1}(\boldsymbol{u}^\top, 1)^\top \in \mathbb{R}^3$ in camera space. This is derived from the Pinhole camera model which is a strong simplification of the optical lens properties. For a better point cloud estimation, the map $\mathcal{D}^t$ is previously un-distorted if required.

Figure A.1.1 shows the 3-D point cloud computation given by the raw depth map and the smoothed depth map applying a bilateral filter. A kernel radius of 5 is used here to set the bilateral filter with a spatial sigma of $\sigma_s = 2.5$ px and the range sigma $\sigma_r = 0.05$ m.

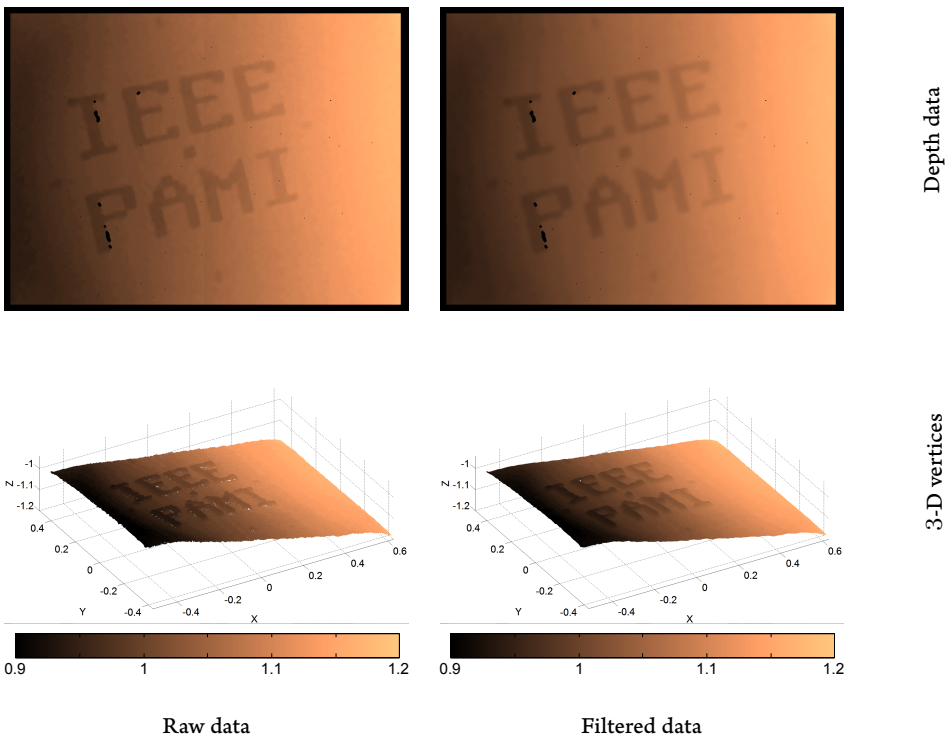


Figure A.1.1: An example of vertex map estimation for one of our data set at frame 361 (acquired with a Kinect^{SL} camera). The first column refers to the Raw depth (where outliers were filtered out), and the second column to the same depth map smoothed via a bilateral filter.

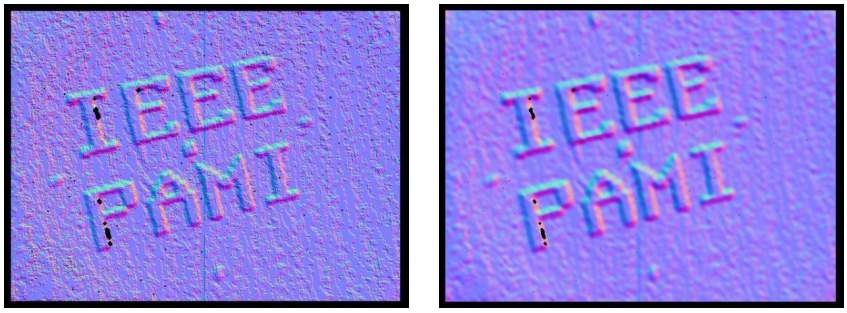
A.1.2 Surface Normal

The surface normal vector $\boldsymbol{n}_i$, also simply known as normal, is the vector which is perpendicular to the surface at a given point $\boldsymbol{v}_i = (X_i, Y_i, Z_i)^\top$. Let $\hat{\boldsymbol{n}}_i = \frac{\boldsymbol{n}_i}{||\boldsymbol{n}_i||}$ be the unit normal vector. The normal at a point (X_0, Y_0) of a surface $Z = \mathcal{S}(X, Y)$ is given by:

$$\boldsymbol{n}_0 = \begin{pmatrix} 1 \\ 0 \\ \mathcal{S}_X(X_0) \end{pmatrix} \times \begin{pmatrix} 0 \\ 1 \\ \mathcal{S}_Y(Y_0) \end{pmatrix} = \begin{pmatrix} -\mathcal{S}_X(X_0) \\ -\mathcal{S}_Y(Y_0) \\ 1 \end{pmatrix}, \tag{A.1}$$

where $\mathcal{S}_X = \frac{\partial \mathcal{S}}{\partial X}$ and $\mathcal{S}_Y = \frac{\partial \mathcal{S}}{\partial Y}$.

Many techniques exist to compute surface normals from noisy range data (see [KAWB09] for a recent survey). However, as described by Newcombe et al. [NDI+11], depth sensors measure and discretise the



Raw normal map Filtered normal map

Figure A.1.2: An example of normal map estimation using a raw vertex map and a filtered one via the central difference.

surface on a regular grid, thus a simple cross product between the 4-connected vertices $\{\mathcal{V}^t(\boldsymbol{u}_\text{l}), \mathcal{V}^t(\boldsymbol{u}_\text{r}), \mathcal{V}^t(\boldsymbol{u}_\text{t}), \mathcal{V}^t(\boldsymbol{u}_\text{b})\}$ (*left, right, top, bottom*) is used to estimate the surface normal:

$$\mathcal{N}^t(\boldsymbol{u}) = \widehat{\boldsymbol{n}}_{\boldsymbol{u}}, \text{where}$$

$$\boldsymbol{n}_{\boldsymbol{u}} = (\mathcal{V}^t(\boldsymbol{u}_\text{r}) - \mathcal{V}^t(\boldsymbol{u}_\text{l})) \times (\mathcal{V}^t(\boldsymbol{u}_\text{b}) - \mathcal{V}^t(\boldsymbol{u}_\text{t}))\,.$$

Since depth maps given by range cameras are usually noisy, special care needs to be taken in order to compute robustly surface normal. In fact, as seen previously, normal vectors are computed from the first derivative of the surface, it is by definition sensible to noise. Newcombe et al. [NDI+11] show that applying on the raw depth maps, an edge-preserving smoothing filter such as the well-known bilateral filter [TM98], lead to a proper normal map estimation.

Figure A.1.2 shows an example of surface normal computation using the simple central difference method. The normal map is visualised using the following colour mapping:

$$\begin{pmatrix} r_{\boldsymbol{u}} \\ g_{\boldsymbol{u}} \\ b_{\boldsymbol{u}} \end{pmatrix} = 0.5\,[\widehat{\boldsymbol{n}}_{\boldsymbol{u}} + 2]\,.$$

The central difference algorithm leads to unreliable surface normal estimation if noisy depth data is used (see Figure A.1.2-Raw). In fact, high frequency noise is present on the normal map causing false surface properties.

Pauly et al. [PGK02] present another method to estimate the surface normal which leads to a notion of

surface curvature. It is based on the analysis of eigenvalues and eigenvectors of the covariance matrix of a local neighbourhood as introduced by Hoppe et al. [HDD+92]. Pauly et al. [PGK02] define the surface variation σ, closely related to curvature, using the eigenvalues of the covariance matrix:

$$\sigma = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}$$

Since $0 \leq \lambda_0 \leq \lambda_1 \leq \lambda_2$, the maximum surface variation σ is $\frac{1}{3}$ for the isotropic case where all λ_i are equals and not null. Figure A.1.3 (second row) shows the surface variation σ of two depth maps of a specific data set. Note how surface variation is sensitive to noise (*Raw* column) and leads to unreliable surface variation. It is also clear that the surface variation does not provide all the information of the local surface. For example, the surface variation σ is a positive entity only and does not robustly differentiate cases such as edges and corners (whereas curvature information does [see Section 5.3.2]).

A.2 Additional details on ICP

This section described in detail the camera tracking principle based on the ICP algorithm.

ICP estimates the rigid homogeneous transformation $\boldsymbol{T}$ consisting of a rotation matrix $\boldsymbol{R}(\alpha, \beta, \gamma) = \boldsymbol{R}_z(\gamma)\ \boldsymbol{R}_y(\beta)\ \boldsymbol{R}_x(\alpha)$ and a translation matrix $\boldsymbol{t}(t_x, t_y, t_z)$ that transforms the input (or source) data to the model (or target) data coordinates. The rigid-body transformation $\boldsymbol{T}$ is expressed by:

$$\begin{aligned} \boldsymbol{T} &= \boldsymbol{t}(t_x, t_y, t_z)\ \boldsymbol{R}(\alpha, \beta, \gamma) \\ &= \begin{bmatrix} r_{00} & r_{01} & r_{02} & t_x \\ r_{10} & r_{11} & r_{12} & t_y \\ r_{20} & r_{21} & r_{22} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{A.2}$$

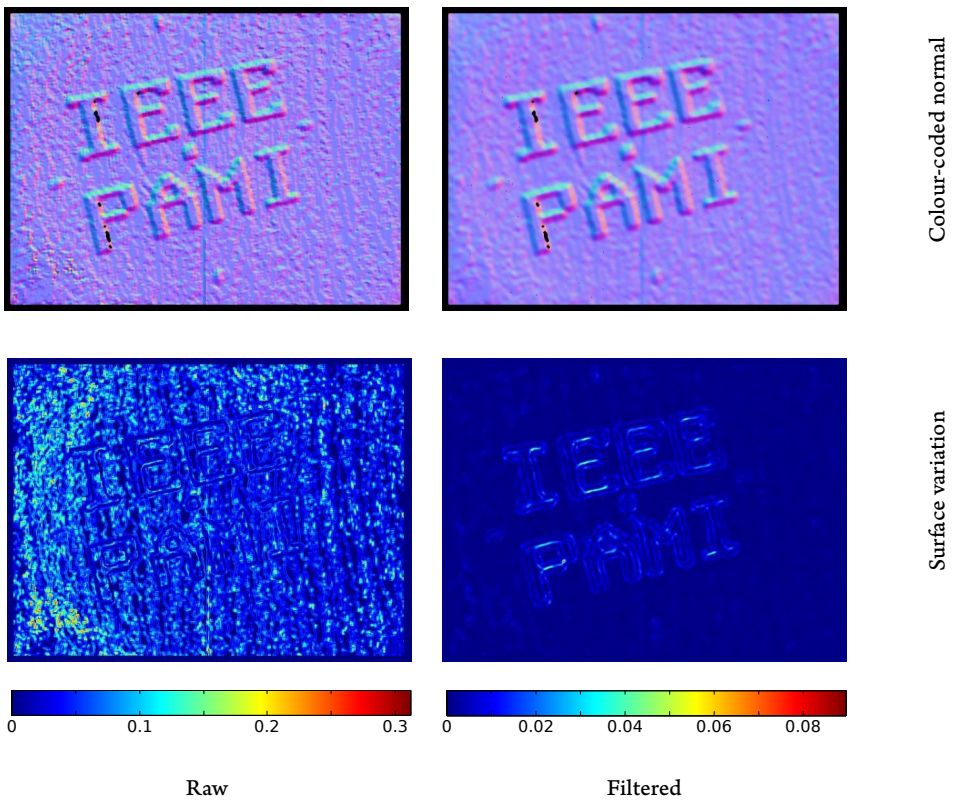


Figure A.1.3: An example of normal map estimation using a raw vertex map and a filtered one via an eigen decomposition of the covariance matrix. The second row refers to surface variation in the surface normal direction as proposed by Pauly et al. [PGK02].

with

$$
\begin{aligned}
r_{00} &= \cos\beta\cos\gamma \\
r_{01} &= -\cos\alpha\sin\gamma + \sin\alpha\sin\beta\cos\gamma \\
r_{02} &= \sin\alpha\sin\gamma + \cos\alpha\sin\beta\cos\gamma \\
r_{10} &= \cos\beta\sin\gamma \\
r_{11} &= \cos\alpha\cos\gamma + \sin\alpha\sin\beta\sin\gamma \\
r_{12} &= \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\
r_{20} &= -\sin\beta \\
r_{21} &= \sin\alpha\cos\beta \\
r_{22} &= \cos\alpha\cos\beta
\end{aligned}
\tag{A.3}
$$

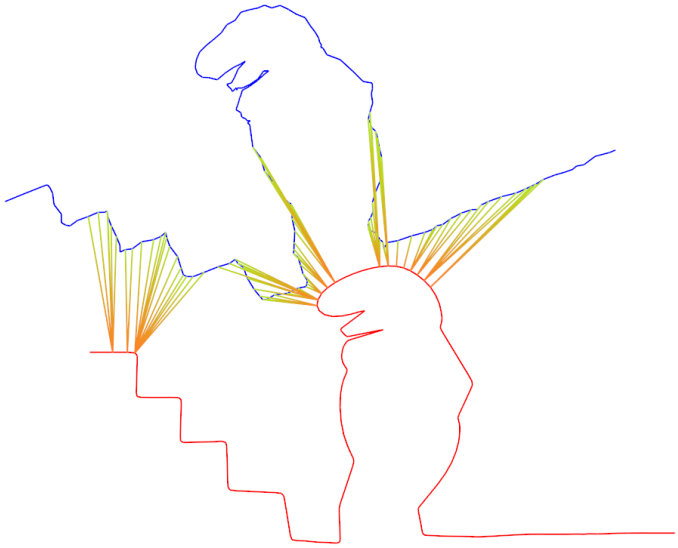


Figure A.2.1: Example of the brute-force ICP correspondences search (closest points) between the model curve (red) and the noisy input curve (blue). Note that here the correspondence set is trimmed using a maximum distance threshold.

Section 4.1.4, two main steps of the ICP algorithm were presented:

- Correspondence search between input points and model points using the current iteration of the transformation (see Figure A.2.1).
- Compute the best transformation from the trimmed set of correspondences (see Figure A.2.2).

These steps are repeated until convergence (incremental refinement of the correspondences set and the best transformation) or the number of iterations reaches a maximum (see Figure A.2.3). In practice, the convergence is detected once the variations of the 6 DoF are small between two consecutive iterations. Note that, in the original work of Besl and McKay [BM92], the ICP algorithm was shown to terminate in a minimum. However the proof was designed only when complete sets of source and target data were used for each iteration. Having different trimmed sets at each iteration could cause an unexpected behaviour for specific configurations where this proof will no longer hold.

A.2.1 Correspondences Search

The most important step of the ICP algorithm is to extract the set of correspondences between the input and the model sets of points. The idea is to build a set of correspondences as the closest points ($\mathcal{L}_2$-norm)

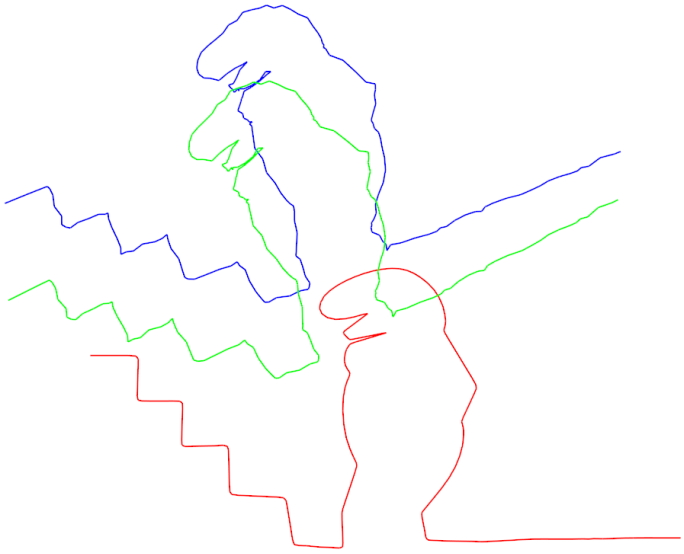


Figure A.2.2: Computation of the best transformation from the set of correspondences during the first iteration (Figure A.2.1). Green curve indicated the input curve transformed by the current best transformation.

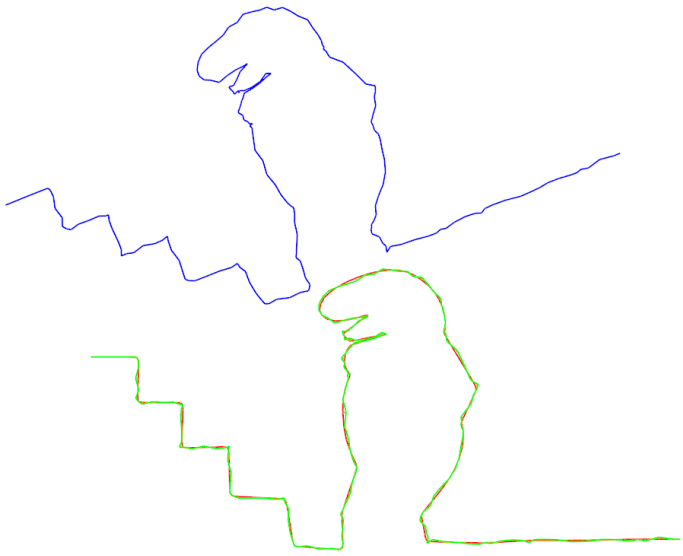


Figure A.2.3: Example of ICP convergence after 15 iterations. Note how close the green and red curves are.

between input and model points. Thus, for each input point, the algorithm will search its closest model point. This process is known to be the costliest in term of computation. In fact, the brute force method requires

to compute all possible pair combinations in order to retrieve the closest ones. Fortunately, data structures (such as kd-tree [FBF77]) allows to decrease the number of computed distances drastically compared to the brute-force approach as proposed by Zhang [Zha94].

Input data from depth cameras is intrinsically organised as a grid structure. Thus, the costly task of correspondences search can be reduced to a simple perspective search, also known as projective data association [BL95] (projected input point and corresponding model point should lie on similar 2-D pixel coordinates). Having online 3-D reconstruction using handheld motion, it is reasonable to assume that the motion between two consecutive frames is small. This assumption is crucial to allow the use of projective data association. The reason is that ICP algorithm initialised the transformation as an identity matrix, leading to the same pixel coordinate for the input and model ($\boldsymbol{u} = \boldsymbol{u}^*$). If the camera motion was too high, the projective data association will result on completely wrong surface correspondences leading to a failure of the algorithm or a localised minimal solution.

In order to avoid as most as possible influence of outliers during the minimisation process, pairs that do not match to the same part of the surface are rejected. A first condition checks the simple Euclidean distance between the source point and its corresponding model point. Practically, if the distance between the pair of points is greater than a threshold R, i.e. $||\boldsymbol{T}^{t\rightarrow(t-1)}\ \mathcal{V}^t(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)|| > R$, then the pair of points will be marked as outlier. Analogously, a pair of points is considered as an outlier if their respective surface normals are not similar. If the dot product between both normals is smaller than the cosine of the angle threshold θ_n, i.e. $\langle \boldsymbol{R}^{t\rightarrow(t-1)}\ \mathcal{N}^t(\boldsymbol{u}), \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^*)\rangle >= \cos\theta_n$, the pair is rejected.

If the exact set of correspondences is known a priori, the correct transformation can be directly retrieved (in one iteration). But since they are usually not known, an iterative method must be applied to refine the correspondences set resulting in a better transformation solution. Note that this minimum is not guaranteed to be optimal since the objective function has no reason to be convex.

A.2.2 Minimisation

Having the current set of correspondences, the optimal current transformation that minimises a certain error metric can be retrieved. The following describes two different error metrics (the point-to-point error metric and the point-to-plane error metric) and how to find the optimal transformation in a fast and practical manner.

A.2.2-i Point-to-Point

Point-to-point error metric simply describes the $\mathcal{L}_2$-norm between each pair of input and model points. The following cost energy describes the point-to-point error metric:

$$E_{point}(\boldsymbol{T}^{t\to(t-1)}) = \sum_{\boldsymbol{u}\in\mathcal{S}} \left\| \boldsymbol{T}_l^{t\to(t-1)} \mathcal{V}^t(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*) \right\|^2, \tag{A.4}$$

where $\mathcal{S}$ refers to the subset of all source points $\mathcal{V}^t(\boldsymbol{u})$ for which a valid correspondence has been found in the target set as point $\mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)$, and l refers to the current iteration of the minimisation process.

A.2.2-II Point-to-Plane

The following cost energy describes the point-to-point error metric:

$$E_{plane}(\boldsymbol{T}^{t\to(t-1)}) = \sum_{\boldsymbol{u}\in\mathcal{S}} \left\langle \boldsymbol{T}_l^{t\to(t-1)} \mathcal{V}^t(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*) \, , \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^*) \right\rangle^2 \tag{A.5}$$

Analogously to Equation A.4, $\mathcal{S}$ refers to the subset of all source points $\mathcal{V}^t(\boldsymbol{u})$ for which a valid correspondence has been found in the target set as point $\mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^*)$, and l refers to the current iteration of the minimisation process.

Minimising the cost energy E_{plane} would also require a non-linear optimisation approach due to the non-linearity of the rotation matrix (see Equation A.2). However, some techniques have already been proposed to accelerate the computation of the optimal transformation for one iteration l via valid assumptions. A linear approximation of the rotation matrix is possible if the rotation angles α, β and γ are smalls (which is already assumed since most of the methods use the fast-projective data association described in Section A.2.1). The following describes this approximation, leading to a direct least square closed-form solution.

Linear approximation Using the first-order Taylor series approximation of both sine and cosine functions for small angles

$$\begin{aligned} \sin\theta_\epsilon &= \theta_\epsilon - \tfrac{\theta_\epsilon^3}{3} + \tfrac{\theta_\epsilon^5}{5} + \ldots \\ \cos\theta_\epsilon &= 1 - \tfrac{\theta_\epsilon^2}{2} + \tfrac{\theta_\epsilon^4}{4} + \ldots \end{aligned} \tag{A.6}$$

The rotation matrix from Equation A.2 becomes:

$$\boldsymbol{R}(\alpha,\beta,\gamma) \approx \begin{bmatrix} 1 & -\gamma+\alpha\beta & \beta+\alpha\gamma & 0 \\ \gamma & 1+\alpha\beta\gamma & -\alpha+\gamma\beta & 0 \\ -\beta & \alpha & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.7}$$

This first-order approximation of the rotation matrix is still not enough for a complete linearised solution, a second approximation step is applied by simply omitting the product of small angles (i.e. $\theta_{\epsilon}^{1} \cdot \theta_{\epsilon}^{2} \approx 0$), leading to:

$$\boldsymbol{R}(\alpha,\beta,\gamma) \approx \begin{bmatrix} 1 & -\gamma & \beta & 0 \\ \gamma & 1 & -\alpha & 0 \\ -\beta & \alpha & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \hat{\boldsymbol{R}}(\alpha,\beta,\gamma) \tag{A.8}$$

Substituting $\hat{\boldsymbol{R}}(\alpha,\beta,\gamma)$ to the point-to-plane energy function E_{plane}, and after simple calculations, the cost function becomes:

$$\begin{aligned} E_{plane}(\boldsymbol{T}^{t\rightarrow(t-1)}) \approx \sum_{\boldsymbol{u}\in\mathcal{S}} \big(& \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^{*}) \cdot [\mathcal{V}^{t}(\boldsymbol{u}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^{*})] \\ & + (\alpha,\beta,\gamma)^{\top} \cdot (\mathcal{V}^{t}(\boldsymbol{u}) \times \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^{*})) \\ & + (t_x,t_y,t_z)^{\top} \cdot \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}^{*}) \big)^{2}. \end{aligned} \tag{A.9}$$

The cost function is now fully linear and is composed of 6 parameters α, β, γ and t_x, t_y, t_z. A linear system of the form $\boldsymbol{A}\,\boldsymbol{x} = \boldsymbol{b}$ can be extracted from each pair of correspondences $[\mathcal{V}^{t}(\boldsymbol{u}), \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}^{*})]$ which built a single line of the linear system. The unknown vector $\boldsymbol{x}$ is a 6-D vector and equals $(\alpha\ \beta\ \gamma\ t_x\ t_y\ t_z)^{\top}$. The $|\mathcal{S}|$-D vector $\boldsymbol{b}$ is built from the constant values of Equation A.9 (first term):

$$\boldsymbol{b} = \begin{pmatrix} -\mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_1^{*}) \cdot [\mathcal{V}^{t}(\boldsymbol{u}_1) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}_1^{*})] \\ -\mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_2^{*}) \cdot [\mathcal{V}^{t}(\boldsymbol{u}_2) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}_2^{*})] \\ \vdots \\ -\mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_{|\mathcal{S}|}^{*}) \cdot [\mathcal{V}^{t}(\boldsymbol{u}_{|\mathcal{S}|}) - \mathcal{V}_{\mathcal{M}}(\boldsymbol{u}_{|\mathcal{S}|}^{*})] \end{pmatrix} \tag{A.10}$$

And finally, the matrix $\boldsymbol{A}$ is a $|\mathcal{S}| \times 6$ is:

$$\boldsymbol{A} = \begin{pmatrix} (\ \mathcal{V}^t(\boldsymbol{u}_1) \times \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_1^*)\ )^\top & (\ \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_1^*)\ )^\top \\ (\ \mathcal{V}^t(\boldsymbol{u}_2) \times \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_2^*)\ )^\top & (\ \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_2^*)\ )^\top \\ \vdots & \vdots \\ (\ \mathcal{V}^t(\boldsymbol{u}_{|\mathcal{S}|}) \times \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_{|\mathcal{S}|}^*)\ )^\top & (\ \mathcal{N}_{\mathcal{M}}(\boldsymbol{u}_{|\mathcal{S}|}^*)\ )^\top \end{pmatrix}. \tag{A.11}$$

$\boldsymbol{x}$ is retrieved using the usual least-squares approach via $\boldsymbol{x} = \left(\boldsymbol{A}^\top \boldsymbol{A}\right)^{-1} \boldsymbol{A}^\top \boldsymbol{b}$. One important remark needs to be considered regarding the incremental transformation $\boldsymbol{T}_l^{\text{inc}}$ that is retrieved directly from the corresponding least-square solution vector $\boldsymbol{x}_l^{\text{inc}}$. Since the ICP algorithm is an iterative process, the current optimal transformation $\boldsymbol{T}_l^{t\rightarrow(t-1)}$ is continuously refined by the incremental transformation as follows:

$$\boldsymbol{T}_l^{t\rightarrow(t-1)} = \boldsymbol{T}_l^{\text{inc}}\ \boldsymbol{T}_{l-1}^{t\rightarrow(t-1)}, \quad \forall\, l \in [1, 2, \cdots, L], \tag{A.12}$$

where $\boldsymbol{T}_0^{t\rightarrow(t-1)}$ being a 4×4 identity matrix. Note that the incremental rotation $\boldsymbol{R}_l^{\text{inc}}$ matrix given by the solution $\boldsymbol{x}_l^{\text{inc}}$ should be computed using Equation A.2 and not by $\widehat{\boldsymbol{R}}_l^{\text{inc}}$ from Equation A.8 since the transformation should have all the properties of a rotation matrix.

The computation of the positive symmetric matrix $\boldsymbol{A}^\top \boldsymbol{A}$ (21 unique values) and the vector $\boldsymbol{A}^\top \boldsymbol{b}$ (6 additional values) is perfectly suitable for a modern GPU where efficient reduction techniques are available.

Finally, an additional advantage to use the least-square closed-form is that specific weight could be easily plugged into the least-square system without increasing the complexity. Each correspondence pair is associated to one weight building a single line of the least-square system. Note that fixing a weight of 0 for an outlier pair is an elegant way of rejecting it.

References

[Bar64] Richard Barakat. Application of the sampling theorem to optical diffaction theory. *Journal of the Optical Society of America A*, 54(7), 1964.

[BK08] Christian Beder and Reinhard Koch. Calibration of focal length and 3d pose based on the reflectance and depth image of a planar object. *International Journal of Intelligent Systems Technologies and Applications*, 5(3-4):285–294, 2008.

[BL95] Gérard Blais and Martin D. Levine. Registering multiview range data to create 3d computer objects. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 17(8):820–824, 1995.

[BM92] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992.

[BR07] Benedict J Brown and Szymon Rusinkiewicz. Global non-rigid alignment of 3-d scans. In *ACM Transactions on Graphics*, volume 26, page 21. ACM, 2007.

[BSK04] M. Botsch, M. Spernat, and L. Kobbelt. Phong splatting. In *Proceedings of Eurographics Symposium on Point-Based Graphics*, pages 25–32, Zurich, Switzerland, jun 2004.

[CBI13] Jiawen Chen, Dennis Bautembach, and Shahram Izadi. Scalable real-time volumetric surface reconstruction. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 32(4):113:1–113:16, July 2013.

[CDE+14] Zina H Cigolle, Sam Donow, Daniel Evangelakos, Michael Mara, Morgan McGuire, and Quirin Meyer. A survey of efficient representations for independent unit vectors. *Journal of Computer Graphics Techniques (JCGT)*, 3(2):1–30, April 2014.

[CKH18] Yan-Pei Cao, Leif Kobbelt, and Shi-Min Hu. Real-time high-accuracy three-dimensional reconstruction with consumer rgb-d cameras. *ACM Transactions on Graphics*, 37(5):171:1–171:16, September 2018.

[CL96] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of Computer Graphics & Interaction Techniques*, pages 303–312, 1996.

[CM92] Yang Chen and Géerard Medioni. Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992.

[CSC+10] Yan Cui, Sebastian Schuon, Derek Chan, Sebastian Thrun, and Christian Theobalt. 3D shape scanning with a Time-of-Flight camera. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1173–1180. IEEE, 2010.

[CZ09] ZhangLin Cheng and XiaoPeng Zhang. Estimating differential quantities from point cloud based on a linear fitting of normal vectors. *Science in China Series F: Information Sciences*, 52(3):431–444, 2009.

[DCC+08] Adrian A Dorrington, Michael J Cree, Dale A Carnegie, Andrew D Payne, Richard M Conroy, John P Godbaz, and Adrian PP Jongenelen. Video-rate or high-precision: a flexible range imaging camera. In *Journal of Electronic Imaging*, pages 681307–681307. International Society for Optics and Photonics, 2008.

[DCS+17] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas A Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, page 10, 2017.

[DGB03] James Davis and Hector Gonzalez-Banos. Enhanced shape recovery with shuttered pulses of light. In *IEEE Workshop on Projector-Camera Systems*, 2003.

[DKD+16] Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. Fusion4d: Real-time performance capture of challenging scenes. *ACM Transactions on Graphics*, 35(4):114, 2016.

[DRB+18] Angela Dai, Daniel Ritchie, Martin Bokeloh, Scott Reed, Jürgen Sturm, and Matthias Nießner. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 2, 2018.

[EJ09] M. Erz and B. Jähne. Radiometric and spectrometric calibrations, and distance noise measurement of ToF cameras. In *Workshop on Dynamic 3-D Imaging*, pages 28–41. Springer, 2009.

[EKC18] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 40(3):611–625, March 2018.

[Erz11] Michael Erz. *Charakterisierung von Laufzeit-Kamera-Systemen für Lumineszenz-Lebensdauer-Messungen*. PhD thesis, IWR, Fakultät für Physik und Astronomie, Univ. Heidelberg, 2011.

[FB07] Dragos Falie and Vasile Buzuloiu. Noise characteristics of 3d time-of-flight cameras. In *Proceedings of International Symposium on Signals, Circuits and Systems (ISSCS)*, volume 1, pages 1–4, 2007.

[FBF77] Jerome H Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):209–226, 1977.

[FPR+09] Mario Frank, Matthias Plaue, Holger Rapp, Ullrich Koethe, Bernd Jähne, and Fred A. Hamprecht. Theoretical and experimental error analysis of continuous-wave time-of-flight range cameras. *Journal of Optical Engineering*, 48:1–16, 2009.

[FTF+15] Nicola Fioraio, Jonathan Taylor, Andrew Fitzgibbon, Luigi Di Stefano, and Shahram Izadi. Large-scale and drift-free surface reconstruction using online subvolume registration. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 4475–4483, June 2015.

[GI04] Jack Goldfeather and Victoria Interrante. A novel cubic-order algorithm for approximating principal direction vectors. *ACM Transactions on Graphics*, 23(1):45–63, 2004.

[GIRL03] Natasha Gelfand, Leslie Ikemoto, Szymon Rusinkiewicz, and Marc Levoy. Geometrically stable sampling for the ICP algorithm. In *International Conference on 3D Digital Imaging and Modeling (3DIM)*, pages 260–267, October 2003.

[GM13] Daniel Girardeau-Montaut. CloudCompare OpenSource Project. 2013. last access: Oct. 27, 2018.

[God12] John P. Godbaz. *Ameliorating systematic errors in full-field AMCW lidar*. PhD thesis, School of Engineering, University of Waikato, Hamilton, New Zealand, 2012.

[GPF10] David Gallup, Marc Pollefeys, and Jan-Michael Frahm. 3D reconstruction using an n-layer heightmap. In *Pattern Recognition*, pages 1–10. Springer, 2010.

[GPKT10] Varun Ganapathi, Christian Plagemann, Daphne Koller, and Sebastian Thrun. Real time motion capture using a single time-of-flight camera. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 755–762, 2010.

[GRB94] Guy Godin, Marc Rioux, and Rejean Baribeau. Three-dimensional registration using range and intensity information. volume 2350, pages 279–290, 1994.

[GYB04] S Burak Gokturk, Hakan Yalcin, and Cyrus Bamji. A Time-of-Flight depth sensor-system description, issues and solutions. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, pages 35–35. IEEE, 2004.

[HDD+92] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *ACM Transactions on Graphics*, 26(2):71–78, 1992.

[HHE11] Stephan Hussmann, Alexander Hermanski, and Torsten Edeler. Real-time motion artifact suppression in ToF camera systems. *IEEE Transactions on Instrumentation and Measurement*, 60(5):1682–1690, 2011.

[HKH+12] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *Internation Journal of Robotic Research*, 31:647–663, April 2012.

[HLCH12] Miles Hansard, Seungkyu Lee, Ouk Choi, and Radu P. Horaud. *Time of Flight Cameras: Principles, Methods, and Applications*. SpringerBriefs in Computer Science. Springer, November 2012.

[HLK13] Thomas Hoegg, Damien Lefloch, and Andreas Kolb. Real-time motion artifact compensation for pmd-tof images. In *Time-of-Flight and Depth Imaging*, pages 273–288. Springer, 2013.

[HZ03] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[IKH+11] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011.

[IZN+16] Matthias Innmann, Michael Zollhöfer, Matthias Nießner, Christian Theobalt, and Marc Stamminger. Volumedeform: Real-time volumetric non-rigid reconstruction. In *Proceedings of European Conference on Computer Vision*, pages 362–379. Springer, 2016.

[JH97] Andrew Edie Johnson and Martial Hebert. Surface registration by matching oriented points. In *Proceedings of IEEE International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, NRC '97, pages 121–128, Washington, DC, USA, 1997. IEEE Computer Society.

[JK99] Andrew Edie Johnson and Sing Bing Kang. Registration and integration of textured 3d data. *Image and Vision Computing*, 17(2):135–147, 1999.

[JU04] Simon J. Julier and Jeffrey K. Uhlmann. Unscented filtering and nonlinear estimation. In *Proceedings of the IEEE*, volume 92, pages 401–422, 2004.

[KAWB09] Klaas Klasing, Daniel Althoff, Dirk Wollherr, and Martin Buss. Comparison of surface normal estimation methods for range sensing applications. In *International Conference on Robotics and Automation (ICRA)*, pages 3206–3211. IEEE, 2009.

[KH13] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics*, 32(3):29:1–29:13, 2013.

[KK09] Maik Keller and Andreas Kolb. Real-time simulation of time-of-flight sensors. *Journal of Simulation Practice and Theory*, 17:967–978, 2009.

[KLL+13] Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *Proceedings of IEEE International Conference on 3D Vision (3DV)*, 3DV '13, pages 1–8. IEEE Computer Society, 2013.

[KM07] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 225–234. IEEE, 2007.

[KM12] Kurt Konolige and Patrick Mihelich. OpenKinect: ROS' technical description of Kinect calibration. `http://wiki.ros.org/kinect_calibration/technical`, 2012. last access: Oct. 27th, 2018.

[KRI06] Timo Kahlmann, Fabio Remondino, and Hilmar Ingensand. Calibration for increased accuracy of the range imaging camera swissrangertm. *Image Engineering and Vision Metrology (IEVM)*, 36(3):136–141, 2006.

[KSC13] Christian Kerl, Jurgen Sturm, and Daniel Cremers. Dense visual slam for rgb-d cameras. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2100–2106. IEEE, 2013.

[KSC15] Christian Kerl, Jorg Stuckler, and Daniel Cremers. Dense continuous-time tracking and mapping with rolling shutter rgb-d cameras. In *Proceedings of IEEE International Conference on Computer Vision*, pages 2264–2272, 2015.

[KSSC14] Christian Kerl, Mohamed Souiai, Jurgen Sturm, and Daniel Cremers. Towards illumination-invariant 3d reconstruction using ToF RGB-D cameras. In *Proceedings of IEEE International Conference on 3D Vision (3DV)*, pages 39–46, 2014.

[KV01] Aravind Kalaiah and Amitabh Varshney. Differential point rendering. In *Proceedings of the 12th Eurographics Workshop on Rendering*, pages 139–150, London, UK, June 2001.

[Lan00] Robert Lange. *3D Time-of-Flight distance measurement with custom solid-state image sensor in CMOS/CCD-Technology*. PhD thesis, University of Siegen, 2000.

[LC87] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM Transactions on Graphics (Proc. SIGGRAPH)*, volume 21, pages 163–169. ACM, 1987.

[LHK13] Damien Lefloch, Thomas Hoegg, and Andreas Kolb. Real-time motion artifacts compensation of tof sensors data on gpu. volume 8738, pages 1–7. SPIE, 2013.

[Lin10] Marvin Lindner. *Calibration and real-time processing of Time-of-Flight range data*. PhD thesis, CG, Fachbereich Elektrotechnik und Informatik, Univ. Siegen, 2010.

[LK06] Marvin Lindner and Andreas Kolb. Lateral and depth calibration of PMD-distance sensors. In *International Symposium on Visual Computing (ISVC)*, volume 4292 of *Lecture Notes in Computer Science*, pages 524–533. Springer, 2006.

[LK07] Marvin Lindner and Andreas Kolb. Calibration of the intensity-related distance error of the PMD ToF-camera. In *Intelligent Robots and Computer Vision*, volume 6764. SPIE, 2007.

[LK09] Marvin Lindner and Andreas Kolb. Compensation of motion artifacts for Time-of-Flight cameras. In Andreas Kolb and Reinhard Koch, editors, *Dynamic 3D Imaging*, volume 5742 of *Lecture Notes in Computer Science*, pages 16–27. Springer Berlin Heidelberg, 2009.

[LKS+13] Frank Lenzen, Kwang In Kim, Henrik Schäfer, Rahul Nair, Stephan Meister, Florian Becker, Christoph S Garbe, and Christian Theobalt. Denoising strategies for time-of-flight data. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, pages 25–45. Springer, 2013.

[LKS+17] Damien Lefloch, Markus Kluge, Hamed Sarbolandi, Tim Weyrich, and Andreas Kolb. Comprehensive use of curvature for robust and accurate online surface reconstruction. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 39(12):2349–2365, 2017.

[LLK08] Marvin Lindner, Martin Lambers, and Andreas Kolb. Sub-pixel data fusion and edge-enhanced distance refinement for 2D / 3D images. *International Journal of Intelligent Systems Technologies and Applications*, 5:344–354, 2008.

[LNL+13] Damien Lefloch, Rahul Nair, Frank Lenzen, Henrik Schäfer, Lee Streeter, Michael J Cree, Reinhard Koch, and Andreas Kolb. Technical foundation and calibration methods for time-of-flight cameras. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, pages 3–24. Springer, 2013.

[LS09] Tsz-Wai Rachel Lo and J. P. Siebert. Local feature extraction and matching on range images: 2.5D SIFT. *Computer Vision and Image Understanding*, 113(2009):1235–1250, 2009.

[LSHW07] O Lottner, A Sluiter, K Hartmann, and W Weihs. Movement artefacts in range images of Time-of-Flight cameras. In *International Symposium on Signals, Circuits and Systems (ISSCS)*, volume 1, pages 1–4. IEEE, 2007.

[LSKK12] Seungkyu Lee, Hyunjung Shim, James D. K. Kim, and Chang Yeong Kim. Tof depth image motion blur detection using 3d blur shape models. volume 8296, pages 1–6. SPIE, 2012.

[LWK15] Damien Lefloch, Tim Weyrich, and Andreas Kolb. Anisotropic point-based fusion. In *International Conference on Information Fusion (Fusion)*, pages 2121–2128. IEEE, 2015.

[MAB08] David K. MacKinnon, Victor C. Aitken, and Francois Blais. Review of measurement quality metrics for range imaging. *Journal of Electronic Imaging*, 17:1–14, 2008.

[MBE+08] B. Moser, F. Bauer, P. Elbau, B. Heise, and H. Schöner. Denoising techniques for raw 3d data of tof cameras based on clustering and wavelets. volume 6805, pages 1–12. SPIE, 2008.

[MHFdS+12] Lena Maier-Hein, Alfred Michael Franz, Thiago R dos Santos, Mirko Schmidt, Markus Fangerau, Hans-Peter Meinzer, and J Michael Fitzpatrick. Convergent iterative closest-point algorithm to accomodate anisotropic and inhomogenous localization error. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 34(8):1520–1532, Aug 2012.

[Mit07] Martin Mittring. Finding next gen: Cryengine 2. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07, pages 97–121, New York, NY, USA, 2007. ACM.

[MSR07] Evgeni Magid, Octavian Soldea, and Ehud Rivlin. A comparison of gaussian and mean curvature estimation methods on triangular meshes of range image data. *Computer Vision and Image Understanding*, 107(3):139–159, 2007.

[NA13] Sachin Nigam and Vandana Agrawal. A review: Curvature approximation on triangular meshes. *International Journal of Engineering Science and Innovative Technology (IJESIT)*, 2(3):330–339, May 2013.

[NDI+11] Richard A Newcombe, Andrew J Davison, Shahram Izadi, Pushmeet Kohli, Otmar Hilliges, Jamie Shotton, David Molyneaux, Steve Hodges, David Kim, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2011.

[NFS15] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, pages 343–352, 2015.

[NIL12] Chuong V Nguyen, Shahram Izadi, and David Lovell. Modeling Kinect sensor noise for improved 3D reconstruction and tracking. In *Proceedings of IEEE International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 524–530, 2012.

[NZIS13] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3D reconstruction at scale using voxel hashing. *ACM Transactions on Graphics*, 32(6):169, 2013.

[PBP08] Kaustubh Pathak, Andreas Birk, and Jann Poppinga. Sub-pixel depth accuracy with a Time of Flight sensor using multimodal Gaussian analysis. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3519–3524, 2008.

[PDCC10] Andrew D Payne, Adrian A Dorrington, Michael J Cree, and Dale A Carnegie. Improved measurement linearity and precision for AMCW Time-of-Flight range imaging cameras. *Applied Optics*, 49(23):4392–4403, 2010.

[PGK02] Mark Pauly, Markus Gross, and Leif P Kobbelt. Efficient simplification of point-sampled surfaces. In *Proceedings of IEEE International Conference on Visualization*, pages 163–170. IEEE, 2002.

[PH03] Emil Praun and Hugues Hoppe. Spherical parametrization and remeshing. In *ACM Transactions on Graphics*, volume 22, pages 340–349, 2003.

[Pul99] Kari Pulli. Multiview registration for large data sets. In *International Conference on 3D Digital Imaging and Modeling (3DIM)*, pages 160–168, oct 1999.

[Rap07] Holger Rapp. Experimental and theoretical investigation of correlating ToF-camera systems. Master's thesis, University of Heildeberg, 2007.

[RDP+11] Malcolm Reynolds, J Dobos, Leto Peel, Tim Weyrich, and Gabriel J Brostow. Capturing Time-of-Flight data with confidence. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 945–952. IEEE, 2011.

[RHHL02] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3D model acquisition. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 21(3):438–446, 2002.

[RL01] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *International Conference on 3D Digital Imaging and Modeling (3DIM)*, pages 145–152. IEEE, 2001.

[RV12] Henry Roth and Marsette Vona. Moving volume kinectfusion. In *Proceedings of British Machine Vision Conference*, volume 20, pages 1–11, 2012.

[SA07] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, volume 4, pages 109–116, 2007.

[SBK08] Ingo Schiller, Christian Beder, and Reinhard Koch. Calibration of a pmd-camera using a planar calibration pattern together with a multi-camera setup. *The international archives of the photogrammetry, remote sensing and spatial information sciences*, 21:297–302, 2008.

[SBSS08] Agnes Swadzba, Niklas Beuter, Joachim Schmidt, and Gerhard Sagerer. Tracking objects in 6D for reconstructing static scenes. In *Proceedings of the 2008 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, pages 1–7. IEEE, 2008.

[Sch11] Mirko Schmidt. *Analysis, modeling and dynamic optimization of 3D Time-of-Flight imaging systems*. PhD thesis, IWR, Fakultät für Physik und Astronomie, Univ. Heidelberg, 2011.

[Sei08] P. Seitz. Quantum-noise limited distance resolution of optical range imaging techniques. *IEEE Transactions on Circuits and Systems I*, 55(8):2368 –2377, sept. 2008.

[SFW08] Olivier Steiger, Judith Felder, and Stephan Weiss. Calibration of Time-of-Flight range imaging cameras. In *International Conference on Image Processing (ICIP)*, pages 1968–1971. IEEE, 2008.

[SG14] Jacopo Serafin and Giorgio Grisetti. Using augmented measurements to improve the convergence of icp. In *Simulation, Modeling, and Programming for Autonomous Robots*, pages 566–577. Springer, 2014.

[SG15] Jacopo Serafin and Giorgio Grisetti. Nicp: Dense normal based point cloud registration. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 742–749. IEEE, 2015.

[Sha56] Roland V. Shack. Characteristics of an image-forming system. *Journal of Research of the National Bureau of Standards*, 56(5):245–260, 1956.

[SJ09] Mirko Schmidt and Bernd Jähne. A physical model of Time-of-Flight 3D imaging systems, including suppression of ambient light. In *Dynamic 3D Imaging*, pages 1–15. Springer, 2009.

[SLK15] Hamed Sarbolandi, Damien Lefloch, and Andreas Kolb. Kinect range sensing: Structured-Light versus Time-of-Flight Kinect. *Computer Vision and Image Understanding*, 13:1–20, 2015.

[SMD+08] H Schöner, Bernhard Moser, Adrian A Dorrington, Andrew D Payne, Michael J Cree, Bettina Heise, and Frank Bauer. A clustering based denoising technique for range images of time of flight cameras. In *Proceedings of IEEE International Conference on Computational Intelligence for Modelling Control & Automation*, pages 999–1004. IEEE, 2008.

[SMGKD14] Renato F Salas-Moreno, Ben Glocken, Paul HJ Kelly, and Andrew J Davison. Dense planar SLAM. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 157–164, 2014.

[SPH08] Michael Sturmer, Jochen Penne, and Joachim Hornegger. Standardization of intensity-values acquired by Time-of-Flight cameras. In *Proceedings of the 2008 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, pages 1–6. IEEE, 2008.

[SPK18] Hamed Sarbolandi, Markus Plack, and Andreas Kolb. Pulse based time-of-flight range sensing. *Sensors*, 18(6):1679, 2018.

[SSP18] Thomas Schöps, Torsten Sattler, and Marc Pollefeys. Surfelmeshing: Online surfel-based mesh reconstruction. *arXiv preprint arXiv:1810.00729*, 2018.

[ST91] Bahaa E. A. Saleh and Malvin Carl Teich. *Fundamentals of Photonics*, chapter 10, pages 368–372. John Wiley and Sons, New York, New York, USA, 1991.

[STDT09] Sebastian Schuon, Christian Theobalt, James Davis, and Sebastian Thrun. Lidarboost: Depth superresolution for ToF 3D shape scanning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 343–350. IEEE, 2009.

[SYS07] Michal Sofka, Gehua Yang, and Charles V. Stewart. Simultaneous covariance driven correspondence (cdc) and transformation estimation in the expectation maximization framework. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[TM98] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Proceedings of IEEE International Conference on Computer Vision*, pages 839–846, 1998.

[Wei97] S. Weik. Registration of 3-d partial surface models using luminance and depth information. In *Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pages 93–100, May 1997.

[WJK+12] Thomas Whelan, Hordur Johannsson, Michael Kaess, John J Leonard, and John McDonald. Robust tracking for real-time dense rgb-d mapping with kintinuous. Technical report, 2012.

[WLSM+15] Thomas Whelan, Stefan Leutenegger, Renato F Salas-Moreno, Ben Glocker, and Andrew J Davison. ElasticFusion: Dense SLAM without a pose graph. In *Robotics: Science and Systems (RSS)*, Rome, Italy, July 2015.

[WMS16] Oliver Wasenmüller, Marcel Meyer, and Didier Stricker. Corbs: Comprehensive rgb-d benchmark for slam using kinect v2. In *Proceedings of the 2016 Winter Conference on Applications of Computer Vision (WACV)*, pages 1–7. IEEE, 2016.

[WWLVG09] Thibaut Weise, Thomas Wismer, Bastian Leibe, and Luc Van Gool. In-hand scanning with online loop closure. In *Proceedings of the 2009 International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1630–1637. IEEE, 2009.

[XOT13] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1632, 2013.

[Zha94] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994.

[Zha00] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, November 2000.

[ZK13] Qian-Yi Zhou and Vladlen Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 32(4):112:1–112:8, July 2013.

[ZK14] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. *ACM Transactions on Graphics*, 33(4):155, 2014.

[ZK15] Qian-Yi Zhou and Vladlen Koltun. Depth camera tracking with contour cues. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 632–638, 2015.

[ZLC08] X. Zhang, H. Li, and Z. Cheng. Curvature estimation of 3D point cloud surfaces through the fitting of normal section curvatures. In *Proc. AsiaGraph*, pages 72–79, 2008.

[ZMK13] Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun. Elastic fragments for dense scene reconstruction. In *Proceedings of IEEE International Conference on Computer Vision*, pages 473–480, 2013.

[ZNI+14] Michael Zollhöfer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, and Marc Stamminger. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Transactions on Graphics*, 33(4):156:1–156:12, July 2014.

[ZPB07] Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime TV-L 1 optical flow. In *Pattern Recognition*, pages 214–223. Springer, 2007.

[ZPBG02] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross. EWA splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, 2002.

[ZZZL13] Ming Zeng, Fukai Zhao, Jiaxiang Zheng, and Xinguo Liu. Octree-based fusion for realtime 3d reconstruction. *Graphical Models*, 75(3):126–136, 2013.