Dissertation
submitted to the
Combined Faculties for the
Natural Sciences and for Mathematics
of the Ruperto-Carola University of Heidelberg, Germany
for the degree of
Doctor of Natural Sciences

Put forward by

Diplom-Physicist: **Stephan Nicolas Robert Meister**
Born in: **Mannheim, Germany**

Oral examination: 08.01.2014

On Creating Reference Data for Performance Analysis in Image Processing

Referees: Prof. Dr. Bernd Jähne

Prof. Dr. Karl-Heinz Brenner

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Erzeugung von Referenzdatensätzen für Bildverarbeitungsalgorithmen, insbesondere für Methoden zur Lösung des dichten Korrespondenzproblems.
Es werden drei Arten von Referenzdaten unterschieden: Reale Bilddaten mit dichten Referenzdaten, Reale Bilddaten mit unvollständiger (sparse) Referenz sowie synthetische Bilddaten.
Für die Erzeung von dichten Referenzdaten wird ein existierendes, auf Tiefenkameras basierendes Verfahren evaluiert. Die untersuchte Methode ist besonders geeignet für die Erzeugung großer Datenmengen mit kontrollierbarer Genauigkeit.
Die Bedeuting von partiellen Referenzdaten wurde durch die Erzeugung zweier Datensätze für Fahrassistenzsysteme im Automobilbereich demonstriert. Mehrere Arbeiten zur Bildverarbeitung konnten dieser Datensätze für Tests sowie zur Verifikation nutzen.
Desweiteren wird untersucht inwiefern Methoden der Computer Graphik zur Erzeugung von synthetischen Referenzdaten genutzt werden können. Insbesondere die Erzeugung photorealistische Bildsequenzen mittels globaler Beleuchtungmodelle für die Evaluation von Algorithmen wird dabei untersucht. Die Ergebnisse zeigen dass synthetische Bildsequenzen zwar geeignet sind, deren Erzeugung aber noch praktische Probleme aufwirft Als Anwendungsbeispiel wurde hierzu ein neuartiger Simulator für Time-of-Flight Tiefenkameras entwickelt, der als erster das volle Spektrum an Fehlerquellen dieser Kameras simulieren kann.

Abstract

This thesis investigates methods for the creation of reference datasets for image processing, especially for the dense correspondence problem.
Three types of reference data can be identified: Real datasets with dense ground truth, real datasets with sparse or missing ground truth and synthetic datasets.
For the creation of real datasets with ground truth a existing method based on depth map fusion was evaluated. The described method is especially suited for creating large amounts of reference data with known accuracy.
The creation of reference datasets with missing ground truth was examined on the example of multiple datasets for the automotive industry. The data was used succesfully for verification and evaluation by multiple image processing projects.
Finally, it was investigated how methods from computer graphics can be used for creating synthetic reference datasets. Especially the creation of photorealistic image sequences using global illumination has been examined for the task of evaluating algorithms. The results show that while such sequences can be used for evaluation, their creation is hindered by practicallity problems. As an application example, a new simulation method for Time-of-Flight depth cameras which can simulate all relevant error sources of these systems was developed.

Acknowledgements

I would like to thank all the people that helped me create this thesis.

First, I'd like to express my gratitude to Prof. Dr. Bernd Jähne for letting me work on this topic and for allowing me to pursue my research interests unrestrictedly. He was a source of constant inspirations and invaluable professional guidance throughout my project. I would also like to thank Prof. Dr. Karl-Heinz Brenner for acting as my second referee.

Very special thanks go to Dr. Daniel Kondermann for his guidance during my many years at the HCI, both during my diploma as well as my doctoral thesis. I am particularly grateful for his patience and encouraging words at times I felt that my work wasn't up to standard. I'm looking forward to many more years of joint work and friendship.

I also thank my many colleagues at the HCI including Rahul Nair, Jens-Malte Gottfried, Moritz Becker, Henrik Schäfer, Gerald Mwangi and Burkhardt Güssefeld for both their scientific support the good working atmosphere. It really made me feel at home. Regardless of the topic, there was always someone with expertise I could draw from. Additionally, I thank all of those who helped contribute to this thesis and offered to proofread all of my work. I am also indebted to all members of the Intel Visual Computing Institute (IVCI) group, including Dr. Christoph Garbe and Dr. Frank Lenzen for helpful advice and for bringing structure into this work.

Furthermore, I appreciate the support of Anja Schäfer, Julia Freudenreich and Dr. Susanne Krömker of the IWR who provided me with data which I was able to draw from far more than ever anticipated. Likewise, I thank Bernhard Höfle, Markus Forbriger and all the other members of the Institute for Geography for their support regarding terrestrial LiDAR and 3D scanning.

Also many thanks to everyone at the CR/AEM-CVS and CR/AEM5 working groups of Robert-Bosch Gmbh for helping me create enough data to keep many many interns occupied for months.

I'm also grateful for my job and the friendly people at the Institute for Anatomy and Cell Biology which was like a third home during my years at the University.

Special thanks to my all my friends including Christoph, Alici, Julian, Karin, Jens and Lena for making the years in Heidelberg the best of my life, to my DnD group for being almost as nerdy as I am and to the *Pfälzer Supp* for staying close to me . They helped me not to take myself too seriously and were always eager to show me what is important in life.

Finally, I'd like to thank my parents, brothers and my nephew for their emotional support and encouragement. This thesis would not have been possible without them.

Contents

1 Reference Data and Datasets

1.1 Introduction

Image processing as a field of research as well as its application to concrete problems touches many different aspects of measurement sciences. Seldom raw image data as generated by a camera is exactly what we want, rather some information derived from this data is sought to perform tasks or make some statements about reality. An image processing pipeline may involve many steps, starting from the acquisition using a physical sensor (the camera), over some low-level image processing like denoising, debluring or motion estimation to high-level algorithms like pedestrian detection or air/water gas-exchange computation. Each of these steps involves algorithms, regardless of whether they are directly implemented in hardware on the sensor or computed long after the image data has been captured.

After multiple decades of research and development alone in image processing, many of the above steps have been approached from many different angles, giving birth to sometimes thousands of different algorithms or algorithms variations for a specific task. Selecting a suited algorithm for one subtasks can already be quite challenging, for a whole pipeline, it is daunting, especially for newcomers in the field. The construction of such a complex system can roughly be split into four phases: Planing, Assembly, Testing, Deployment. Many products follow an iterative cycle where results from the previous version help to improve the next one. Knowledge about the problem at hand as well as the algorithms to be used to solve them is crucial to each of these phases.

Unfortunately the third point, testing, or more often benchmarking, has been somewhat of a stepchild for image processing. It has long been considered as necessary, but not very popular diligence work. Further testing an already published algorithm generally yields less credit. This is even more unfortunate as good developed tests can also be beneficial in the planing stage of a project, take for example test driven development in software engineering. Fortunately, this attitude is changing and testing and benchmarking is drawing more attention in research and development.

On particular successful and often employed method of testing is the comparison of computation results with a value that is known to be true. This *Ground Truth* or *Reference Data* was either obtained by an alternate system or the test case was constructed in a way that the result is known beforehand. By comparing expected and actual results in a quantitative manner, it is often possible to identify distinct problems and errors in systems. Additionally, the method is applicable to both *white box testing*, where the inner workings of a system are known by the developer or practitioner, as well as *black box testing*, where only input and output of a system are accessible.

A common practice in image processing is to create multiple image sequences with ground truth for a certain field of application. This can be either a low level imaging task such as motion estimation, object detection or segmentation or a high-level task such as vehicle control, robot navigation or human-computer-interfaces (HCI). Together with additional metadata these sequences form benchmark datasets, which can be used for algorithm testing and performance evaluation using different quantitative performance criteria.

In this work I will discuss the fundamentals of benchmark datasets and ground truth in image processing, give information if and how it can be created and present several specific use cases for synthetic and real-world ground truth creation. As related work is often domain specific to an certain image processing task and not to ground truth generation in particular, it will be mentioned in the corresponding chapter introductions.

In Chapter 1 the principle considerations of dataset creation, as well as definitions and limitations will be covered. Chapter 2 does cover measurement basics for geometry based ground-truth datasets. The focus lies on 3D reconstruction or dense correspondence problems, such as optical flow. Chapter 3 deals with the importance of reference datasets without dense ground truth. To emphasize the possibilities of this type of data, two datasets targeted at the automotive sector are presented. The theory of synthetic reference data creation using modern computer graphics will be the topic of Chapter 4. Limitations and design considerations for synthetic ground truth will be demonstrated on multiple test cases. Chapter 5 will present a detailed example for synthetic ground truth creation by describing a simulation method for phase-modulated Time-of-Flight cameras based on global illumination methods. Chapter 6 finally will conclude the results and give an overview on alternative or future ground truth creation techniques.

Figure 1.1: Assumed satellite calibration pattern in Gansu, China. Source: Google Maps, Copyright 2013 Cnes/Spot Image, Digital Globe, 2013, AutoNavi

1.2 Ground Truth or Reference Data?

The definition of *ground truth* is not particularly easy. The origin of the term lies in remote sensing, more specifically in the examination of aerial images. Images taken from airplanes or satellites are used for many different tasks like geological surveys, plant growth estimation or weather forecast. To verify the content of a pixel in a aerial image, someone has to visit the depicted site on the *ground* to examine the *truth*. K. Kraus [95] for example summarized the efforts of multiple projects during which the positions of ten-thousand of geographical points were measured and compared to aerial images.

From here the term has found its place in many other disciplines of image processing or image analysis. The practical definition would be: '*What is the true (physical) value of some property at the location defined by a image pixel.*' One alternative definition could be: '*Ground truth is the value an imaging system or algorithm should report under optimal conditions, devoid of any systematical or statistical errors.*'

The sought for value could be a directly measurable physical value such as intensity, distance or in case of thermographic imaging the temperature. Alternatively, we may be interested in a value that is the result of an algorithm or semantic interpretation such as the disparity, a feature histogram or an object labeling. Here the problem starts, as often the property we are interested in may not be well defined in all cases. Typically, a image sensor can only measure the amount

of incident light coming from a certain direction. This light may be correlated to the sought value in a complex non-linear manner.

Take for example depth imaging systems, which will be featured prominently in this work. The job of these imagers is to measure the distance between itself and the object they are pointed at. Some, like laser rangefinders, do so for only one point, others like stereo camera system or Time-of-Flight cameras also have a lateral resolution and do so for many pixels. Regardless of whether these systems are passive or active (e.g. using an active lightsource like lasers) they can not perform a measurement at a single point. The area under the beam or projected onto the pixel will already show different distances unless the observed object is a perfect spherical cap. So in practice the system will only measure the mean distance, although as we will see later in Chapter 5 the returned depth can take other values depending on the system. Other problems could arise if the object is partially transparent. What is the correct value the system should report? The distance to the foreground object,the one in the background or just simply an error indicator?

Furthermore the term is also problematic as it suggest that the desired quantity can be measured with arbitrary precision. Physically there are always limits to that notion, based on the used equipment, resolution and noise. Only for synthetic datasets which will be discussed later this may be justified, as here we are in theory only limited by the available memory and algorithms.

Because of this problems the term ground truth is not always justified as it implies that the sought value always exist, regardless of the properties of the measurement system. A more appropriate name which will also be used trough this book is *reference data*. Reference data is a more general term as it includes any data that can be used for testing and evaluation without the implication of being perfect in any regard. It may also include data which does not directly allow quantitative evaluation.

1.2.1 Types of Reference Datasets

In [92] Kondermann defined three types of reference datasets:

Reference data with ground truth Images and sequences which includes the sought values with a higher accuracy than the system which should be evaluated typically can provide. Higher means here at least one magnitude more exact. Additionally the ground truth should be dense, meaning be available for every pixel where applicable. This is the most useful but logically also the most complicated to produce reference data. Many well

known datasets such as the Middlebury database for stereo or optical flow [133] [12] or the Pascal Visual Object Classes (VOC) Challenge[44] fall into this category.

Reference data with partial or weak ground truth Here the included results are either less exact than for real ground truth or only partially available. Examples could be stereo image sequences with depth maps which are accurate but have a lower lateral resolution such as the KITTI vision benchmark suite [53]. If confidence values for the ground truth can be provided, this data is quite useful to detect cases where algorithms could fail. See section 1.4 for more details.

Reference data without ground truth Here the expected results of the vision system are unknown. However, engineers and practitioners with a profound understanding of a problem can still use this data for approximate or manual inspection. Furthermore the results low-level vision tasks are often only an intermediate step for higher-level tasks, for example robot navigation via simultaneous localization and mapping (SLAM). A dataset depicting a complex navigation task must not necessarily contain feature tracker data etc. to be used for such an application. The advantage of this data is that it can often be created automatically in huge amounts. System which have to deal with a huge variety of different situation will naturally profit from such datasets. Examples can again be found in the automotive sector or in object recognition system. Unannotated databases such as the 101 Object categories database [46] or car scene sequences from [5] described in Chapter 3 fall into this category.

1.2.2 Creation and Measurement

Remark on Terminology. This thesis deals mostly with topics related to measurement sciences. The terms accuracy and precision are well defined for measurement processes [150] with the first describing the difference to the true value and the second describing the statistical variation given multiple realizations. For brevities sake I will mostly use the term accuracy in the general discussion and only give distinct values where they are relevant and known. Most of the measurement methods described in later chapters are more precise than they are accurate, so only the larger error source will be given.

The creation of reference data with ground truth is an unsolved, some would even say practically unsolvable problem. An ideal benchmark dataset would contain ground truth of the best achievable accuracy, consist of enough distinct image

sequences to cover the whole problem domain and should be easy and inexpensive to produce. If we had a system that could produce such datasets, it could actually replace the imaging system that we want to evaluate in many cases. For real life dataset creation there are always tradeoffs involved. Which priorities a certain dataset should focus on is often application dependent.

Take for example pedestrian detection. A single sequence of a man crossing a street is not very useful, even if taken at 200 frames per second and with very high resolution and supplemented with a full 3D scan. Given only this one sequence, we could only confirm that the algorithm which we want to test is able to detect this single person. The scene content is simply to specific and a learning algorithm would be prone to overfitting. Real evaluation can only be done with large amounts of data, taking scenes of many different people, walking or standing around, carrying luggage, etc. Generally speaking a benchmark dataset must contain enough images to cover the whole problem domain, including negative cases (here, images where there are no persons at all). This is related to the problem of content selection and requirements engineering which will be discussed in Section 1.3. Similarly, in many pedestrian detection scenarios, such as in video surveillance, the used cameras have rather low temporal and spatial resolution. Creating test data with high resolution is therefore not necessarily useful.

This means that we have to consider the question of how accurate data must be so that it can be used for evaluation purposes. Typically, reference data should be at least as accurate as the expected accuracy of the system to be evaluated, otherwise we wouldn't be able to give quantitative error estimates. Data for which this is not the case can however still be useful, see Section 1.4 on weak ground truth.

For nearly all measurement or calculation tasks we have a cost/accuracy progression. In the case of a camera the resolution can be changed in multiple dimensions: spatial (lateral or even volumetric), temporal, spectral and radiometric. Depending on these values the sensor or cameras weight, bulk, cost or handling complexity may change. Increasing any value will come at a cost. Higher lateral resolution (more pixels) will either increase the sensor size or reduce the pixel size, thereby increasing noise and reducing the dynamic range or radiometric resolution. A higher temporal resolution (higher framerates) will also decrease the amount of light per frame and increase the amount of data that must be handled, thereby increasing the total system cost.

A cheap or fast system may be evaluated with the next more expensive system until the state-of-the-art of existing systems is reached. A rather inaccurate depth camera can be tested and evaluated by using depth maps from a more accurate system such as multi-camera stereo or laser scanners, while those can again be

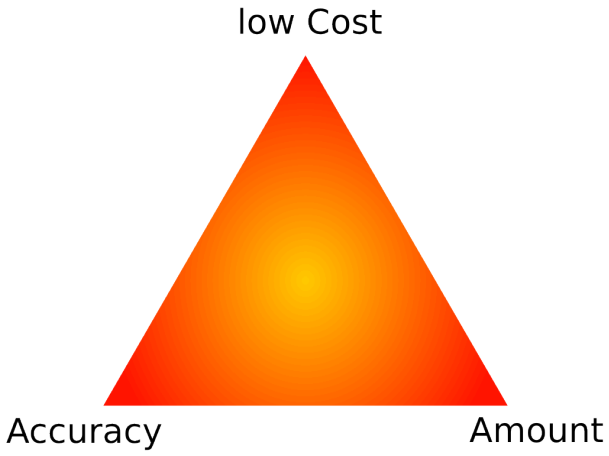


Figure 1.2: When creating reference data, there is always a tradeoff

evaluated by structured light scanners, then physical contact scanners etc. For results obtainable purely by computation we could generate reference data for a fast (real-time) algorithm by resorting to a slower, more reliable algorithm.

Often this does impose some constraint in the time domain as more accurate measurement typically take longer. This tactic for creating reference data may not be feasible in these cases. Take for example a driver assistance system that relies on image data. Reliable data must be supplied to the system in a fixed time-frame, otherwise it wouldn't be able to react. Creating reference data for a traffic situation can therefore not rely on measurement methods that are intrinsically slower than the cameras used in the actual system. In lab conditions it is sometimes possible to slow down processes to measure them but in most cases this will change process results in some maybe unpredictable manner. Analogous, it is not feasible to slow down or even stop traffic to take reference measurement. For this case a measurement method which is at least as fast but also more accurate must be devised.

A possible alternative to this dilemma could be synthetic reference data. Computer graphics has advanced significantly and is nowadays capable of creating photorealistic images which are very hard to distinguish from real camera footage. Such renderings are often based on a specific scene description which uses 3D polygon data, material properties, textures and animation rigs which can be modified freely. Ground truth data for various tasks can be derived directly from this description. Synthetic reference data generation will be discussed in detail in Chapter 4.

Resources in money, time and available workforce are always limited, especially in academia. The amount of data that can be created for a test set is additionally limited, given the fact that accurate data is more expensive and time consuming to create. Often measured data must be postprocessed, annotated and archived. Tasks which can not always be automated. A dataset with high accuracy may therefore include less variety than one with low resolution. A certain experience in which gaps can be left unclosed is therefore also part of reference data creation. The following section will show how this can be done by formalizing requirements for applications and datasets.

Additionally, a too small size of the dataset or content fragmentation can lead to long term problems in algorithm development. Some of the problems are discussed in Section 1.6.

1.3 Requirements Engineering and Content Selection

The image processing community largely agrees that reference data is useful and necessary [47]. However the opinions on what a reference dataset should include differ.

On the one side there is algorithm development. The first step in evaluation is to verify that a new algorithm does work in principle. This can usually be shown with a small and limited test set. Existing datasets for different low-level vision tasks fall partially in this category. Such reference data can be useful to devise an initial placement of an algorithm along the row of alternative solutions.

Following this, the aim of many researchers and developers is generalization: Making sure that their method or algorithm performs at least reasonably good on many problems, or to develop the one-size-fits-all algorithm. Often a single or narrow set of performance measures is used to decide on whether an algorithm is considered good.

On the other side, there are practitioners and engineers with one specific problem who need an algorithm that performs very well under very tight constrains. They also have an easier time naming and formalizing concrete scenarios for testing. If the task is to detect dogs in images, one does not care if the algorithm works well on cats, however it should probably never mistake the one for the other. Additionally, their performance measures may be quite different with graceful degradation or predictable behavior on faulty input data being potentially more important than pure accuracy on perfect input data.

Seldom, the constraints of both groups can be met. One solution is to formalize requirements and performance indicators for both algorithms as well as applications. A developer can formalize under which general circumstances an algorithm performs well while an practitioner can define requirements for a specific application scenario.

Typical requirements for a depth estimation algorithm could be: speed of execution, accuracy, depth range, depth edge behavior, noise, etc. Each of these can be tested individually given that quantifiable error metrics based on comparison with reference data exist. An application, for example gesture recognition, could have the following requirements: high framerate, low accuracy and depth range, stable and sharp depth corners, noise largely irrelevant. The appropriate algorithm can then be selected based on these constraints and further tests can be devised.

1.4 Weak Ground Truth

As already mentioned, weak ground truth datasets are such ones, where reference data is only sparse or has limited accuracy. However, these sets can also be useful for evaluation purposes.

One scenario is related to *graceful degradation*. This describes how a certain algorithm reacts when the quality of the input data decreases. In many systems the error of the output data is correlated to the error of the input, yet not necessarily in a linear manner. Unless the system is able to make predictions about its own errors, a user may be interested in this correlation. Datasets with reduced accuracy or even a controlled accuracy progression can help to decide on the reliability of an algorithms output. Advanced algorithms may even be able to detect when they will fail on a certain set of input data. The number of images where a specific algorithm will not work is also tremendously larger than the number where it will. This means that especially for negative testing potentially large amounts of data are necessary.

Related to this are scenarios where weak ground truth can be used for binary decisions. Optical flow (OF) for example, which is defined as the apparent per-pixel motion between two frames of an image sequence, is dependent on certain assumptions about the illumination in a scene. Most often this is realized using a brightness constancy constraint (BCC) which states that the total brightness between two consecutive frames stays constant. But of course there are image sequences where this constraint is violated. Moving shadows or specular reflexes on mirror like surfaces are good examples. A OF benchmark dataset which does not contain dense ground truth flow fields but instead marks regions where the

BCC is violated could be considered weak ground truth and is still very useful to evaluate the behavior of OF algorithms. Other examples for such underdetermined or practically unsolvable problems are untextured regions for stereo algorithms or edge detection in heavily blurred images.

1.5 Best Practices of Data Representation

A benchmark dataset used for evaluation and performance estimation is not limited to the raw data. If the intention is to create reusable datasets, probably for a broad research community, additional considerations have to be taken care of.

Scientific practice must adhere to certain standards. These include, among others, testability and reproducibility. Formally speaking, the creation of reference data is a scientific experiment. As such it must also be subject to these standards, including the constraint that it should be possible to repeat the process of creation under the same circumstances.

This includes the inclusion of metadata, description of the software and hardware as well as additional documentation regarding the data. Additional data may also make applications and test that weren't part of the original intent possible. Foerstner [47] for example suggested that reference data creation should be a joint effort. With a broad experimental design the same datasets can be reused for different tasks. The Middlebury stereo datasets [133] for example, which were initially developed only to evaluate stereo algorithms have recently been used in light field research (See Section 4.4).

Metadata

Reference data in image processing is not limited to the raw pixel values. For many tasks the metadata is equally important. This may include information such as: When and where was the image taken? With which lens, aperture and focal length? At which temperature and illumination conditions? Was postprocessing applied? If yes, what kind and with which parameters? All this information is part of the experiment and must therefore be made part of the dataset. Correct requirements engineering as described in Section 1.3 can help decide which information is crucial and which is optional. Camera calibration data for example is vital for nearly all applications that deal with geometric scene properties such as sizes, movement, structure etc. Also included in this information is scene description and documentation.

Software and File Formats

The principles of open source have been successfully applied in academics for many years. The publication of reference implementations and source code alongside a more generic descriptions of algorithms is therefore often practised. One can observe that the most popular algorithms for a certain task are not the ones with the best performance, but the ones for which an easily usable implementation is available. Of course this is not always the case as releasing code puts the additional burden of maintaining and documenting it on the researcher. Issues of legal concern, patents and intellectual property can also be limiting factors for this practice.

Unfortunately, software which was used in the creation of reference data is seldom subject to publication. This has many causes. The software may be commercial, the product of rapid prototyping or just considered unimportant. Often it seems that there is little reason to release a camera capture software hastily mashed together from well known libraries and tools. As there is no scientific novelty in it, publication usually yields no or little scientific credit.

However, the software is part of the experiment and should, if possible, be part of the dataset as well. This is especially important considering the long time availability of the data. If data is for example stored in a proprietary format, the original software may be the only method to access it.

How and in which format data should be distributed is also not an easy decision. This starts already at the lowest level of data representation.

For single images there exist many file formats such as PNG, JPEG, PGM, OpenEXR etc. They differ in compression methods, supported metadata formats, supported datatypes etc. A 8-bit RGB image can be read and processed by a variety of software. Here the image processing and computer graphics research communities have the advantage that tools from artistic and or industrial fields of application can often be used for research purposes. The gain in productivity if one can use the same software to process vacation photos and conduct research can not be denied. But once one leaves the domain of basic user formats and wants to process e.g. 16-bit color images the number of supported formats is reduced drastically and it is not guaranteed that the image processing software will not change the data in some unpredictable manner.

For multiple images, image sequences or multidimensional data the range of available generic container formats is generally smaller. Data formats may be proprietary, limited to a certain software or insufficiently documented.

Recently some container formats which address this problem have been developed and the availability of libraries and software that can handle these formats is increasing steadily. An example is the Hierarchical Data Format (HDF5) [61] used in many different scientific fields. It can be used to save large amount of high dimensional data in a structured way along with comments, metadata, etc. Its open structure and excellent support in many programming languages and software make it an ideal candidate for distributing reference data.

Maintenance and Archiving

Finally, one should make sure that the published data is properly maintained and is kept available as long as it is needed. Publication of data on a dedicated website or in a benchmark database is the current standard. Websites however need to be maintained, even if they contain mostly static data. New technologies for data distribution and information presentation such as dynamic websites increase the maintenance requirements but can also help to make the data accessible to a broader audience.

This is long-time responsibility that has to be organized or delegated as seldom the people maintaining the hardware etc. are the same ones that performed the experiments. Also software is subject to changes and updates over time and it is not guaranteed that data in a certain format can still be read years later. Additionally, potentially large amounts of scientific relevant data remain unused as the researchers who created them are now working on other projects, etc. For a fast developing field such as image processing the time after which a certain dataset becomes mostly irrelevant is also not easy to predict. Some optical flow test sequences for example are still used even decades after they have been developed while other are forgotten merely months after publication. Solving these problems however is a more general and related to research management or even archival sciences.

1.6 Limits and Problems

It is the general consensus, that the usage of benchmark datasets and reference data for algorithm evaluation had a largely beneficial impact on computer vision. However, there are also problems related with that practice.

One major problem is *overfitting*, a term that originally comes from the field of machine learning. Overfitting occurs when a model becomes too complex, so that it performs very well on training data but looses it predictive capabilities.

This can for example be the case when the number of model parameters comes close to the number of observations in the training data. The result is that the learning algorithm adjusts its parameters to the layout of the training data and can no longer be used for generic problems of the same kind but with different realizations.

The same problem can occur on a meta level with the algorithms taking the place of the parameters and the developer/researcher acting as the learning algorithm. A reference dataset which is accurate, easy to use and sufficiently broad will of course be used more often and may establish itself as the one benchmark dataset used throughout the specific field. The result is that researchers may (probably involuntarily or unnoticed) tailor the algorithms they develop to that specific benchmark. A new method may only be considered worthwhile (or publishable) if it performs well (or even better than any of its competitors) on this particular benchmark dataset. But one dataset can hardly cover all possible applications (of which many may not even be known yet as science and applications marches on). The only practical solution is to treat benchmarking and reference data creation in the same manner as algorithm development. In the same manner as a single image processing problem can never be solved completely, dataset creation should never be considered finished and should keep track with development.

1.7 Further Reading

Reference data creation is connected closely to the topic of performance analysis. This field deals among others with the comparability and characterization of algorithms. Reference data is one popular method to compare the performance of algorithms regarding different error metrics. Furthermore it can be used to classify algorithms into different groups.

The current state of performance analysis for optical flow evaluation was summarized by Kondermann et al. [93]. They argue that algorithms should not be characterized or even ranked based on a single scalar value and that evaluation should be performed on as much (ground truth-) data as possible.

This is also true for other algorithms were currently benchmarks are performed on very small datasets [63].

Arguments and suggestions for experimental design in computer vision have been brought forth for example by Foerstner [47], Maimone et al. [108] or Courtney and Thacker[35]. On often brought up argument is that proper error statistics are

needed, emphasizing that testing on single images is insufficient for most vision tasks.

Listing all existing benchmark datasets is definitely not possible but publications relevant to specific tasks will be listed in the corresponding Chapters. An overview can for example be obtained at the CVonline website of the University of Edinburgh which lists nearly 250 entries[1] for different types of datasets and benchmark databases Other lists are available for example at the Pilot European Image Processing Archive of the University of Essex[2]. Additionally Thacker et al. [151] list examples for different fields in their overview on best practices for performance characterization.

[1] http://homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.htm
[2] http://peipa.essex.ac.uk/index.html

2 Real World Reference Data with Ground Truth

This chapter will focus on methods for creating real-world datasets with accurate reference data. Special emphasis will be put on low-level vision tasks which are based on the geometric structure of the scenes. This includes depth and movement estimation such as optical flow, stereo or feature tracking. Solving such problems is often an intermediate step for concrete applications, such as 3D scene reconstruction or robot navigation.

Many of these low-level problems are variants of the correspondence problem, determining which pixel in one view corresponds to a pixel in another view. Reference data for such low-level tasks is useful in more than one way as it is relatively independent of the targeted application. It is therefore advantageous for the general task of reference data creation to investigate how geometric scene information can be obtained.

The following sections will summarize the basics of projective geometry and camera models as this will lay the foundation also for the later chapters dealing with synthetic ground truth and image synthesis. Then, we will discuss design choices for acquiring geometric reference data, followed by an detailed examination of one method which is particularly suited for the fast acquisition of large datasets.

2.1 Related Work

The following works and datasets represent the current state of the art for real-word datasets related to geometric reference data and correspondence problems. The mentioned works represent examples for both high accuracy data acquired under lab conditions as well as scenes for real-world and outdoor applications. As such they represent different types of constraints for correspondence problems.

The probably most well known database is the Middlebury Optical Flow database [12]. Herein Baker et al. used fluorescent paint to coat the depicted objects with a easily matchable pattern to determine the optical flow between frames.

Figure 2.1: Schefflera, Army and Meguon example scenes from the OF Middlebury datasets [12]

By additional temporal and spatial downsampling the images they were able to produce ground truth maps with flow amplitudes in the subpixel range. The stated accuracy for the evaluation images is 1 / 60th of a pixel, save for outliers. Much effort was put into each sequence and because of the complex process only four of them were created. To the authors best knowledge, there currently exists no real-world dataset with higher accuracy. Three example images from the dataset can be seen in Figure 2.1

In stereo vision, the correspondence displacement vectors are generally at least a magnitude larger but only estimated to about one pixel accuracy. Existing benchmark datasets reflect these constraints and many of them actually use handlabelled data. The Middlebury Stereo database [133] for example was created by using handlabelled piecewise planar objects. Disparity in the middle of the plane segments could be inferred by affine interpolation from the object edges and bounds. Also often used is the Tsukuba "'Head and Lamp"' dataset [115] with manually created disparity and occlusion information.

Recent developments seem to forfeit accuracy in favor of more realistic non-laboratory scenes. Often they do so by combining different scanning modalities to measure reference data. The KITTI Vision Benchmark Suite by Geiger et al.[53, 54, 50] for example combines stereo images with sparse laser scanner data to create partial depth reference data for automotive scenes. Their stated accuracy is about two to three pixels and the correspondence maps don't reach full density. In a similar manner Strecha et al. [28] use terrestrial LiDAR to create reference data for static scenes. The .enpeda project[1] contains multiple stereo datasets, some of them with segmentation or depth reference data, including for example the works by Morales et al. [114], which combine stereo and laser range finder data.

Simultaneous localization and mapping applications (SLAM) such as robot navigation also can use correspondence data with yet another set of constraints. Often

[1] `http://www.mi.auckland.ac.nz/index.php?view=article&id=43`, Accessed 09.10.2013

they need only sparse correspondence data or feature tracks, however preferably with consistent vectors over very long sequences (potentially thousand of images). The design rationale is, that for example robots must be able to detect if they have visited a location already, even if the camera angle has changed significantly. Reference data for these applications include for example the work presented by Sturm et al.[147] which uses multiple Microsoft Kinect depth sensors. Furthermore, the Rawseeds project[2] [22, 31] aims at creating a whole database of various robot navigation benchmark datasets.

Some of the authors improve their dataset regularly or replace them with newer ones. Interestingly there is a definite trends towards synthetic image sequences for benchmarking, as for example the Tsukuba sequences was succeeded by new rendered data [119, 109]. This and other datasets limited to synthetic or mostly synthetic images will be mentioned in more detail in Chapter 4.

[2] `http://www.rawseeds.org`

2.2 Geometric Image Processing and the Correspondence Problem

Creating reference data for correspondence problems is one of the best understood, but also one of the hardest problems in performance analysis. We will therefore first investigate how it is related to scene geometry and the image formation process.

Imaging always begins with a scene. This could be the real world, a geometric scene description in computer memory or some abstract phasespace representation of high-dimensional data. Mathematically it is a set of properties $\vec{v}(\vec{x})$ with x defined on a geometric space $\mathbb{R}^n$ (n typically equals 3 or 4 if the time-development is of importance or even 5 in the case of the static plenoptic function mentioned in Chapter 4.4). Images are projections of a part of these properties into a lower dimensional space, usually performed using a camera. Unless noted otherwise we use the basic definition of an image as a discrete two-dimensional array of values.

Of course this a very abstract description of the process. In any practical realization there are multiple physical effects involved, including radiometry, ray and wave-optics and sensor electronics. We should always keep this in mind when working with real data but for describing some basic principles the abstract model is still useful.

Image processing is a tool to infer information about the original scene from two-dimensional images. In quite a few application fields the desired information is related to the geometric structure of the scene (including the camera position) or its change over time. The data that can be derived from a single image is limited as depth or other geometry information can not generally be inferred unambiguously from one projection. Of course, ambiguities can always be reduced by making certain assumptions about the image and the scene.

Once multiple images of the same scene are available the number of assumptions that have to be made to arrive at the same conclusions about the scene are reduced drastically. Many low-level image processing problems that are related to geometric properties are variants of the *correspondence problem*. The question here is: For a given point in an image, what is the corresponding point in another image. If this property is known, the position of the original point in space can be calculated by means of triangulation. Algorithms like stereo methods try to find spatial correspondences while while others may try to find temporal correspondences to estimate movement.

For correspondence problems the most useful ground truth would be a exact and dense correspondence map as it allows the direct evaluation of the used algorithm. However, such a reference map is neither accessible by direct physical measurement, nor must it actually exists in all cases. The problem is that correspondences are defined in the image domain, not in the scene domain. Parts of the observed geometry may be visible in one camera view, but could be occluded in another one. Hence the corresponding point does not exist in the image even though it does so in real space. Depth, as the information sought for, may be defined for every pixel in the first image, but disparity (the distance between corresponding points in the two images) as an intermediate value is not.

This is a problem, as it is not clear which kind of possible ground truth is suited best. A disparity map is not optimal as it is not directly measurable by external means and doesn't represent an intrinsic property of an image (A disparity map for a single image is meaningless). A depth map is always defined but doesn't actually represent the output of a stereo algorithm.

Under these assumptions ground truth should be as close as possible to the original geometric representation of the scene as other values can be derived from it. This does of of course impose a tradeoff, as one needs to decide how and how accurate a scene needs to be measured to derive accurate and dense ground truth.

In the following sections I will present methods to derive geometric scene descriptions which can be used for the creation of low-level ground truth data. The basic principle is the usage of high accuracy measurement of different modalities to create reference data for low-accuracy methods.

2.2.1 Camera Calibration

Nearly all applications in this and the following chapters deal with real or virtual cameras in one way or another. The camera represent the connection between a 3-dimensional scene and the 2-dimensional image on a sensor plane. If and how the projection a camera performs can be measured and probably inverted depends on the camera model and available information.

The simplest camera model which performs the mentioned mapping is the pinhole camera which can be described by its principal point C and the image plane located behind it at a distance f (the *focal length*). As a mathematical concept the pinhole camera does not deal with physical effects like exposure time, motion blur, diffraction or distortions. In so far it can be considered as the ideal, but not realizable camera.

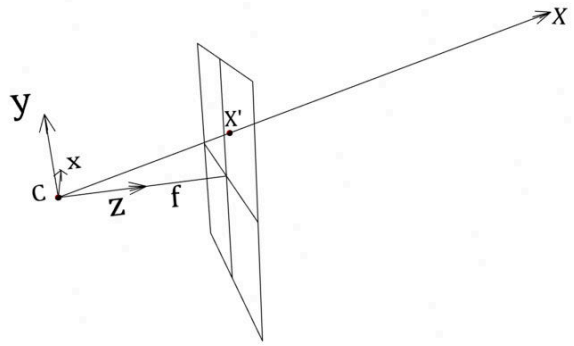


Figure 2.2: Pinhole camera model. The virtual sensor plane is depicted in front of the principal point, not behind it as it would be the case for real cameras.

None the less, the characteristics of a regular camera with lenses, sensor pixels and attached electronics can be modeled as extensions of the pinhole model and so it is useful as a basis for further examinations. This is especially true for virtual cameras which are used as abstractions in computer graphics. Here the base image is often synthesized as if it was taken by a perfect pinhole camera and physical effects like depth of field or motion blur are added as a post processing step if necessary.

Using homogeneous coordinates, a point $P = (X, Y, Z, W)$ in $\mathbb{R}^3$ is projected onto the point $p = (x, y, w)$ as follows:

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = \underbrace{\begin{pmatrix} f_x & s & c_x \\ 0 & f_x & c_y \\ 0 & 0 & 1 \end{pmatrix}}_{C} \underbrace{\begin{pmatrix} r_{00} & r_{01} & r_{02} & t_x \\ r_{10} & r_{11} & r_{12} & t_y \\ r_{20} & r_{21} & r_{22} & t_z \end{pmatrix}}_{E} \begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix} \tag{2.1}$$

with these parameters:

- f_x, f_y : focal lengths in x and y direction respectively
- s : skew factor describing the pixel form. Practically always close to 1 for rectangular pixels.
- c_x, c_y : shift of the principal point from the center of the image plane. For a real camera this describes the point where the optical axis intersects the sensor plane. In most cases this is close to half the sensor width and height.
- r_i : components of a 3x3 rotation matrix

- $t_{\{x,y,z\}}$: translations along the principal axes in $\mathbb{R}^3$

C is also called the internal camera matrix, while E is the external camera matrix which describes the transformation between the world and camera coordinate systems. Due to the use of homogeneous coordinates both transformations as well as rotations can be described as a single matrix multiplication. Transformation from regular to homogeneous coordinates can be performed by dividing the values for x, y, X, Y, Z with w or W respectively.

f_x, f_y, c_x and c_y can be defined in pixel units as it simplifies computation and is independent of the physical sensor size. However, to derive metric units for depth etc. the individual pixel size on the sensor must be known.

Even with high quality lenses, straight lines in space will not necessarily be projected onto straight lines on the sensor plane. For most cases the distortion model by Brown [25] is still sufficient to describe these deviations of the pixel position from their ideal value. It uses a set of polynomial coefficients to describe the shift of points in radial and tangential direction in relation to the distance from the principal point. This pays tribute to the fact that distortions are usually larger at the edges of lenses and take the form of barrel or pincushion deformations. The model is however not well suited for optics with very low focal lengths (fisheye optics) or for special cases such as tilt-shift optics. In this case more sophisticated models must be used.

The distortion of the point (x, y) can be described by Brown's model as follows:

$$x' = x \sum_{i=0}^{n} (k_i \cdot r^{2i}) + 2l_0 \; x \; y + l_1(r^2 + 2x^2) \tag{2.2}$$

$$y' = y \sum_{i=0}^{n} (k_i \cdot r^{2i}) + 2l_0(r^2 + 2y) + 2l_1 \; x \; y \tag{2.3}$$

with $r = \sqrt{(x^2 + y^2)}$ the radial distance from the camera center (in pixel units), the radial distortion coefficients k_i and the tangential coefficients l_i. Polynomial orders higher than $n = 3$ are seldom used, as the radial distortion are not usually that high in practical cases. Other lens errors such as chromatic aberration can be described when distortion coefficients are estimated separately for different color channels.

Unless zoom lenses are used, the internal camera matrix as well as distortion parameters are generally fixed, although small changes can occur due to temperature differences or when the aperture is changed to account for illumination changes.

Many algorithms for scene reconstruction or SLAM depend on the continuous estimation of the camera extrinsics as it moves around the scene.

For an unknown camera system the task of estimating all camera parameters is known as camera calibration. This is typically done by calculating C and E given a set of known 3D Points with corresponding 2D pixel positions. The system consists of at least 10 degrees of freedom (for fixed skew s and zero distortions). This means at least 5 observations or correspondences (2 degrees of freedom each) are needed to estimate the parameters. Once the camera intrinsics are known at least 3 points are needed to estimate the camera position in consecutive images. This case is sometimes called pose estimation as it involves the calculation of a relative object to camera pose.

As the positions of correspondences are subject to noise, external distortions, etc. in practice more observations are used and an error minimization scheme such as least-squares is used to estimate the parameters.

Commonly used implementations of camera calibration algorithms for example the one from the OpenCV library [24] used in this work are based on works by Zhang [168] or Bouguet [23]. They depend on a target or rig, which contains markers at fixed positions in space which can be easily detected using basic image processing. Examples are planar checkerboard patterns or circles whose corner points or centers represent the feature points used for correspondence estimation. Features which are detectable with subpixel accuracy even in out-of-focus or blurred images are advantageous here. A Rig can be seen in Figure 2.3.

The accuracy of camera calibration is most often judged by means of the re-projection error. This value describes the difference between the found feature positions and the position of the known 3D object points when projected using the estimated camera matrices. Root-mean-square values for this error are usually in the range of 0.03 to 0.1 pixels for state-of-the-art methods using regular camera rigs [9], [168]. In photogrametry, using very controlled environments, values of down to 0.01 pixels are possible. These errors limit the accuracy of correspondence reference data.

A more profound overview over camera geometries and calibration methods can be found for example in [67] or [126].

We can take advantage of the fact that camera calibration uses methods for very exact but sparse correspondence estimation. These methods can be modified to create dense correspondence maps for benchmarking purposes as described in the following section.

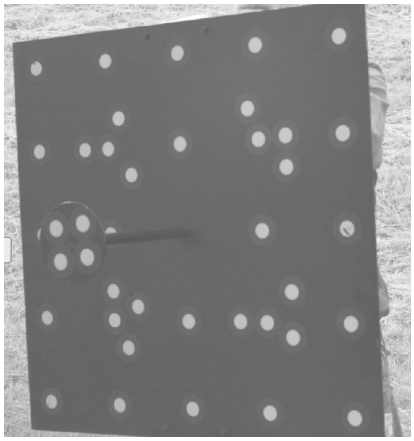


Figure 2.3: Camera Calibration Rig as used in method [9]. Circles or Ellipses can easily be fit to the reflective markers.

Depth and correspondence ground truth using pose estimation

Given an object with known geometry, pose estimation, which is also used to determine external camera parameters, can be used as a method for creating reference data for depth imaging and correspondence problems. Instead of measuring the depth for every pixel, it is only necessary to select a few exact sparse point correspondences to get reference data for all other points.

This method can be applied to systems that produce an intensity image alongside depth data which is the case for e.g. stereo systems, Time-of-Flight cameras, or structured light depth cameras such as the Kinect.

The method can be summarized with the following steps:

- Acquire 3D mesh of object or manufacture object based on mesh
- Record benchmark sequence depicting object
- Perform camera calibration
- Select correspondences in image and object mesh
- Perform pose estimation using the correspondences
- Perform raycasting for every pixel to acquire depth

Given that the internal camera matrix of the system is known (for example by an already performed camera calibration) the transformation between the camera and the object that is used as a test target must be determined. If geometric or optical features such as corners can be found in the image of the object,

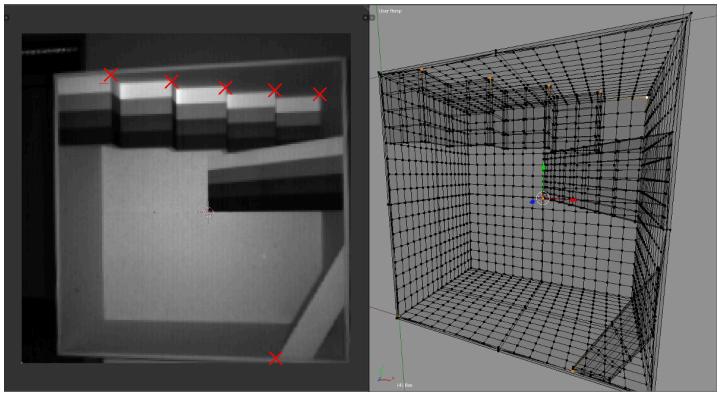


Figure 2.4: Screenshot of pose estimation workflow. Depicted is a 2D image and the 3D Mesh of a reference object. Manually selected correspondences are highlighted. We implemented the algorithm as a plugin for the 3D software Blender [19].

2D-to-3D correspondences can be selected either automatically or manually. This transformation has only 6 degrees of freedom, hence three points are sufficient.

Using a 3D representation of the object (e.g. a polygon mesh) and a virtual camera with the same intrinsics and extrinsics as the real one, the distance from the camera to each object point can be computed for every pixel. Depending on the representation of the object, it may only be possible to compute the depth (or correspondences if multiple cameras are used) for certain vertices (or corner points). However, for pixels in which no vertex is visible, the depth can be computed by interpolating from the nearest vertices. This is done regularly in computer graphic systems using for example raycasting. More details on the problem with this interpolation approach can be found in Section 4.2.1.

The result is a dense depth map for every pixel of the camera where the object is visible. The advantage of this method is that an object can be scanned with a high accuracy method, such as the scanners described below, or even manufactured with known geometry. Using this model it is then possible to create dense depth (or correspondence) reference data for any other modality.

We implemented the method as a plugin for the 3D software Blender [19], a screen shot can be seen in Figure 2.4. The displayed box is used as a depth reference object in several sections of this thesis. The method is used for example in Sections 4.4 or 5.4 to create depth reference data for light fields and Time-of-Flight simulation.

2.2.2 3D Scanning

Various methods exist which can be used to acquire the 3D structure of objects and whole scenes. 3D polygon models or point clouds have many uses, for example for renderings in computer graphics or in industrial inspection (comparing a manufactured object with a reference). Because of this, the accuracy and precision of these methods is well known and multiple industry standards (for example VDI/VDE 2617) are in place to guarantee the reliability of these systems. As described in the previous section, they can also be used to create reference data for various low-level vision tasks.

As a complete examination of all scanning methods is beyond the scope of this work, I will only list those techniques that are best suited for evaluating depth or correspondence related vision tasks. Among others, the described methods are suited for objects between a few centimeters and a few hundred meters in size and operate in the visible or near infrared spectrum. A quantitative examination of different scanners can be found in [81].

Many of these systems are itself based on computer vision, so we are confronted with the problem that we want to evaluate one vision system using another. So in general we employ a fine-to-coarse approach using more accurate systems to evaluate inaccurate ones as described in Section 1.2.2. Which scanner is appropriate for a given scene also depends on the content and scene size.

Structured light scanning For small scenes and objects, *structured light scanners* are one established method. These systems use active illumination to project known patterns (often based on graycodes [59]) onto the object to be scanned. These patterns are distorted due to the object geometry. Using basic triangulation the depth can be inferred. By using multiple exposures of slightly shifted high-frequency patterns, very high resolutions both in depth as well as lateral can be achieved. By capturing multiple depth maps from different angles the complete surface of the object can be reconstructed with resolutions of down to a few microns. As a optical measurement technique it has problems dealing with transparent or strongly reflecting surfaces. One methods to alleviate this problem is the use of diffuse paint, however this may of course change the object properties in a undesired way.

Both precision and accuracy of these scanners is usually way better than that of any depth imaging system targeted at dynamic content such as ToF or stereo cameras. As long as the object pose estimation as described in the previous section is sufficiently exact and under the assumption that there

are no further systematic errors, the data created with these systems can be considered ground truth. Models acquired using this scanning technique are used dominantly in the following chapters.

Laser scanning For larger scale scenes, laser scanners, especially *terrestrial laser scanners* (TLS), have recently received more attention. Dataset which use use these include the KITTI Vision Benchmark Suite [53, 54, 50] and the works by Reynolds et al. [128]. These systems use either triangulation or more often Time-of-Flight to infer depth. Usually, they use a sweeping or rotating beam to scan a scene, which means that the lateral resolution is adjustable. The depth resolution itself is usually fixed. Typical values reach accuracies of a few mm for depths between two and a few hundred meters [21]. Scanners for even larger scenes exist however vision based depth estimation on these scales is practically always based on stereo estimation which can also be tested on smaller scales.

It should be noted that laser scanners show various systematic error sources. How these errors influence the quality of reference data must be evaluated on an individual basis. A detailed examination of the errors can be found for example in [21] and correction techniques are described in [103]. Most of these errors are also present in other depth imaging modalities such as the later described Microsoft Kinect or Time-of-Flight cameras. Problematic are for example occlusions and depth discontinuities which may cause flying pixels or depth results which are material dependent.

Sensor fusion based approaches 3D meshes can also be created by combining depth maps from different depth imaging modalities such as stereo. The resolution of the resulting mesh can actually be higher than the accuracy of the individual depth maps. The KinectFusion algorithm [79] described in the next section (2.3) is an example for such a method.

Industrial inspection systems The here mentioned methods represent only a small subset of available scanning techniques. Physical contact scanners for example belong to the most accurate but also most expensive and slow class. Furthermore, X-ray computer tomography (CT) has been used both in medical applications as well as for industrial inspection. They are practically unrivaled regarding their accuracy but as potential ground truth data is also limited by e.g. camera calibration, there are few cases in image processing were they can be utilized to their full potential.

An overview over the technical characteristics of the mentioned scanner types can be found in Table 2.1 or [81]. The given values are mostly valid for commercial products intended for professional or industrial use. For most scanner types there

	Structured light	Laser scanner	KinectFusion	Contact scanner
Scan Frequency (10^3 points/s)	$10 - 100$	$1 - 100$	700	≈ 0.1
Scan Range (m)	$0.2 - 5$	$0.5 - 10^5$	$0.5 - 7$	< 4
Accuracy[3](mm)	$0.01 - 0.1$	$1 - 100$	$5 - 80$	≈ 0.003
Cost (1000€)	1 - 100	50 -300	0.5	50 - 1000
Advantages		Outdoor capable	Very mobile	Reflective/transparent objects capable
Disadvantages		Self-occlusion problematic	Material compatibility	Only rigid objects

Table 2.1: Properties of different 3D scanner types. The approximate price for the KinectFusion system consists of the camera and a PC with sufficient processing power. Other technical values were taken from [81] as well as different scanner manufacturing homepages[4].

are also less expensive versions available. It is for example possible to construct a structured light scanner using only a video beamer and a webcam. Cheap laser scanner system based on webcams and modified laser pointer are also available [162]. Accuracy of these low-end systems may be lower or their interfaces may be less sophisticated but otherwise the same considerations apply.

Additional information on the use of 3D scanning for the evaluation of Time-of-Flight cameras can be found in Section 3.1 of [7].

[3]It is common practice to state either accuracy and precision or accuracy and repeatability. All of the presented scanning methods have precision values that are at least on magnitude better than their accuracy.

[4]Including Mitutoyo USA (`http://ecatalog.mitutoyo.com`), RIEGL Laser Measurement System Gmbh (`http://www.riegl.com`) and AICON 3D Systems (`http://www.aicon3d.de`)

2.3 Creating Reference Data with KinectFusion

As previously discussed, 3D models and meshes can be used both for creating real-word ground truth as well as for synthetic evaluation sequences. The quality of the reference data does depend highly on the quality of the meshes. The scanning techniques discussed in Section 2.2.2 can reach very high precision and accuracy values although they are also slow. This represent a bottleneck for benchmarks which aim at covering a large application domain.

The Microsoft Kinect camera presented in 2011 is an interesting alternative as it can can produce reasonable accurate depth maps fast and at a low price. The KinectFusion algorithm can create 3D meshes from these depth maps that are more exact than than the camera itself and does run on commodity hardware.

In this Section I will describe how the KinectFusion system can be used as a source for fast reference data creation. Furthermore, this Section has been partially published as "When Can We Use KinectFusion for Ground Truth Acquisition?" [4].

2.3.1 Algorithm and Capturing of 3D Models

The Microsoft Kinect camera uses an active stereo approach by projecting an infrared point pattern into a scene to infer depth information. An infrared camera in the device detects individual subpatterns and estimates their disparity using block matching. From this disparity the depth can be computed. In the KinectFusion system [79] by Izadi et al. this depth data (which can also come from other, similar devices) is used to produce a 3D volumetric reconstruction of the scene.

This is done by integrating the data into a regular voxel grid structure stored on the graphics card (GPU). Surface data is encoded *implicitly* into voxels as signed distances, truncated to a predefined region around the surface. The surface data is continuously updated as new values are integrated with the existing ones using a weighted running average [37]. The global pose of the moving depth camera is predicted using a point-plane iterative closest point (ICP) algorithm [165] while drift is mitigated by aligning the current raw depth map with the accumulated model instead of the previous raw frame, hence the name *Fusion*.

Additionally, the geometric isosurface from the volumetric data can be extracted using a GPU-based implementation of the marching cubes algorithm [105]. For each voxel, the signed distance value at its eight corners is computed. The algorithm uses these computed signed distances as a lookup (into a table stored

as a 1D texture on the GPU) to produce the correct polygon at the specific voxel. This results in an exported mesh in a common format that can be used in 3D modeling applications such as MeshLab[5].

The entirety of the so produced data, consisting of 3D meshes, voxel volumes, synthetic and raw depth maps, RGB images as well as camera poses (location + orientation) can be used for various vision-based tasks: For example, the 3D models with known accuracy can be used to evaluate other reconstruction algorithms such as structure-from-motion or shape-from-shading approaches. Combined with the RGB data, new images of the object can be synthesized and by transforming it based on the known camera movement, optical flow maps can be generated.

2.3.2 Improvements and Alternatives

Multiple improvements or alternatives to the orignal KinectFusion algorithm have been published. Some of these remove certain scene restrictions which makes them even more versatile for the creation of reference data. One of the biggest drawbacks of KinectFusion is the limitation to a fixed scanning volume which is defined by the graphics card memory. This limit has been reduced by Roth et al. [130], in Kinfu Large Scale [71] or by Zeng et al. [167] by using more efficient memory representations or by moving the bounding volume. Kintinuous by Whelan et al. [159] effectively removes this limit by converting and moving of volume data to main memory. Additionally, methods based on similar principles but different representations of the scan volume have been described, for example by Keller [88], Henry et al. [70] or Stückler et al. [146]. A more detailed overview can be found in [1].

In the following Section a few test datasets are compared to high-accuracy, high-cost scans to evaluate the absolute quality of this method, with special emphasis being put on the geometric accuracies.

2.3.3 Quality Analysis

To analyse the accuracy of the KinectFusion method in different scales three test scenes with reference data were created. The test scenes are called ***Statue***, ***Targetbox*** and ***Office***. Many depth cameras and 3D scanners have optics with a fixed focal length as well as a minimal and maximal acquisition depth. Most of them are tailored for indoor use where typical scene depths are below 10 m.

[5]Meshlab software, `http://meshlab.sourceforge.net/`

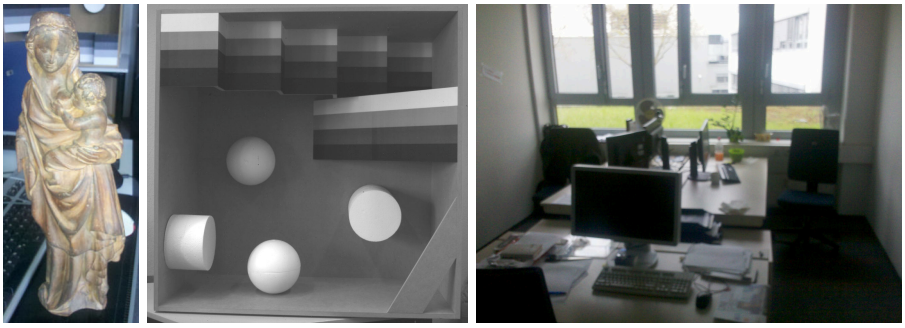


Figure 2.5: Photos of the three test scenes: statue, targetbox and office. The statue is about 40 cm in height, the test box about 1m and the office is ca. 4 by 6 meter. This shows the different scalings at which the system can be used.

This is a limiting factor for the size and resolution of the scenes or objects one wants to scan using these devices. We therefore selected the test scenes to fall into different scale ranges within these limitations.

Although the KinectFusion system is able to work with different depth data sources, the quantitative experiments are limited to the original Kinect sensor. Examples where other input was used can be found in Section 2.3.4.

For each scene the mesh generated by KinectFusion was aligned to the ground truth data using a standard Iterative Closest Point algorithm [16] (ICP) implemented in Meshlab. An initial guess for the ICP algorithm was provided by manually selected correspondences.

Several error measures can be used to evaluate the quality of the meshes.

Euclidean error This error metric is defined for every vertex of the test point cloud. It describes the Euclidean distance to the nearest polygon or face in the reference mesh. For the statue and targetbox scene the KinectFusion generated mesh is the test mesh while the ground truth data is used as reference. For the office scene no ground truth mesh data was available, so we used the ground truth point cloud data instead as test set and the KinectFusion mesh as reference mesh. A closest point metric between two point clouds would not be useful here as due to the different scanning methods the resolutions differ.

angle error This metric is again defined for every vertex of the test cloud. It describes the angle between the normal at the test vertex and the normal of the closest vertex in the reference point cloud. It is more sensitive to corners and depth discontinuities and allows evaluation at sections which

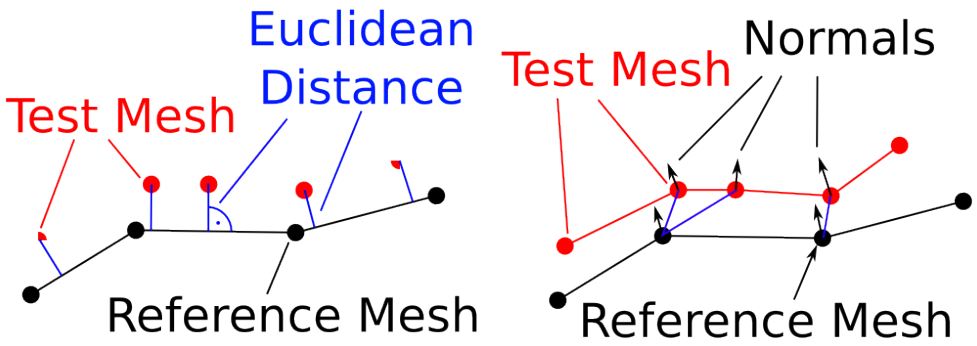


Figure 2.6: Euclidean Error: blue distance vectors. Angular Error: angle between the normals connected by the blue lines.

are critical to some image processing algorithms. Additionally it allows for the evaluation of surface curvature. However, large differences in the mesh resolution can make this error less descriptive as it is also more susceptible to noise.

Average, median and quartiles of these values are a good indicator of the overall reconstruction quality. Large deviations for a small amount of vertices is practically unavoidable, so outliers will occur but are not very descriptive of the reconstruction quality.

If not mentioned otherwise, the values in all images are linearly scaled according to the displayed colorbar. The values for minimum(blue) and maximum(red) are each mentioned in the figure captions.

Statue Scene:

The first scene is composed of an approximately 40cm high wooden statue. According to Khoshelham [89] the Kinect has an accuracy of several millimeters for close distances and up to 4 cm at the maximum range. The fine structures of the statue are well in or below the close range limit. The surface material (wood) is well suited for depth estimation using active illumination as it shows only limited specular reflexes which could cause problematic. The aim with this statue scene is to evaluate the lower limit of resolution KinectFusion can provide (cf. Figures 2.5).

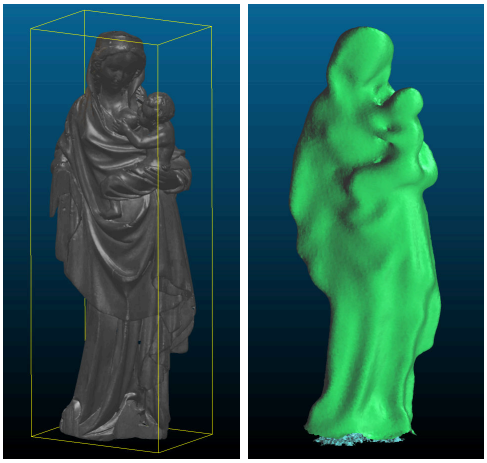


Figure 2.7: Comparison of reference(left) and KinectFusion mesh(right) of statue. Details which are approximately smaller than 1cm are not present in the KinectFusion mesh.

Ground truth for this scene was generated by scanning the statue with a structured light-scanner[6]. The scan contains approximately 2.5 Million Triangles. Scan data for this scene and the next scene was acquired by Julia Freudenreich, Anja Schäfer and Susanne Krömker of the Visualization and Numerical Geometry Group, IWR, University of Heidelberg).[7]

For the reconstruction the KinectFusions implicit voxel volume was chosen to be as small as possible ($(0.8m)^3$ in this case). Smaller values are theoretically allowed by the software but resulted in in frequent loss of tracking. The reason is that the camera must always stay centered at the object to be scanned. Geometry outside of the scanning volume is not considered for camera tracking or icp evaluation.

The resolution of 512^3 voxels is close to the maximum (600^3) the used graphic card[8] could handle and accounts for voxel side lengths of $\approx 1.6mm$. The memory requirements do scale with the third power of the volume resolution, so usage of additional hardware is no valid option to increase it. The camera/object distance was approximately 1 meter in this case which is close to the minimum distance where the system does still work. At closer distances self occlusion as well as the increased disparity range become difficult to handle.

[6] Breuckmann smartSCAN-HE, resolution of down to 10 microns depending on field of view

[7] Additional thanks to the Heidelberg Graduate School of Mathematical and Computational Methods for the Sciences (HGS MathComp) for providing the hardware.

[8] nvidia GTX 480 with 1.5 GB Ram

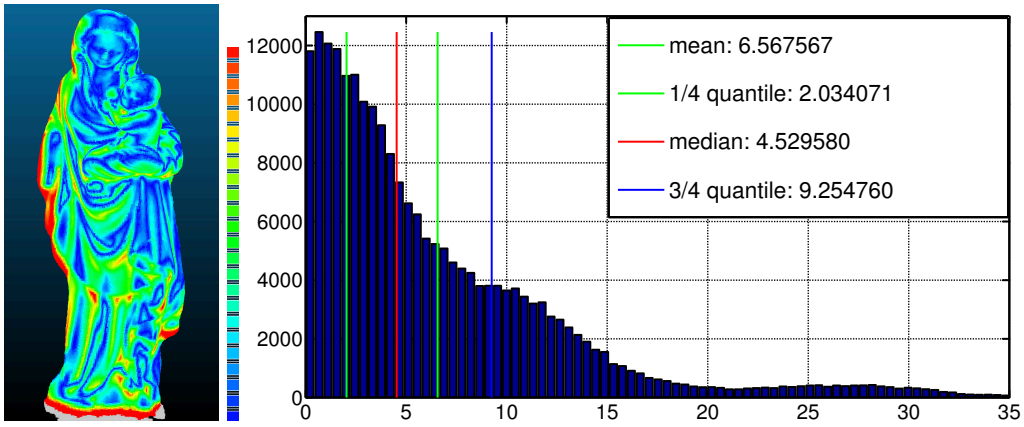


Figure 2.8: Euclidean error of statue (0-8mm, $>$ 8mm is gray), Histogram of Euclidean error. The largest errors are located at the foundation of the statue where either scan was incomplete.

Figure 2.7 shows that the general shape of the statue could be retrieved by KinectFusion but finer surface detail such as facial features is lost. The histogram in Figure 2.8 shows that at least half of all estimated surface points are closer than 5mm to the correct value. Additionally, 75% of all points have an error smaller than 10mm. The highest error values were mostly caused by points at the statue base which were cut off from one of the scans. Other regions which are problematic are folds in the garment. In these concave regions (red in Figure 2.8) the KinectFusion mesh partially follows the outline of the convex hull of the statue, meaning that small indentations are smoothed out.

This resolution limits seem not to be caused by the voxel scaling which is at least a factor three smaller. Lateral and depth resolution however are in the same range or higher so the algorithm is at least partially able to surpass these limitations.

Hence, it seems reasonable that the system can be used for tasks where 10mm resolution or better in absolute coordinates is sufficient.

The error of the surface normals is widely distributed (See Figure 2.9), mainly due to concave sections such as the folds in the garment. From this result one can conclude that highly curved and concave details below the scale of around 10mm cannot be resolved well with the current Kinect system, even if the voxels are small enough. The angle error is however not very descriptive in this case.

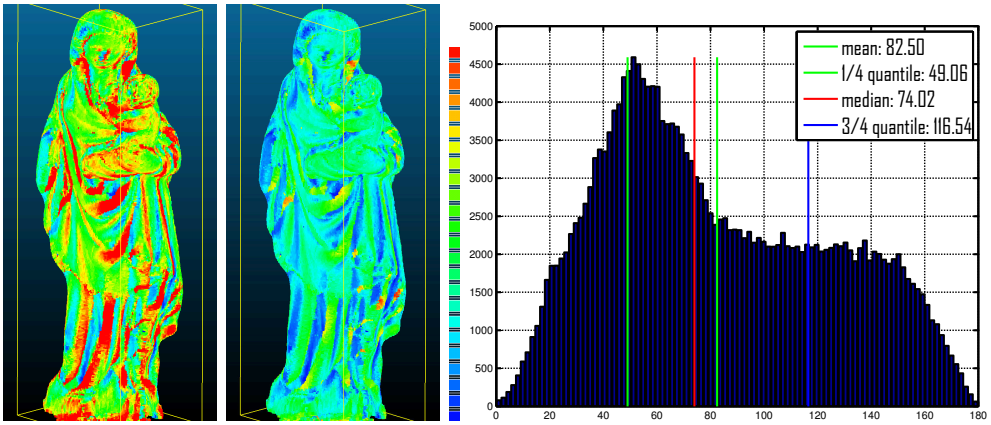


Figure 2.9: Angle errors of statue. Left: angle error$(0 - 90°)$. Center: angle error$(0 - 180°)$. Right: Histogram of angle errors.

Targetbox Scene:

The second test object is a wooden target box especially designed for the evaluation of depth cameras. It is 1 x 1 x 0.5 meter in size and contains several geometric objects made of styrofoam, as well as many regions with slanted surfaces, curvature or sharp 90 degree corners which are typically problematic for any depth acquisition system. (Remark: This box will also be used as a test target in later chapters.)

The 3D model for the box was created by manual measurement and is about $1mm$ exact. Structured light scans would be more accurate but as the box is larger than the scanning volume of most scanners multiple individual scans would be needed to record the whole object. Due to occluded regions it would however be difficult to cover all possible angles.

To accommodate for the box size, the implicit voxel volume was set to to $(1.6m)^3$ with 600^3 voxels, yielding a voxel side length of $\approx 2.7mm$. Only the interior geometry was considered for the evaluation. Erroneous parts on the outside are marked gray in the renderings seen in Figure 2.10. Scanning with KinectFusion was in this case easier as the clear distinct geometry is easier to align using ICP.

The Euclidean error depicted in Figure 2.11 is generally low and in the same range (5-15mm) as in the previous statue scene. Errors higher than $15mm$ result mostly from the free floating artefacts on the top of the box. Only the higher error on the styrofoam sphere suggests that the algorithm underestimates the volume of curved regions. We can conclude that the voxel size of $\approx 2.7mm$ was

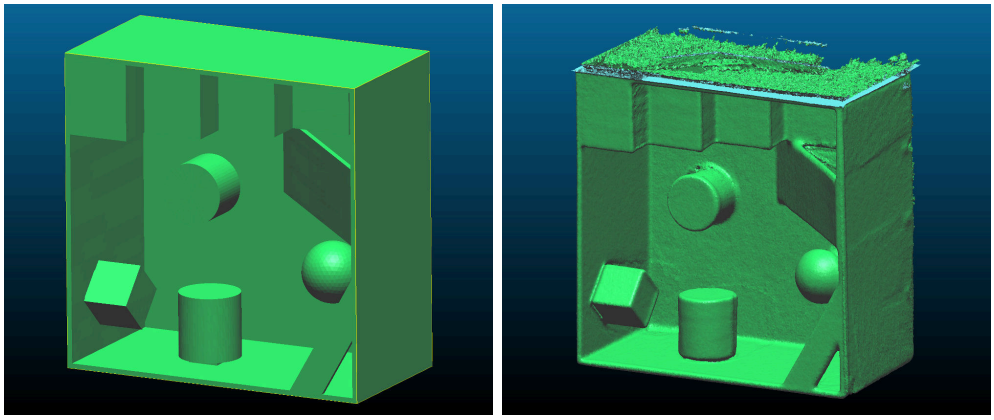


Figure 2.10: Comparison of reference(left) and KinectFusion mesh(right) of targetbox.The topside of the box was not scanned which results in some free floating artifacts. Good visible are rounded corners on the cube and stairs.

sufficiently small for this experiment and that the accuracy of around $15mm$ is also valid for such a medium scale scene.

For this scene the angle errors is significantly better than for the statue scene. See Figure 2.12 for visualization. Angle errors $> 90°$ are partially caused by vertices whose nearest neighbor was matched to one vertex on the other side of the surface. So if a correctly aligned vertex on the inside of test mesh is paired with a vertex on the outside of the reference mesh this would result in an angular error of 180°. The error density is highest around 0° and 180° so we can assume that a significant part of the vertices has an angular error in that region. Generally, surfaces which are flat or have high curvature radii (like the styrofoam sphere or cylinder) are reconstructed well with minimal angular error.

Sharp corners on the other hand are partly smoothed out. This can be seen at object boundaries in Figure 2.12. The histogram density for 45° angle errors (as well as 135° due to the flipping error) is slightly higher than for other angles. This could be caused by the marching cubes algorithm which may add an additional face with a 45° angle if there is a sharp 90° corner in the raw point data.

Office Scene:

The third scene is a small office room of approximately 6 x 4 x $2.5m$. Reference data was acquired by terrestrial LiDAR using a Riegl VZ-400 time-of-flight scanner. Its accuracy is stated with $5mm$. The manufacturer also documents a precision

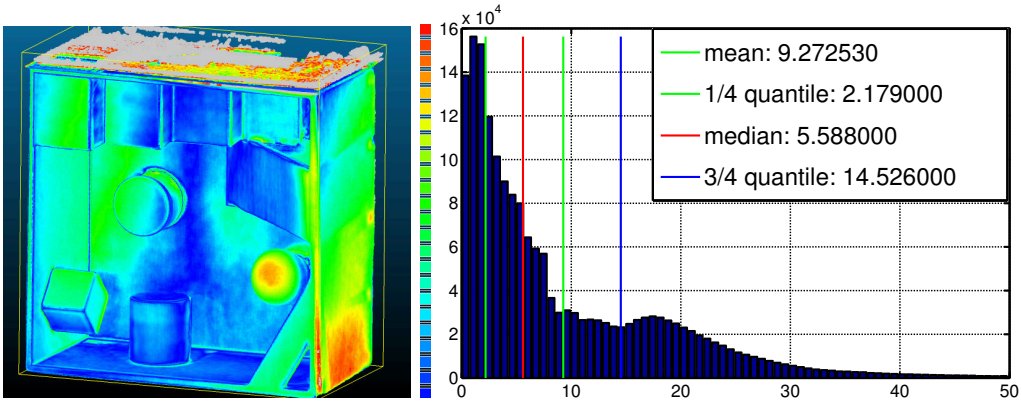


Figure 2.11: Euclidean error of targetbox. Left: 0-15mm, higher errors in gray. Right: Histogram of Euclidean errors. Most errors are well below 15mm. Some outliers are caused by the artifacts on the top and outside of the box.

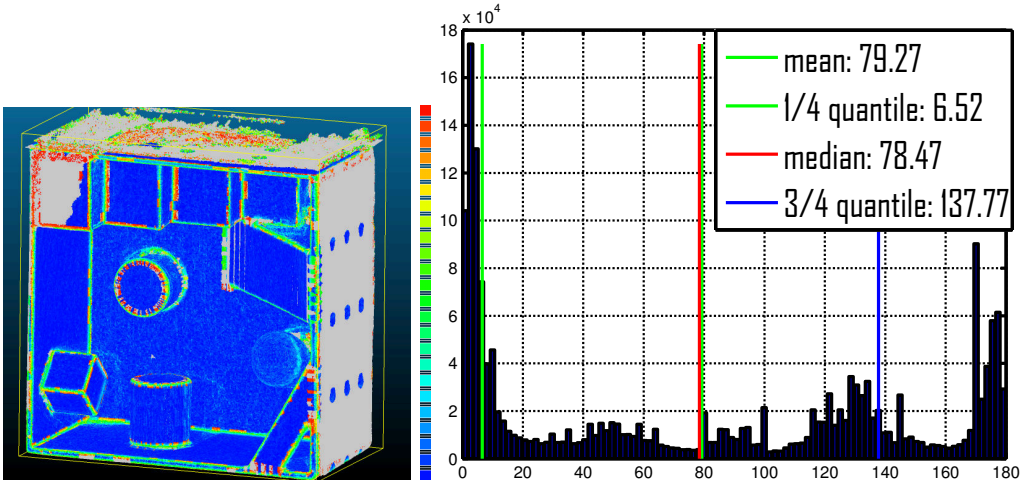


Figure 2.12: Angle error of targetbox. Left: $0 - 90°$, higher errors in gray. Right: histogram of angle errors. Most flat or rounded surfaces show very little angle error. Object corners however are worse, probably due to additional slanted faces created by the marching cubes algorithm.

of $3mm$. Inside the office, overall six scan positions were necessary for a sufficient coverage. The scan data was created by Markus Forbriger, Larissa Müller and Fabian Schütt of the LiDAR Research Group of the Institute for Geography, University of Heidelberg.

In order to fit the whole room into the 512^3-voxel volume we had to choose a voxel side length of about $\approx 13.7mm$ while keeping a scan distance of 1 to 2 meters. This means that the actual lateral and depth resolution of the Kinect system of about 5-10mm can no longer be fully exploited. Given current graphics hardware, the office scene represents the maximum size which can be scanned by the KinectFusion system.

The main axes of the volume do not necessarily coincide with the principal axes of the room so that in the worst case the volume must be scale to the room diagonals, thereby losing resolution. Some of the algorithms mentioned in Section 2.3.2 could be used to handle this problem, however they were not yet available at the time these scans were conducted. Parts of the room such as the walls and ceiling also lack geometric features which results in more frequent tracking loses.

Due to these limitations, a mean Euclidean error of only $50mm$ compared to the $5mm$ accurate LiDAR scanner could be reached.

This scene is highly concave and heavily cluttered, necessitating the use of many different view angles, both for the laser as well as the KinectFusion scan. This is a general problem which applies to nearly all optical 3D reconstructions systems. Accordingly, the reconstruction is not very dense, with many holes or missing parts in the geometry. It can be concluded that very careful acquisition of all concave regions in the office is very challenging with both 3D scanning methods. The resulting scans can be seen in Figure 2.13.

ICP alignment of the KinectFusion mesh and the ground truth mesh are here not perfectly accurate as a small scaling along the object axes was necessary. This is caused by three reasons: first, the scene is heavily cluttered containing many regions were any 3D scanning devices fails due to obstruction. Second, the increased voxel sizes create a coarser mesh which is more difficult to align to the LiDAR results. Third, the LiDAR scan itself is more inaccurate in regions with small scale detail and contains some holes and regions of low point cloud density. The Euclidean error is therefore about one magnitude larger than for the other scenes. Yet, most vertices with errors $> 100mm$ are actually on the outside of the room. This is a result of the marching cubes algorithm which tries to produce closed surfaces. The wall was only scanned from one side and so the algorithm will try to close the surface on the backside with an arbitrary thickness.

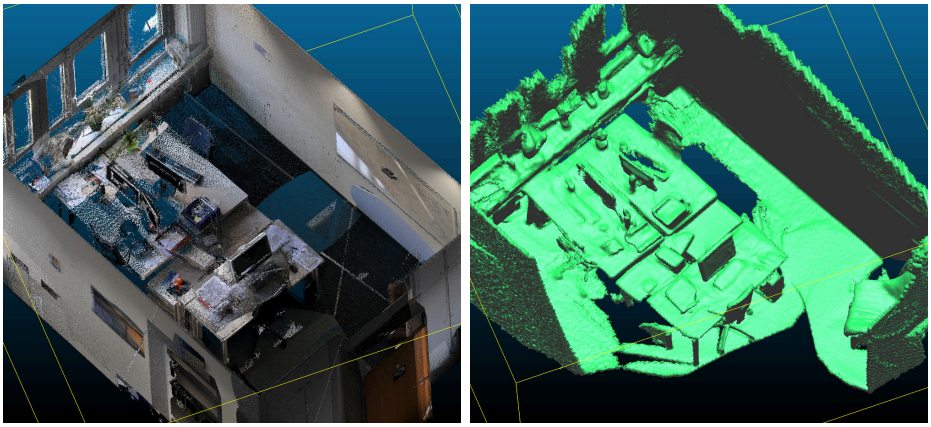


Figure 2.13: Comparison of reference(left) and KinectFusion mesh(right) of office. The laser scan shows a high amount of flying pixels. The KinectFusion scan has some missing sections as tracking errors occurred frequently on the carpet.

As Figure 2.15 shows, the error is well below $80mm$ for most vertices. This is partially caused by the less accurate ICP alignment of test and reference mesh. Additionally these highest errors are caused by regions were both scanning methods fail. Future work should focus on detecting such regions of low certainty in order to mask them out in the resulting benchmark datasets. To get an idea of the accuracy in more confident regions, a robust statistical measure such as the median error can be used whose value is just below three voxel sizes ($36mm$). This indicates that even if the number of voxels were increased, the measurement volume of the current KinectFusion system (at least when using Kinect depth maps as input) should not be much larger than $7x7x7m$ to achieve maximum accuracy.

The problem of the previous dataset where mesh normals were flipped with respect to the correct orientation did occur more frequently for this dataset. The reason is that the LiDAR scanner point clouds did not contain per-vertex normal data which could have helped in the mesh generation process. Instead the reference mesh was created with a ball pivoting approach [15] which did not always yield correct results for surface orientations. Therefore the computation of the angle error was changed slightly by flipping the corresponding normal each time the error was above 90°. The results can be seen in Figure 2.16. By flipping the normals most parts of the ground and walls show now a relatively low angle error which is consistent with the results visible in the rendering (Figure 2.13).

Mainly, high angle errors can be found at sharp corners and boundaries to missing data in highly concave regions. See for example the corners of table and the

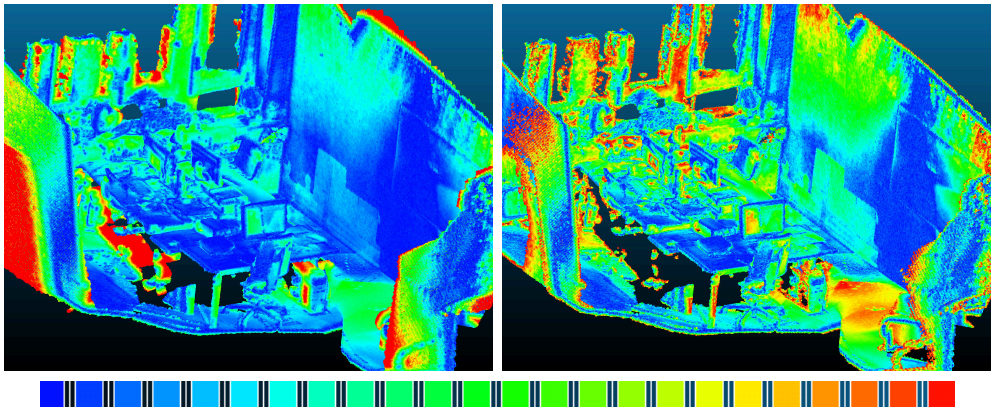


Figure 2.14: Euclidean error of office. Left: 0-100mm, higher errors are red. Center: 0-50mm, higher errors are transparent.

window frames in Figure 2.16. The high error rates at mesh discontinuities shift the error distribution which can be seen at the high density on the right of the angle error histogram (Figure 2.17).

Overall reconstruction quality is actually better than the results suggest. Sections where the mesh was incomplete due to insufficient scan coverage influence the statistics of the errors heavily. A better quantitative analysis could be devised by explicitly masking out holes or invalid regions which were missing from both scans.

2.3.4 Using Alternative Depth Imaging Modalities

In principle behind KinectFusion pipeline is not limited to depth data provided by the Kinect sensor. Yet, we observed that depth map density and noise are limiting factors and not all depth imaging modalities are equally suited as input data. Figure 2.18 shows some exemplary results when either Time-of-Flight data or depth maps from a stereo system are used as input.

The Time-of-Flight based scan of the targetbox in the figure was only possible by using temporal averaging over five to eight frames. This introduces of course severe motion blur based artefacts every time the camera is moved. The experiments showed that ToF is too noisy in the time-domain and additional artefacts (e.g. due to interreflections of the modulated light source) lead to significant geometry distortions. More details on the problems of ToF cameras can be found in Chapter 5.

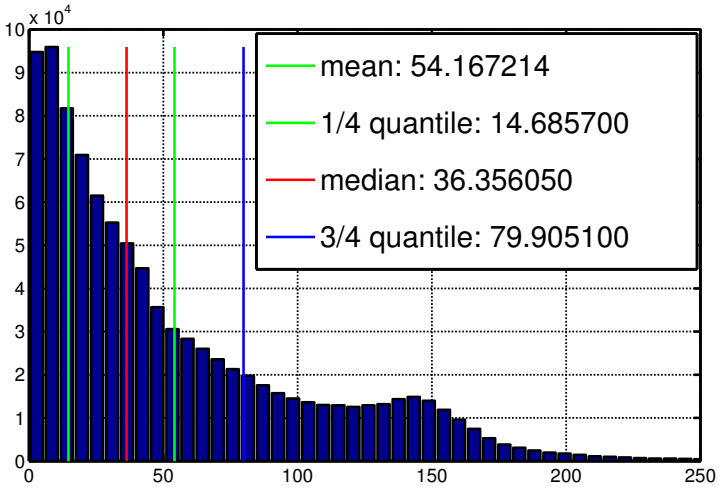


Figure 2.15: Histogram of Euclidean error of office. High error values are caused outside regions on the walls (See Figure 2.14. For correctly aligned regions the error is approximately in the range of the voxel volume size.

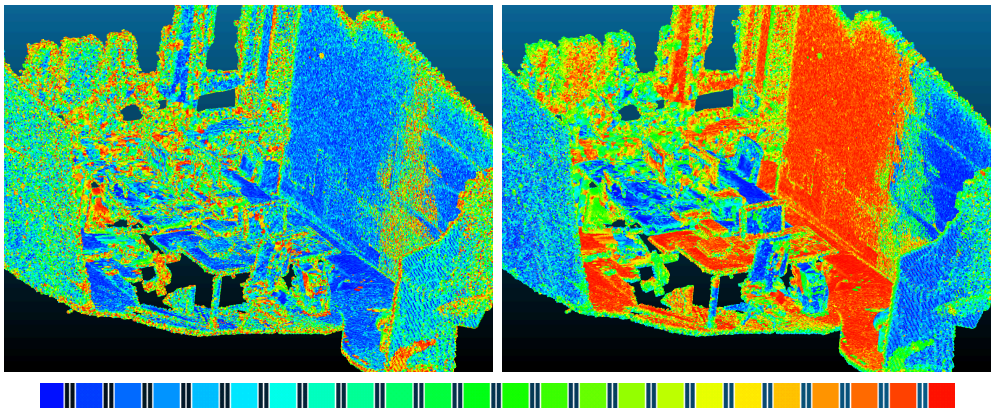


Figure 2.16: Angle error of office. Left Top: $0 - 90°$, normals where the error was $> 90°$ were flipped. Left Bottom: $0 - 180°$, normals remained unflipped.

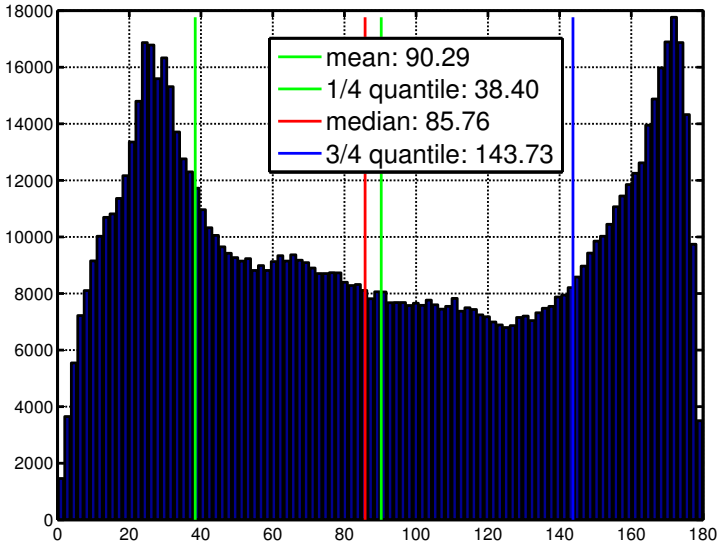


Figure 2.17: Histogram of full angle error.

Stereo data on the other hand often is not dense enough or can contain isolated segments with consistent, but highly incorrect depth.

This can occur e.g. in featureless regions, images of the sky, etc. The right of Figure 2.18 shows an example street scene from the data used in Section 4.3. In this image significant artefacts where removed by hand to make the underlying structure of the building visible. Otherwise the whole scene was heavily cluttered with flying unconnected mesh segments caused by incorrect disparity estimates.

It can be concluded that post-processing as well as data specific adaptions of the KinectFusion pipeline are needed to deal with these challenges.

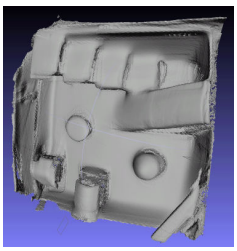


Figure 2.18: KinectFusion mesh using Time-of-Flight data

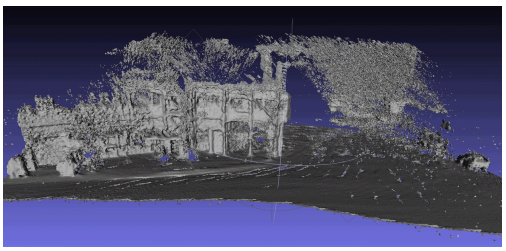


Figure 2.19: KinectFusion mesh using stereo data computed with [75].

2.3.5 Accuracy Analysis for Dense Correspondence Problems

Apart from the geometric reconstruction quality we are interested in whether the Kinect sensor can be used to acquire ground truth for dense correspondence problems. In realtime computer vision applications a sufficient accuracy often is the order of one pixel. Hence, to achieve ground truth quality, the KinectFusion system should record data which is one order of magnitude more accurate.

We synthesized a stereo disparity map and an optical flow field from two virtual views of the targetbox scene (camera distance $\approx 1.3m$, field of view $40°$, maximum flow magnitude ≈ 25 pixel). The flow/disparity at each vertex can be computed by comparing the position of their projections in both views. Values at pixel positions where no vertices are present can be interpolated over the area of a polygon by using the values of the three vertices defining that polygon.

This process was done for both the test and the reference mesh. Figure 2.20 shows the per pixel endpoint error, a widely used error measure for optical flow evaluation [12]. The mean endpoint error for this scene was 0.06 pixel with a median of 0.02 pixel. Most errors occurred on depth discontinuities and edges.

To evaluate stereo disparity accuracies we transformed the depths to disparity values using a focal length of 1100 pixel and a virtual camera distance of $7.5cm$ which corresponds to a regular human eye separation. The mean disparity error was 0.25 pixel with a median of 0.11 pixel (Visualized in Figure 2.22). As most common stereo algorithms have no subpixel accuracy this is a very small difference which should be unobservable in most use cases.

We conclude that KinectFusion based geometry data can indeed be used to generate ground truth optical flow and stereo information in case the application requires accuracies in the order of magnitude of around one pixel. Optical Flow evaluation is hereby limited to static scenes but still useful e.g. for simultaneous location and mapping (SLAM) problems.

2.3.6 Conclusion

This section has shown that the KinectFusion algorithm in combination with the Kinect camera can under certain circumstances be used to create geometric reference data.

The Kinect sensor has several practical advantages over more exact 3D scanners : The setup and actual scanning process is fast as no calibration is needed. Meshed results are available within minutes as in contrast to LiDAR or structured light

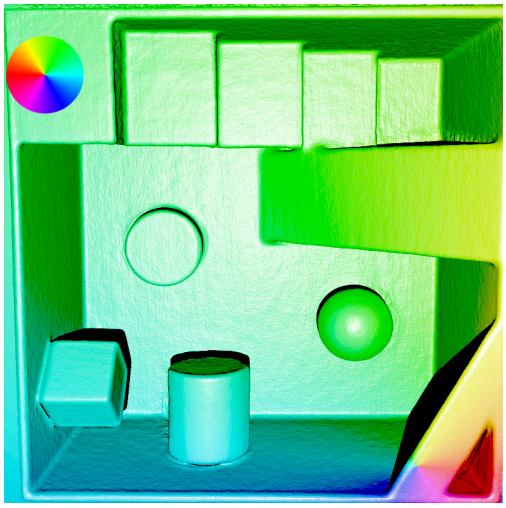


Figure 2.20: Rendering of box with synthetic optical flow as hsv color overlay

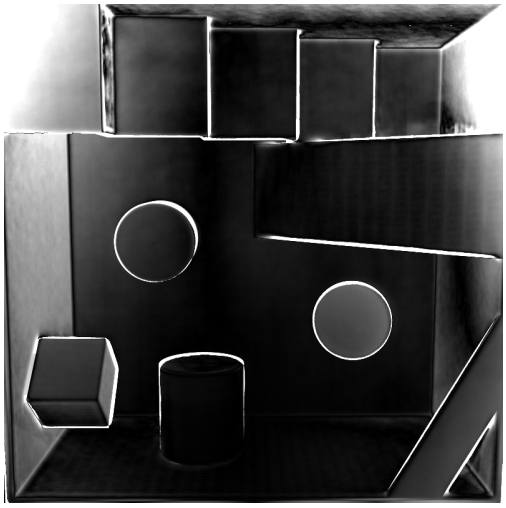


Figure 2.21: Optical flow endpoint error (0-0.2 pixel, higher errors are white) between ground truth and KinectFusion based scene. The largest errors occur at depth discontinuities.

scanners, no extensive manual postprocessing is needed. The Kinect sensor also is more portable and smaller compared to other devices, facilitating the acquisition of additional viewpoints in highly complex scenes. Finally, the effective field of measurement is quite large, closing the gap between portable structured light scanners which are typically restricted to volumes $< (1m)^3$ and LiDAR equipment for larger outdoor scenes. This is useful to create reference data without or only with weak ground truth as many sequences can be recorded in a fast manner. Most of these advantages are also valid for other depth sensors using the same modalities.

The system can resolve object details with a minimum size of approximately $10mm$. This also represents the minimum radius of curvature for slanted or curved surfaces which can be reconstructed reliably. Sharp (depth) edges or highly concave scenes are as problematic for KinectFusion as for many other 3D scanning technologies. For indoor scenes with a volume of $(7m)^3$ this accuracy drops to $\approx 80mm$ with GPU memory and the Kinects minimum object distance as the limiting factors. Insofar there are few 3D or depth scanners which could be evaluated directly with the system as it is located at the lower end of the accuracy scale. Some of the improved methods mentioned earlier are known to produce even better scene representations. At least some of the problems are

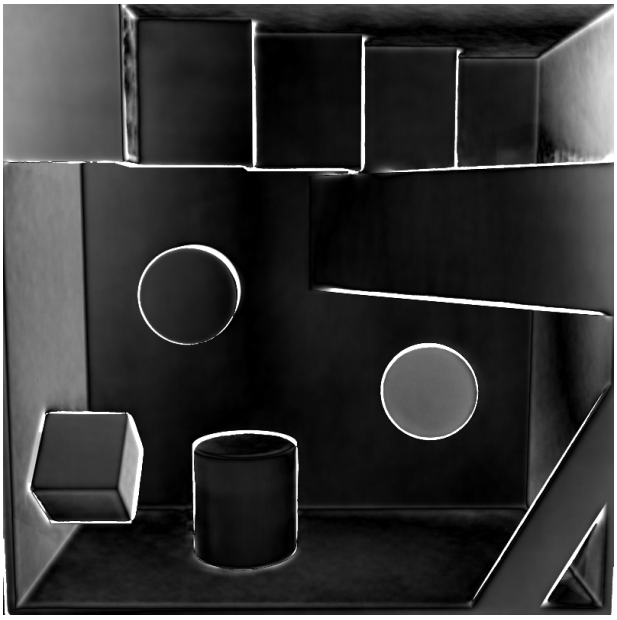


Figure 2.22: Stereo disparity error (0-1 pixel, higher errors are white) between ground truth and KinectFusion based scene.

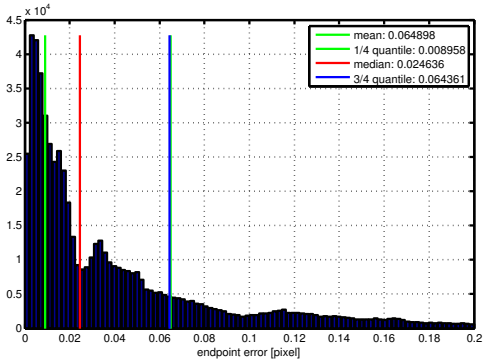


Figure 2.23: Histogram of endpoint errors for synthetic optical flow field.

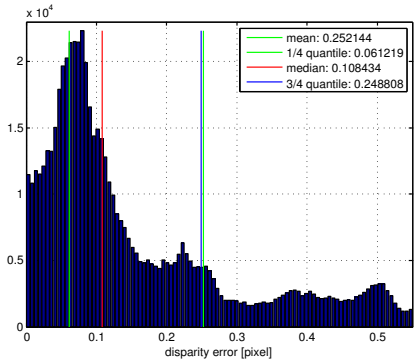


Figure 2.24: Histogram of disparity errors for synthetic disparity map.

related to the mesh generation. Alternative algorithms for meshing point clouds such as the algebraic point set surfaces method [62] could produce better results, although at the cost of higher computation time.

Data derived from the created scans however can be quite exact. Optical flow and stereo ground truth created in the above manner is only 0.1 pixels worse

than high precision systems. This is well within the range of current OF or stereo algorithms and synthetic test data created with KinectFusion should therefore be well suited for algorithm evaluation.

2.4 Summary

In this Chapter we have investigated methods for creating reference data for real-world scenes. The focus was on dense correspondence and depth estimation which are still not fully solved. Some would argue that they are unsolvable in principle [48]. Hence, it is even more important to evaluate algorithms based on this or other geometric problems.

The problem in this field is that many methods which are used for creating ground truth are itself vision based. We must therefore apply the aforementioned fine-to-coarse approach; evaluating cheaper systems using more exact ones.

We have seen that objects with known geometry can be used in conjunction with camera calibration methods to create reference data for depth and correspondence problems. For high accuracy constraints we have identified structure light and laser scanners as appropriate methods for acquiring 3D meshes and object data. An alternative would be to manufacture reference and test objects based on known specifications or blueprints.

These methods however are rather slow and may not be ideal if the goal is to create reference data for multiple objects or scenes. In this cases or when we don't need high accuracies there are alternatives that can create 3D data faster and with much smaller costs. We have investigated the KinectFusion method as a potential substitution.

The system can create 3D meshes with accuracies of a few millimeters and the results suggest that we can create reference data for correspondence problems which differ only slightly from high accuracy methods. Furthermore the system can be used with other depth imaging modalities, a few of which are now entering the consumer market.

The release of the Kinect camera in 2010 for example has sparked great research efforts, also for other depth imaging modalities [1]. We can assume that the recently presented successor of the camera, which is based on Time-of-Flight principles, will have an equally large impact. However, only recently have we begun investigating how these systems can be used for the acquisition of reference data.

3 Real World Reference Data without Ground Truth

3.1 Introduction

Datasets without dense ground truth are often considered inferior to those that contain high quality reference data. Accordingly, there is for one little motivation to create such data, and for another it is difficult to get them published or at least recognized. We have already shown the usefulness of dense and accuracy reference data for development and research but there are additional aspects on the application side which we will now investigate.

Benchmark databases are available for nearly all conceivable tasks and application fields related to computer vision. The CVonline website of the University of Edinburgh for example lists nearly 250 entries[1] and is definitely not intended to be exhaustive. Reference data is often available, but generally limited to single specific tasks. Many datasets, for example the Robotic 3D Scan Repository [116] contain mainly raw data of different formats. But this lack of ground truth does not diminish the usability of these datasets.

Some arguments for datasets without ground truth were already brought up in the sections on requirements engineering (1.3) and weak ground truth (1.4). These included test scenarios for graceful degradation or examples of failure cases.

Most useful is data without ground truth if the results of an algorithm can be judged without a quantitative analysis. For complicated tasks such as decision making or navigation, for example on mobile robots, there isn't a single good quality measure for the result of an algorithm and until one can be developed a human must judge the results. Engineers or even layman can often decide on whether the results of a computation is potentially correct or definitely wrong. This is possible because humans are still superior in many vision related tasks. The decision process is also simply known as eyeballing.

[1] `http://homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.htm`

Another usage scenario for data without ground truth is object detection and identification. A system based on unsupervised learning algorithms for example can be trained on a large set of unlabeled data. Datasets used in this approach are, among others, the 101 Object categories database [46] or the Tiny Image Dataset [152]. Machine learning can also be applied to dense correspondence problems, so that even these field may profit from long image sequences [148].

To show the potential benefits of reference data without dense ground truth we will investigate its use in a specific field of application. Automotive applications are an excellent candidate as it is a broad field with requirements and constraints that expand beyond pure numeric accuracy for single vision tasks. In the following we will first describe a large stereo dataset without ground truth which was used successfully multiple times for various evaluation tasks. The results and experiences from this dataset were subsequently used in a new joint research effort to create a improved dataset.

3.2 Related Work

The datasets presented in the following two sections were inspired by similar works, of which the most important will be listed. Most of the datasets contain stereo sequences as this depth imaging modality is still more robust than any other method in challenging traffic situations.

The aim of the .enpeda project[2] is to collect and present datasets mainly for, but not limited to automotive applications. Currently, it contains 9 distinct sets of data with approximately 3-4 scenes per set. The length of the sequences is typically 100 to 400 frames at 25 Hz with two longer exceptions. Most of the contained data is accommodated by application specific investigations regarding driver assistance systems. The contents of the articles can reach from general purpose examinations to very specific cases. The work by Klette et al. [90] for example examines the performance of correspondence algorithms regarding robustness and stability. To facilitate this effort they presented a set of traffic sequences with ego-motion data which are significantly longer than other datasets for correspondence problems. Furthermore, Barth et al. [13] presented a segmentation method for traffic scenes alongside stereo test data. In contrast to many other segmentation algorithms, their method depends solely on depth information and scene flow. Schauwecker et al. [134] on the other hand provide a dataset on which they investigate, among others, the effect of windscreen wipers on the results of stereo algorithms.

[2] `http://www.mi.auckland.ac.nz/index.php?view=article&id=43`

Figure 3.1: Example scenes from the KITTI dataset [53, 54, 50]

Comport et al. [32] use quadrifocal data to perform odometry estimation for a long car sequence. Interesting is that they don't use reference data from the car itself to test their results but instead superimpose the estimated track on a road map and perform a manual consistency check.

The Cheddar Gorge Data Set is a very impressive contribution by Simpson et al. [142]. It contains grayscale stereo, high-resolution color and infrared images as well as GPS/IMU and odometry data in addition to laser scan data on a single platform. The depicted scene is a single one hour drive trough a about 30 km of a mostly rural area.

The KITTI Vision Benchmark Suite [53, 54, 50] which was already mentioned contains partial depth ground truth based on laser scanning as well as odometry data. It is interesting for this chapter as the contained stereo data is very similar to the other described datasets. The contained stereo frames are synchronized to the laser scanners, hence they have only framerates of 10 Hz, although the raw data may have higher temporal solution. It is also on of the larger available datasets with currently 28 distinct scenes. Examples can be seen in Figure 3.1. Furthermore new scenes are added regularly.

Similar datasets with supplementing laser scanners have been presented in [118] or the already mentioned Rawseeds project[3]. The former is more aimed at robot navigation instead of driver assistance. Datasets for this or SLAM solutions are more numerous but are usually aimed at more controlled environments. Examples are the New College Vision and Laser Data Set by Smith et al. [145] which contains about 300 GB of stereo images as well as omnidirectional camera and laser range scan data.

Table 3.1 gives a short comparison between three well-known databases and the two datasets presented in the following sections. Most of the presented data was

[3] `http://www.rawseeds.org`

	KITTI [53, 54]	Klette[90]	Simpson[142]	Hildesheim^{+} 3.3	Erlensee 3.4
# Sequenzes	28	6	1	15 ($\approx$ 800)	75
Sequence length(frames)	100 - 1K	200 - 400	(57 min)	1K - 2K	5K-20K
# Cameras	4	2	4	2	2
Resolution (spatial,px)	1392 x 512	768 x 488	1024 x 768	656 x 541 (1312 x 1082)	2560 x 1080
Resolution (time, Hz)	10	25	20	25 (100)	200
Bit depth	12	12	12	8 (12)	8*
Odometry	yes	yes	yes	yes	no
Depth GT	yes	no	partial	no	partial

(+)Given are values for published data. Real capabilities in brackets.

(*)12bit with quadratic scaled lookup table.

Table 3.1: Overview over five benchmark datasets for automotive applications. If the dataset contains multiple cameras, the resolution of the primary stereo pair is given.

recorded with rather low framerates of 25Hz or lower. Considering that movement speeds in traffic scenes can be high there is much room for future improvements.

3.3 The Hildesheim Dataset

The acquisition system and data in the following section was first presented in [2] and [5]. The datasets are also available at `http://hci.iwr.uni-heidelberg.de//Benchmarks/`. The data was acquired in cooperation with Sebastian Lauer, Anita Sellent and Wolfgang Niehsen of Robert-Bosch-GmbH.[4]

3.3.1 Design Rationale

Usage of image processing in the automotive sector has a long tradition, ranging from passive driver assistance systems for lane departure warning, over semi-automatic systems like parking assistants to fully-automatic driverless vehicles as for example presented in the DARPA Grand or Urban Challenge [5]. The requirements in this field for both, low-level as well as high-level applications, are very different from typical research considerations. For example, robustness and safety are way more important than e.g. accuracy. A car that doesn't behave as expected due to some erroneous or unexpected input can inflict substantial damage including injuries to bystanders or worse. Testing of such systems is therefore mandatory and does indeed include a significant if not the largest part of development time and resources.

Traffic situations are very numerous and even classifying them is a complicated task in itself. Image sequences of such situations can depict other cars, trucks, pedestrians, animals etc. Traffic participant could be crossing the street, walking alongside or standing in the vicinity. It could be sunny, raining or snowing: situations where even humans with their (still) superior pattern recognition and decision making capabilities fail. Reference data for systems that must deal with these situations must therefore cover a very broad field.

Alone capturing a representative sample from these situations is a daunting tasks. Supplementing those sequences with accurate ground truth for (possibly) multiple image processing tasks is even more complicated. For segmentation or ego motion estimation this has been done multiple times but ground truth for depth or correspondence problems is rare. Examples such as [53, 54] or [127] are very impressive but the data is not fully dense or limited to a few scenes.

Nevertheless, ground-truth-less reference data can still be used for testing purposes. For example, an algorithm that estimates the relative movement speed of

[4] Additional thanks to Annika Berger, Julian Coordts, Tobias Praetsch and Christoph Koke for their programming and post-processing efforts

[5] `http://archive.darpa.mil/grandchallenge/index.asp`

other vehicles shouldn't report incorrect values if there's no other vehicle visible. Generally, no movement ground truth is necessary to determine if there actually is a vehicle visible in a given test image.

The following section will describe a stereo acquisition system that has been developed with the intention of producing large amounts of stereo image sequences under different traffic situations for testing automotive systems. The data produced with this system has since then been successfully used for the development and testing of different low-level vision algorithms.

3.3.2 System Description

This section will shortly motivate and summarize the hard and software used in the camera system. The exact technical specification can be found in [5].

It consists two high-speed monochrome CMOS cameras[6], a standard PC for data recording as well as a combined GPS/IMU system for odometry acquisition.

Most consumer cameras, but also many industrial or research systems operate at rates of 25 or 30 frames per second. This is also true for most of the cameras used in existing datasets. In traffic scenes, where objects can move at several dozen meters per second, those relatively low framerates can produce severe motion blur (depending on exposure time) or temporal aliasing effects. Take for example wheels which appear to rotate backwards if their angular velocity is above a certain value. To deal with such situations cameras that achieve a high framerate of 100 Hz have been chosen.

The cameras lateral resolution was chosen as large as possible respecting technical constraints of manageable datarates. With 1.5 Megapixels the resolution is not particularly high compared to existing consumer cameras, but sufficient for most tasks related to automotive systems.

More important for outdoor scenes is the dynamic range of the cameras as the illumination situation can change rapidly due to sunlight, shadowing and oncoming traffic. The cameras are able to operator in a pseudo-logarithmic mode (using a piecewise linear responsivity curve) to increase their dynamic range. Additionally, they operate at a bitdepth of 12 bits, allowing to work on a finer intensity resolution. To ensure the comparability of the data in a radiometric sense, the cameras were calibrated according to the EMVA 1288 standard [7] for the characterization of image sensors and cameras [43, 41].

[6] Photonfocus MV1-D1312-160-CL

[7] `http://www.emva.org/cms/index.php?idcat=26`

The additional IMU/GPS system provides odometry information with a high heading and acceleration accuracy of $< 1.0°$ and $< 9.8 \cdot 10^{-3} \frac{m}{s^2}$ and an circular error probable of $< 3.0m$. The system operates at frequencies of up to 100Hz reaching the same speed as the cameras.

Figure 3.2: Stereo rig on tripod

Figure 3.3: Stereo rig mounted in car. The GPS/IMU unit is located in the center.

Stereo rectification of the resulting data has been performed using the method by Abraham and Hau [9] with a mean reprojection error of 0.25 pixels.

3.3.3 Data

Multiple datasets have been produced with the described system. The largest one consists of approximately 800 sequences taken the German city of Hildesheim. This amounts to about 10 TB of stereo data. Each sequence is between 10 and 20 second long, yielding 1000 to 2000 frames. This is sufficient to cover situations such as waiting at intersections or train crossings, long overtaking maneuvers or crossing pedestrians. While some of the other presented databases also contain such scenes, their temporal resolution is often limited.

Each of these short sequences is also tagged with various keywords to allow a quick search for conditions which are of interest for a given application or algorithm. Tags include terms like *horizon, traffic sign, clouds, tunnel* and *reflections* etc. Two examples can be seen in Figure 3.4.

The dataset is also interesting as it contains points of the same route at different times. This way an algorithm can be tested on practically the same data but at different illumination or weather conditions. Test evaluations have shown that, while many methods work in favorable daylight conditions, they may fail quickly when confronted with images containing rain or snow.

(a) Tags: predriving cars, oncoming cars, green traffic light, houses on both sides, double tracked, dreary horizon

(b) Tags: fields on right, oncoming cars, single tracked, trees sideways, floodlight, no speed limit sign, horizon

Figure 3.4: Two traffic scenes with example Tags

Some examples for situations which pose a challenge to many image processing algorithms can be seen in Figures 3.5 and 3.6. The examples in Figure 3.6 were part of the HCI Robust Vision Challenge described further below.

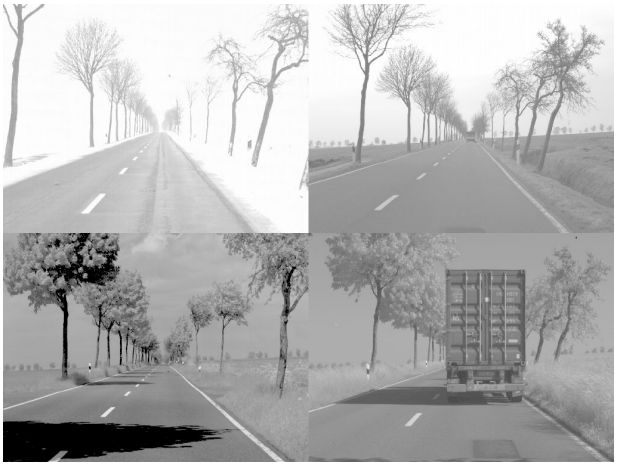
Applications

One example use for the data is a method for optical flow reconstruction on rigid scenes described in [2]. The method uses sparse feature tracking and stereo depth maps to calculate dense movement information for the scenes. An resulting example flow field encoded as an hsv overlay can be seen in Figure 3.7. The method does not provide the high sub-pixel accuracy of state-of-the-art optical flow methods but does not depend on brightness constancy constraints yielding better results in case of moving shadows. This behavior is closer to the requirements of many industry partners which deem an one-pixel accuracy as completely satisfying for most applications. What they consider more important are robustness, illumination independence and predictable and improved corner behavior.

Another application which uses the data is the robust reconstruction of 3D scenes. A method which can jointly estimate the scene depth and external camera parameters was described by Becker et. al.[14] and subsequently tested on the dataset. Their method uses a optical estimation combined with an variational approach to create a dense depth map from as less as two images of a monocular

(a) Intense reflections

(b) Reflections on windshield



(c) Same location on rural street at different seasons

(d) Comparison with VGA resolution system

Figure 3.5: Difficult example situations contained in the datasets

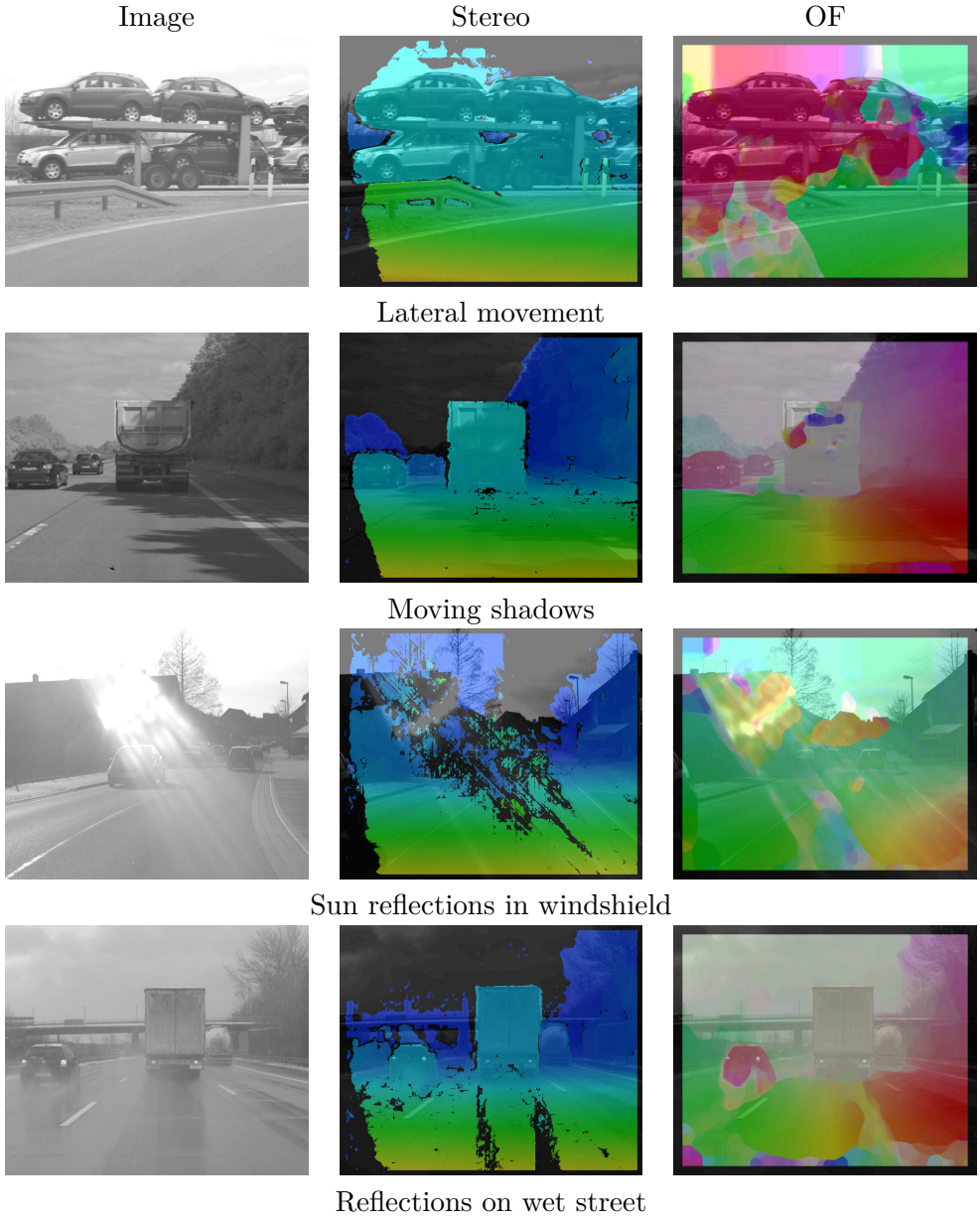
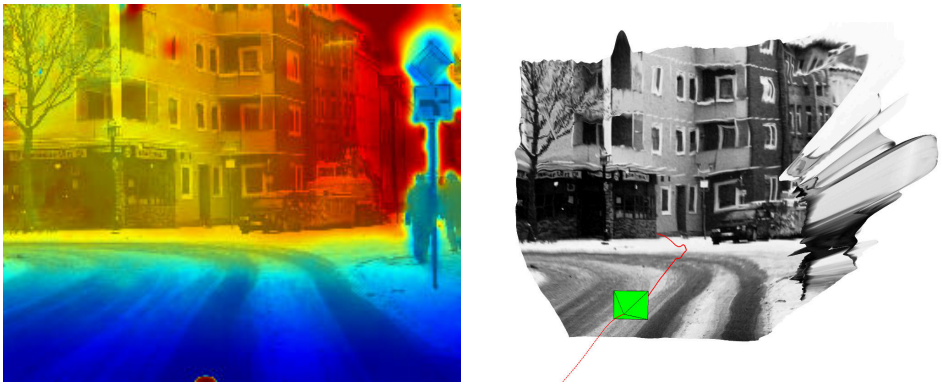


Figure 3.6: Challenging sequences for correspondence problems. Stereo results were obtained using [74], optical flow results using [149]. The first scene shows fast lateral movement, the second scene moving shadows on the back of the truck. Scenes three and four show reflections in the windshield and the wet street.

Figure 3.7: Traffic scene with flow field generated in [2] as HSV overlay

camera setup. See Figure 3.8 for an visualization of the reconstructed depth maps. The dataset was especially useful for the evaluation as the high framerate was favorable for OF estimation while the stereo data could be used to estimate the results of the monoscopic depth reconstruction.



(a) estimated depth as color overlay(arbitrary units, blue=close, red=far)

(b) 3D reconstruction

Figure 3.8: Robust 3D scene reconstruction as demonstrated by Becker et. al.[14] (Images courtesies of the authors)

Finally the dataset has been used in the HCI Robust Vision Challenge [94] presented to researchers and engineers developing optical flow algorithms. The image sequences for this challenge were selected with the intention of demonstrating that optical flow is still an unsolved problem. As such they contain many difficult effects such as big motion vectors, occlusions and repetitive structures. A few examples can be seen in Figure 3.6. Example examinations for stereo and optical

flow were performed using the algorithms by Hirschmueller et al. [74] and Sun et al. [149] which are among the most sophisticated methods in their respective field. Stereo estimation is especially difficult in case of reflections as can be seen in scenes three and four. Optical flow on the other hand has problems with large flow amplitudes, as is evident on the fast truck in scene one and the moving shadows in scene two.

The results of the challenge are presented on the project homepage. The winning methods by Hermann and Klette [73, 72] were both based on semi-global matching and fared especially well in case of disturbing shadows. However, results from the challenge have also shown that most general purpose algorithms fail on at least some of the sequences. Additionally interesting is the rather low number of participants, as the challenge data was downloaded around one hundred times, but only four methods were submitted. It is unknown if this was due to missing interest or if the data was actually too challenging.

3.4 The Erlensee Dataset

3.4.1 Experiences and Rectification

The test data presented here was met with definite success, but certain aspects limited its use. By evaluating feedback from researchers and engineers the following problems were identified:

Information privacy Due to applicable law in Germany and the European union, public datasets may contain no information by which individual persons can be identified unless they agreed to the publication. The presented datasets contain large amounts of pedestrians as well as vehicles which are identifiable by their license plates, etc. As it is not possible to ask all visible persons for approval it is necessary to identify faces, license plates etc. and to remove them before the data can be published. For the large amounts of data this is course tedious work, even if automatic face detection etc. is used. Furthermore the necessary changes to the images limits their usability as censored regions can be considered artefacts.

Search for relevant scenes As the traffic scenes were captured automatically without influence on the actual content, there is a large number of scenes which contain no or only a few interesting events. This means that the data must be searched and annotated manually to sort out data with less relevance. And even though the number of scenes is large, not all relevant events occurred multiple times so that proper comparisons could be made.

Fast movement Despite the high framerate, there were still scenes were the cameras egomotion was so high, that nearly all optical flow algorithms fail [39]. This was the case in curves or when cars where moving fast in lateral direction to the cameras, for example on highway intersections.

Dynamic Range To prevent recalibration of the cameras, multiple scenes were captured with the same aperture and focal settings. Additionally the exposure time was also kept fixed to keep the noise behavior constant. Although the cameras were optimized for low-light conditions the high variability of illumination strength during a full capture session proved problematic. While the relevant regions were not necessarily under or overexposed, the theoretical range of the cameras was not used to their full potential.

3.4.2 Second Generation Dataset

With these shortcomings in mind a new dataset was created in August 2013 with the cooperation of the Robert-Bosch-Gmbh, AEON Verlag & Studio GmbH & Co. KG., the Institute for Cartography and Geoinformatics of the University of Hanover and the Institute for Scientific Computing of the University of Heidelberg. This new dataset consists of ca. 80 individual sequences with lengths between one and two minutes showing traffic scenes in a residential area on the former military airbase near Erlensee, Germany. The pedestrians and other vehicles in the scenes are all committed background actors so that the legal status of the material is clarified. Additionally, the content of each scene was partially choreographed so that each one contains events relevant to automotive image processing tasks such as pedestrians crossing the street, carrying luggage, people suddenly appearing between cars etc. Two example images can be seen in Figure 3.9. The scenes are now also supplied with partial reference data for geometric evaluations as static parts of the scene geometry (buildings, parking cars, etc.) were scanned beforehand using terrestrial LiDAR. As the GPS/IMU information from the former dataset was problematic to synchronize and due to the fact that georectified point cloud data was available, we didn't use external odometry sensors for this dataset.

The following paragraphs give a short summary of the technical details of the sequences.

Stereo camera system Since the production of the previous dataset various advancements in the area of sensor and camera design were made. The new sequences were captured using two pco.edge 5.5 cameras [8] by the PCO AG, Germany.[9] These cameras offer higher framerate, horizontal resolution, significantly lower dark signal non-uniformity (DSNU) and photo response non-uniformity (PRNU) as well as increased dynamic range. This provides us with the following advantages:

Larger field of view The vertical resolution and field of view of the system is roughly the same as for the previous cameras. In horizontal direction the cameras have nearly double the number of pixels, resulting in a very broad field of view.

[8] `http://www.pco.de/categories/scmos-cameras/pcoedge/`

[9] Assembly and programming of the acquisition system as well as partial data recording was carried out by Wolfgang Mischler, IUP, University of Heidelberg.

(a) Traffic scene with pedestrians and vehicles. A calibration marker is visible in the lower right.

(b) Low light conditions with wet street (Logarithmic intensity for better visibility)

Figure 3.9: Example Sequence of 2nd generation dataset

Higher Framerate With 200 frames per second, the system is capable of resolving very fast movements without temporal aliasing. The examination of time slices of the data (Figure 3.12) shows approximate optical flow amplitudes of one pixel per frame or lower.

High dynamic range The cameras can capture difficult light situations like sun reflections as well as dark shadows at the same time and without changing the exposure. Image 3.9(b) was captured at dusk against the sun yet still is neither under nor significantly overexposed. Additionally the cameras produce extreme low noise values of approximately 1.1 electrons at a full well capacity of 40000.

The mounted cameras are displayed in Figure 3.10. Camera calibration and stereo rectification was performed with the same method as in Section 3.3.

Figure 3.10: Cameras mounted in test vehicle. The stereo baseline was at 30 cm

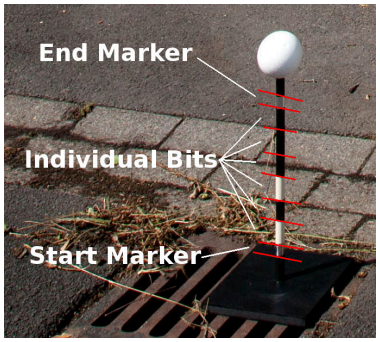


Figure 3.11: Calibration marker # 5 for movement estimation. Each Marker has an individual number which is encoded using a 6bit binary pattern.

The high framerate of the data allows for additional examinations in the time domain. Figure 3.12 shows part of a sequences where the cameras were mounted to the side, capturing a view orthogonal to the vehicles movement direction. Using slices along the YT direction of the data (Figure 3.12(c)) one can for example examine the bending angles of straight structures (such as the window sills) to estimate the pitch movement of the car. The XT slices (Figure 3.12(b)) are similar to lightfield data described in Chapter 4.4. Objects which are closer to the camera move faster along the image plane. This speed and therefore also the distance is directly related to the orientation of the visible structures. The cars backhatch for example is responsible for the black low angle line cutting trough the white ones which are caused by the windows. Structures with an angle larger than 45° move faster than one pixel per frame which is for example a limit for

optical flow estimation. If the flow is larger than this value, methods like pyramid schemes, which can lead to reduced accuracy, must be used.

Partial reference data The whole location was scanned using terrestrial LiDAR before individual scenes were captured. The data was acquired using a Mobile Mapping System VMX-250 by Riegl LMS GmbH.[10] These scans provide geometric reference data with accuracies of approximately a few millimeters. A section of the 3D pointcloud is visible in Figure 3.13 and an aerial view with local point cloud curvature is visible in Figure 3.14. Note the stop sign and lamp post in the closeup which gives a rough estimate about the point resolution. Of course the geometric data represents only a single moment in time and is only valid for static scene components. Moving objects like pedestrians are not included in the scan data and stationary objects like trees etc. may still deform over time due to wind etc. As such the reference data is definitely not dense and falls into the category of weak ground truth.

To estimate the egomotion of the car, additional visual markers were placed in the scene. About 50 of these markers were placed along the scene and their relative position was determined by laser distance measurements. Each marker has an individual number which is encoded in binary on its pole. See Figure 3.11 for an example. By means of pose estimation the relative positions of the cameras to the markers can be determined, similar to the method described in Section 2.2.1.

Metadata To supplement the pure stereo sequences and laser scans, additional data from different sources are part of the dataset. This does include a high number of high-resolution but uncalibrated photos of the surroundings which can be used for alternate 3D reconstruction methods as well as information about sun angle and cloud coverage to estimate local lighting conditions.

No datasets have been published yet as postprocessing is still pending. None the less, it is to be expected that the data will be extremely useful for the evaluation of driver assistance systems, 3D reconstruction methods and similar tasks.

3.5 Summary

In this Chapter we have summarized the potential applications of reference datasets without or with weak ground truth. Reference data can be called weak

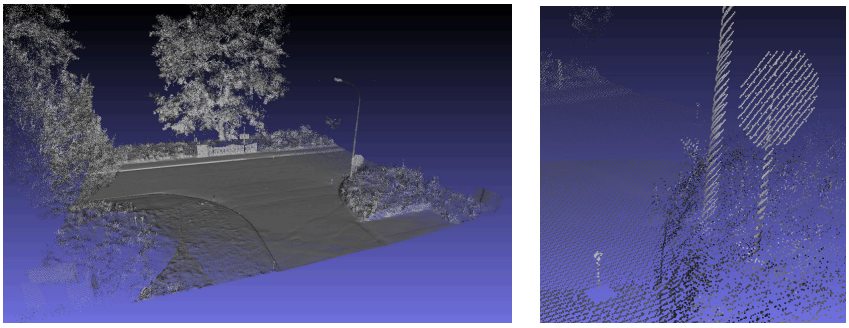
[10] Scans and postprocessing performed by the Institute for Cartography and Geoinformatics of the University of Hanover.

(a) Regular XY frame of side-mounted camera

(b) Slice along the horizontal line in (a). Time in vertical direction.

(c) Slice along the vertical line in (a). Time is in horizontal direction.

Figure 3.12: Slices of XY, XT and YT dimension of side mounted camera. The XY slice in (b) is similar to lightfield approaches. The orientation of the structures encodes their distance.



(a) Overview

(b) Closeup. Note the calibration marker from 3.11 in the lower left.

Figure 3.13: Partial LiDAR scan of scene

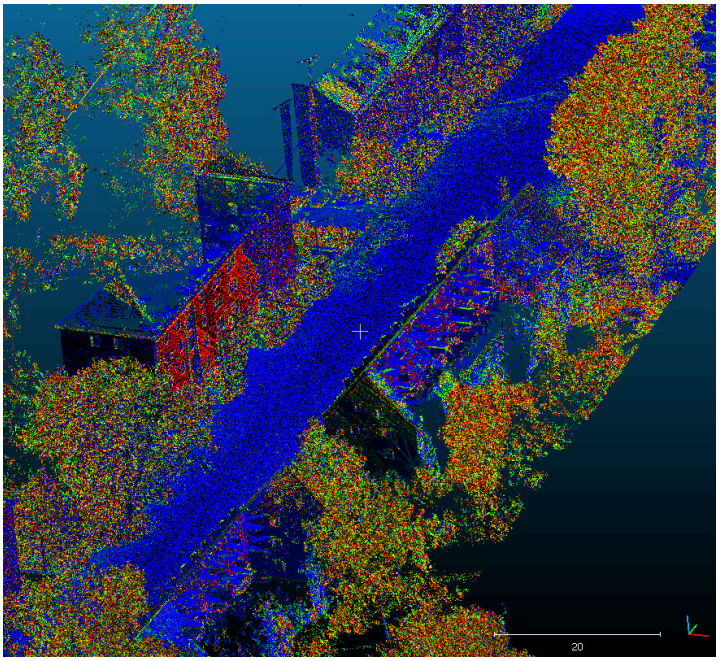


Figure 3.14: Aerial view of laserscan data. The colors depict the local curvature. Flat areas are blue, highly curved or fragmented parts like treetops and undergrowth are red.

if it is less accurate than the system which should be evaluated or not densely available. Datasets of this type can be used in multiple ways, ranging from machine learning applications to scene understanding.

The automotive field is an excellent example for the use of such data as the requirements for image processing algorithms used here are not limited to pure accuracy values. Together with engineers working on these applications, problems of existing datasets have been identified. Available datasets are, with a few exceptions, limited regarding their temporal resolution and show only a very limited set of possible traffic events. This is especially problematic for security relevant applications as the robustness of an algorithm can not be guaranteed if the test space is too small.

Two stereo datasets, both aimed at driver assistance systems, have been created to solve these problems and demonstrate the usefulness of data missing ground truth. The first dataset contains a multitude of traffic situations at different times and weather conditions. It has been used successfully in tasks like monocular camera tracking and optical flow evaluation even though it does not contain reference data for the presented problems. Furthermore, it showed that image sequences from this field still represent a significant challenge to algorithms dealing with correspondence problems.

The second dataset has recently been created to correct shortcomings and problems of the first dataset and is about to be published in the near future. In addition to even higher resolution stereo data it contains partial ground truth in form of terrestrial laser scans as well as additional metadata.

The feedback from both researchers as well as engineers has shown that such data has multiple use cases and the assumption that this is also valid for other fields of application is not far fetched.

4 Real vs. Synthetic Reference Data

Examples from the previous chapters have established that the creation of benchmark datasets with reference data can require considerable effort. There are no theoretical limits to the complexity of the acquisition process, especially if dense per-pixel information about a scene such as depth or optical flow is required. But for the most basic evaluations and proofs-of-concepts, high-accuracy benchmark data is not strictly necessary. Synthetic images have therefore always been used to test the practicality and basic working principles of algorithms.

But once development leaves the early prototype stage these simple test images are often considered insufficient for serious benchmarking. One argument that is frequently brought up, is that synthetic sequences are too simplistic or unrealistic and that only by testing on real data one can reliably judge an algorithm.

Computer graphics has come a long way in the recent years. Large scale media productions like movies and computer games demonstrate to us each year anew what computer graphics algorithms are capable of and it is getting harder and harder even for experts to decide whether an image sequence is real or not. As synthetic images become more and more realistic we start to wonder why we shouldn't use these techniques to create benchmark data customized to any problem we are able to conceive.

And indeed, computer graphics has been used for this tasks, however only on a small scale that is neither related to the amount of needed benchmark data nor to the amount of synthetic images we are confronted with on a daily basis. What is the reason for this lack of synthetic benchmark data and can we overcome the problems related to it?

This chapter tries to answer these questions by investigating certain aspects of ground truth creation for synthetic image data. A full investigation on the state-of-the-art in computer graphics is definitely beyond the scope of this work However such a investigation is also not necessary to demonstrate the problems and principles behind synthetic reference data. The subsequent chapter will also demonstrate some of the work needed to create synthetic benchmark data for a concrete case.

Figure 4.1: Synthetic scenes from the Middlebury dataset [12].

4.1 State-of-the-Art and Related Work

Synthetic datasets for algorithm evaluations are by no means a new development. Although, in some fields of image processing they are used more frequently than in others. This depends partly on whether ground truth can be extracted from the scene representation automatically or if additional postprocessing or manual annotation is needed. The works mentioned here present just a small set of previous and current developments. More detailed lists on available image processing databases including synthetic ones can be found among others at CVonline[1].

Optical flow estimation is a field where synthetic reference data creation is relatively easy while for real sequences it is still considered an unsolved problem [48]. Unlike e.g. in segmentation or object detection, the manual creation of dense ground truth flow fields for existing image sequences is considered impracticable, at least until recently [39]. Synthetic flow fields have therefore long been in use as it is easy to warp an existing image using a predefined flow field or to extract the motion from a full geometric scene representation. A case study on the use of computer graphics for optical flow evaluation is presented in Section 4.3.

Basic examples for datasets are the Yosemite sequence[69], the stereo benchmark by Haeusler and Kondermann [64] or other relatively simple scenes [110]. Slightly more complex 3D renderings based on scanline rendering or simple raytracing have been shown e.g. by Vaudrey et al. [154], Aodha et al. [106] or by Baker et al. in the Middlebury Datasets [12] (Figure 4.1).

Only recently have more sophisticated uses of computer graphics for ground truth generation emerged. An example is the work by Butler et al. [26] in which the

[1] `http://homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.htm`

Figure 4.2: Footage from the movie Sintel [102], CC BY 3.0

authors used scenes from the animated open source movie Sintel to evaluate flow algorithms. Example footage from the movie showing can be seen in Figure 4.2. Additionally, a new stereo ground truth dataset has been presented by Peris and Martull et al. [119, 109].

But also outside of optical flow, synthetic ground truth is getting traction. Haltakov et al. [65] for example presented a framework for the creation of ground truth sequences for driver assistance systems. Isaza et al. [78] use rendered images to create ground truth for shadow detection in outdoor scenes. They assume that the sun is the primary light source in a scene and calculate the shadows projected by geometric primitives and more complex 3D objects. Additionally, Garcia et al. [52] recently presented a framework for the generation of synthetic stereo images.

For the field of medical imaging Prastawa et al. [123] created synthetic magnetic resonance images with ground truth to validate brain tumor segmentation methods. Rezatofighi and colleagues [129] generated synthetic sequences of total internal reflection fluorescence microscope (TIRFM) images. Furthermore Hamarneh and Jassi [66] presented a method for simulating vascular trees with segmentation data which can be used to train and test algorithms that deal with blood vessels or lung airways.

The aim of most of these methods is not photorealism but instead plausible renderings for a specific narrow field of application. More sophisticated methods from computer graphics such as global illumination are rather seldom employed. At least for medical applications, one reason is that experiences from photorealistic rendering can not be directly applied to microscopy images.

4.2 Image Synthesis for Image Analysis

Computer graphics as a research discipline deals among others with two tightly related tasks: *Visualization* and *Simulation*. The final result of both tasks is the creation of new images, however with different intentions.

The goal of visualization is to emphasize or highlight properties contained in a set of data. Examples are the marking and rendering of bones or tissue in CT scans, the visualization of gas flow directions or turbulences in a combustion engine or colorcoding the vegetation types of plants in an aerial image. Simulation on the other hand aims at creating either physically correct or at least aesthetically pleasing images from geometric scene descriptions. Both tasks can be handled partially with the same algorithms. Volume rendering techniques, for example, are used for both, visualization as well as photorealistic image creation. So the true difference lies neither in the results (a image or image sequences) nor in the used tools.

What is different is the handling of information. *Visualization* is a form of *decoding*. Certain information contained in a set of data is extracted and emphasized. For example: Which datapoints in this volumetric block of data describe blood vessels and which describe neural pathways. *Simulation* on the other hand takes structured data (the scene description) and *encodes* it in images. As such, simulation is an excellent tool for benchmarking as we can control which information is encoded in the images so that we can decode it using image processing later on. This does, of course, mean that we have to make certain guarantees about this encoding process:

First, the data we encode must be relevant for the image processing tasks we finally want to solve or at least evaluate. This is related to the above mentioned question of whether ground truth data does actually exist. For geometric scene data, as mentioned earlier, we can model artificial ground truth data with nearly arbitrary precision. The only limits here are memory and computing power.

Second, we must make sure that the process we use to encode the data, namely the render process, represents or reproduces the real image formation process sufficiently exact. While some rendering techniques do only aim at aesthetic convincing results, many achievements were also made regarding physically correct or predictive rendering [161]. It stands to reason that synthetic images can be equally well suited to perform benchmarking and algorithm evaluation as real-world images.

Now the pure possibility to create even good reference data does not mean that it is easy. Unless very high effort is put into the creation, the synthetic scenes may have some shortcomings or be different from real-world images in subtle ways. This may manifest itself in render artifacts, different noise behavior or missing ambient and global lighting. Much of the criticism related to synthetic image data is related to these problems.

The main difference between real and synthetic reference datasets is now: Real datasets may have reference data with limited or unknown precision and accuracy while synthetic dataset will have nearly perfect reference data at the cost of reduced realism. This poses a dilemma for researchers who want to create test cases. Does an algorithm that performs well on a synthetic dataset still do so on real-world data? And even if that is the case, is the performance difference between both types of data (which will definitely exist) a result of the image creation process or based on the different accuracies of the reference data?

Until recently, real-world data was often preferred for all more complicated and high-level vision tasks, and that with good reason. At least this stance was taken by practitioners, not necessarily by developers which is related to the conflict addressed in Chapter 1. The rationale behind it is, that an algorithm that performs well on some artificial and eventually irrelevant data is in the long run inferior to an algorithm that produces only mediocre results but on relevant input data.

The advantage of synthetic scenes does also not necessarily lie in a faster or easier scene creation. Setting up a synthetic scene can be as taxing or time consuming as a real one, possibly even more so. Render times are an additional factor as complicated scenes can take hours or even days to render when sophisticated global illumination methods are used. Where synthetic datasets excel is the possibility to parametrize and automate their creation. When a scene is created it is rather easy to change various aspects, such as movement speeds, materials, camera resolution, etc. This may pay off in the long run if the goal is to create many different test sequences with slightly varying parameters.

Furthermore, the great level of control being available on synthetic scene setups can be very useful to create tests for specific aspects. Time-of-Flight cameras which are dealt with in Chapter 5 for example exhibit different errors which can depend on distance, intensity, stray light etc. In real test scenes it can be complicated to separate and vary these error sources individually. One can, for example, not just double the distance of objects from the camera without reducing the illumination due to the inverse-square-law. In a synthetic setup is is however easy to quadruple the light flux from a light source by just changing a value. This opens the door to creating tests with very fine granularity regarding the covered problems and scene parameters.

It seems reasonable that the advancements in computer graphics can be used to overcome the problems of synthetic reference datasets and that they may be featured more prominently in the near future. Advancements in image based rendering, match moving and possibly even movie production may give us the possibility to produce hybrid sequences which combine the advantages of both

approaches. One could record a realistic street scene and change the illumination and weather afterwards without the need to wait for the right conditions to record the same scene anew.

4.2.1 Practical Problems

Here we will discuss if and how experiences from applied computer graphics such as from the entertainment industry (games, movies etc.) can be used for reference data creation. As such the section deals with the creation of image sequences in scales which are roughly within the range of human perception. Image acquisition tasks that need special equipment (microscopes, astronomic telescopes, imaging outside of the visible spectrum, etc.) may have constraints that can not be simulated or rendered without significant adaption.

The creation of real world test scenes is considered to be an engineering problem and the effort put into it is easily justifiable. Both algorithm developers as well as practitioners have usually some profound insight into the specific problem they are trying to solve. This generally includes access to test and training data and image sequences. Creating real ground truth for existing test setups is therefore the obvious course of action.

Creating synthetic scenes on the other hand requires a quite different set of skills. These skills are at least partly more of the artistic variant as they may include texture creation, scene composition and lighting setup. Software used for this tasks is becoming more sophisticated and user friendly but the learning curves are still high.

The Asset Problem Another problem is the availability of assets, including 3D meshes and textures. To render an automotive scene one would need a mesh describing the street, the surrounding environment, cars, pedestrians etc. Theses objects must have materials and probably color and reflection textures to appear realistic. Additionally, we may want correct movement animation, scripted interactions, etc.

The previous chapter described some 3D scanning techniques that can be used to acquire polygonmeshes at different scaling and accuracy levels. Some of these also capture the per-vertex or per-face colors of the scanned objects, reducing the need to create textures. Current research focuses on measuring surface and material properties alongside the 3D structure, but so far these systems are still in the experimental stage and not necessarily suited for productive use. The

ability to create 3D data gives the user a fine control over the scene content but also increases the needed workload.

An alternative would be the possibility to use existing assets. Examples would be publicly available object or scanning databases. In that case one would still need to combine the objects to a meaningful scene setup, complete with animation etc. Interesting would be the possibility to use footage and production material from existing movies and computer games that are considered sufficiently realistic. That way advancements in computer graphics and content production could directly be used.

Unfortunately, getting access to production material of that high quality is often not possible, due to legal and intellectual property considerations. The amount of material under more permissive licenses such as Creative Commons [2] is however steadily increasing. An example is the movie Sintel [102] that has been used by Wulff et al. [163] for the creation of an optical flow dataset. The option to use computer game engines to create content is discussed in Section 4.5

Limits of Synthetic Ground Truth Reference data based on scene descriptions is usually superior to measured data. None the less, there are still problems and uncertainties related to this data. Some of this is related to how geometric data is represented in computer graphics.

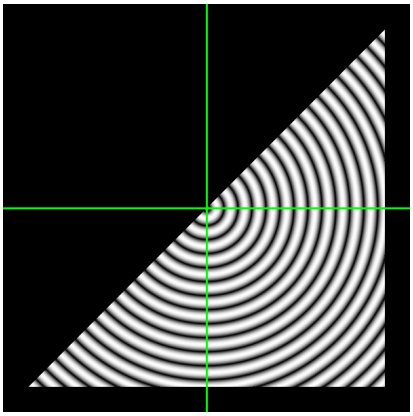
Rendering is often done on the basis of 3D meshes which describe the geometry via points in 3D space (vertices), connections between these points (*edges*) and *polygons* or *faces* bounded by the edges. Additional properties such as color, normals, etc. are often only defined at vertex positions but not on polygons surfaces, unless textures are used. This can pose problems if we want to create dense reference data. Merely interpolating these values across the rendered image can lead to errors such as distortions.

An example is depicted in Figure 4.3. The displayed triangle is moved and rotated in 3D space between 4.3(a) and 4.3(b). Optical flow reference data for these two images can be computed by projecting the transformation vectors at the vertices onto the image plane. As the triangle is characterized by only three points, all intermediate flow vectors must be interpolated between the corners. Using this flow to warp the triangle yields the result in Figure 4.3(c). The resulting image shows huge distortion as interpolating the projected movement is different from projecting the interpolated vertex movement. If the triangle is first subdivided into smaller segments, resulting in a higher number of vertices, before projecting the flow vectors we get much better warping results (Figure 4.3(d)).

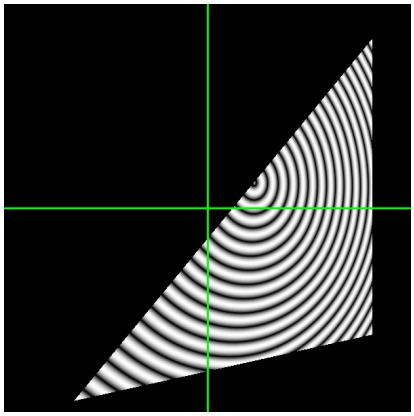
[2] `http://creativecommons.org`

This is but one of many pitfalls that have to be avoided when creating synthetic reference data. There may also be problems of numeric precision, etc. involved. High quality render and modeling software maybe is not necessarily prone to this error. However, in software aimed at artistic or entertainment purposes, such as game engines, smaller errors or inaccuracies may be considered acceptable, especially if performance is an issue. The discrete 8bit color value of a pixel, for example, may be off by one digital value if float precision values get truncated instead of proper rounding. This is hardly an issue for a game, as most humans can barely notice the difference between adjacent colors in the 32bit RGB colorspace. For image processing algorithms however this may be a mayor difference. Similar problems may arise when graphics buffers with low float precision are used or due to poorly documented post processing steps.

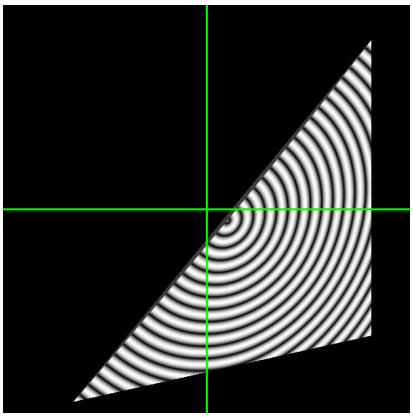
Further Reading Furthermore, physically correct rendering in computer graphics is a very broad topic that cannot be handled in detail here. Interested readers may have a look at [141] by Shirley et al. or Pharr and Humphreys [120] work for an overview. Articles that deal with predictive rendering, the synthesis of images that not only *look* real but can be compared quantitatively to real images are, among others, the works by Wilkie [161] or Ochoa et al. [117]. The global illumination section in Chapter 5 will also give specifics on Monte-Carlo based simulations.



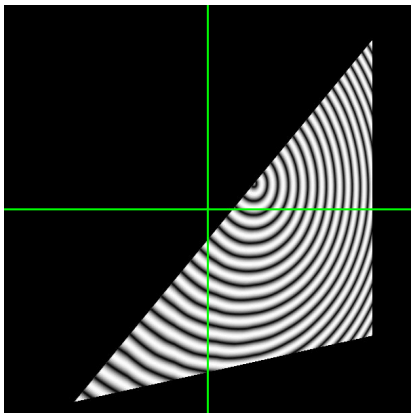
(a) Frame 0 of rendered Triangle



(b) Frame 1 with transformed rendered Triangle



(c) Frame 0 warped with flow of low resolution mesh. Note the center of the rings.



(d) Frame 0 warped with flow of high resolution mesh

Figure 4.3: The shown triangle is transformed in 3D space. Reference optical flow can be computed by projecting the movement vectors at the vertices onto the image plane. Bilinear interpolation of the flow at only the triangle corners yields poor results. Subdividing the mesh into smaller triangles reduces this problem. (Image center marked in green)

4.3 Case Study: Synthetic Optical Flow

The results of the following section were first published in my diploma thesis [2] and in [3]. This summary will put additional emphasis on the results in the context of general dense correspondence problems and the applicability to other vision tasks.

Optical flow evaluation has already been mentioned as an example for the use of synthetic reference data. We will briefly investigate whether there are fundamental differences between the flow fields computed from synthetic images and real-world ones. A special emphasis will be put on the influence of different levels of realism in the renderings.

Optical flow can be measured with very high accuracy. Many state-of-the-art algorithms claim mean endpoint errors in the lower subpixel range. Subsequently even small differences between two images can result in quantifiable and significant flow differences. This vision task is therefore very well suited to estimate differences between synthetic and real world imagery, .

As a form of case study, these examinations can only give hints on whether synthetic reference data is useful. The test scene is very limited and in total only two of the many different optical flow algorithms were used. However, by doing an analysis of individual aspects of image synthesis and putting them in relation to the resulting optical flow results, it seems at least plausible that the findings can be generalized.

The performed examinations are based on works by Vaudrey et.al. [154], who concluded that the mayor problems which make synthetic sequences inferior to real-world sequences is the fact that object and texture boundaries in synthetic images are much more distinguished. Additionally they argue that the brightness consistency between two frames is violated more often in real-world images.

To perform a quantitative comparison a real-world scene and a synthetic one with the same geometry was created. For both scenes optical flow fields were computed and compared to each other and to reference data. By reducing the realism of the synthetic scene one can investigate which effects are responsible for the discrepancies between the computed optical flow of the real and the synthetic scene.

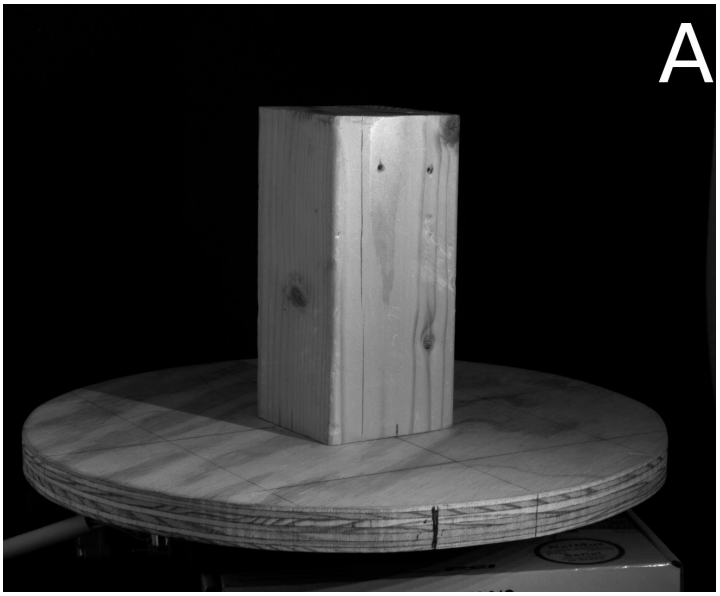


Figure 4.4: Real test sequence

4.3.1 Experiments

The constructed test sequence is limited to a relatively simple geometry to reduce errors caused by stray light, occlusions and specular reflection. It depicts a wooden block standing on a circular wooden plate which is fixed to a stepper motor. Scenes are recorded in a stop-motion manner with the plate rotating at 0.5 degrees per frame. The rationale behind this setup is that rotational motion represents a challenge to typical optical flow algorithms which use spatial smoothing based on L_2 regularizers. These regularization schemes suppresses divergences and rotations in the flow field. A real recoding of the test sequence can be seen in Figure 4.4.

Real Scene

The real scene is lit by a desk lamp consisting of six linearly arranged white LEDs. We chose those deliberately, as they can be better approximated by virtual point or spot light sources. They also produce slightly crispier shadow edges than more spatially extended light sources like light bulbs. Furthermore the scene was shielded against stray and ambient light.

The scene was recorded using a CMOS machine vision camera[3] and a 25mm lens at a distance of approximately 50cm. Lens distortions of the optical system were minimal, but the sensor showed significant *dark response non uniformity* (DRNU) which was subsequently removed by subtraction.

Synthetic Scene Reconstruction

The real world scene was reconstructed with a mixture of classical and image-based modeling and rendering. Polygon meshes for the used objects were created by manual measurement which was aided by the simplistic shapes. The wooden plate was additionally scanned using a flatbed scanner. Final accuracy for the meshes was 1 mm or less which was later confirmed using a structured light scanner. Additionally, the meshes were created with a high subdivision level, resulting in a sufficiently high number of vertices even in areas where one large polygon would be enough to model the surface. This was done in order to avoid the interpolation errors described in Section 4.2.1.

The textures were created from images taken with the same camera system which was used to create the real-world scene. Hence, the textures have the same dynamic range as the real materials and there are no significant intensity deviations. The final texture resolutions were at least two times higher than the size of the objects in the final renderings, so interpolation errors were minimal.

The materials diffuse channel was simulated using the method by Oren and Nayar [111]. By using a microfacet model this shader simulates the surface of natural materials like sand or wood better than a simple lambertian model.

For the specular reflections we used a basic Blinn-Phong shader [20]. This model is slightly inferior to the more accurate Phong model [121] as it computes only an approximation to the correct reflection angle. The problems introduced by this were minimal as the used wood showed only minimal specular reflexes.

Finally, the objects were positioned inside a virtual environment by using the previously captured real-world scene as a reference. Pose estimation for the objects was performed using the camera calibration method by Tsai [153].

In the final renderings the objects are aligned with an approximate error of 2-3 pixels or more at points close to the camera (e.g. the front of the plate). This is caused by errors in the meshes themselves (the plate for example was no perfect disk). Also the actual displacements can seem to be larger due to errors in the texture mapping.

[3] Photonfous MV1-D1312, 1312x1082 pixel

	$\lambda = 100$	$\lambda =$ **200**	$\lambda = 400$	$\lambda = 700$
real(A)	0.51 ± 0.81	$\mathbf{0.48 \pm 0.75}$	0.47 ± 0.71	0.51 ± 0.70
realistic(B)	0.43 ± 0.86	$\mathbf{0.39 \pm 0.83}$	0.38 ± 0.83	0.39 ± 0.90
no specularity(C)	0.42 ± 0.83	$\mathbf{0.38 \pm 0.75}$	0.36 ± 0.74	0.38 ± 0.88
simple shader(D)	0.40 ± 0.81	$\mathbf{0.36 \pm 0.74}$	0.34 ± 0.71	0.36 ± 0.82
minimum realism(E)	0.36 ± 0.74	$\mathbf{0.29 \pm 0.63}$	0.30 ± 0.74	0.36 ± 1.11

Table 4.1: Mean endpoint errors and standard deviations for different regularization strengths. The errors for $\lambda = 400$ were slightly lower than for $\lambda = 200$ but the edge strength at discontinuities was worse.

The scene was illuminated with 6 spotlights which are decent approximations of the LEDs used in the original scene. Global Illumination (multiple diffuse reflections) in the scene was simulated using radiosity rendering which was first introduced by Goral et al. [55].

The original recording as well as the different renderings can be seen in Figure 4.5. The realism was then subsequently reduced by removing effects which could influence optical flow results. First the specularity of the wood material was set to zero, effectively removing all specular highlights (scene C). These highlights are a major violation of the brightness constancy constraint which is modeled in the optical flow algorithm. Secondly the diffuse shader was changed from Oren-Nayar to a lambertian model thus simulating a more simple material (scene D). In a final step global illumination as well as shadows were deactivated (scene E). This last setup bears the most resemblance to the previously mentioned synthetic sequences.

4.3.2 Evaluation

This section will only summarize the relation between simulation quality and optical flow results. A more detailed error analysis can be found in [2] and [3].

The optical flow was computed using a L_1-total-variation based algorithm by Zach et. al. [166] which performs well on the Middlebury dataset. The computations were performed using a 4 level pyramid scheme to accommodate for larger flow vectors. Flow results can be seen in Figures 4.6 and 4.7. Error analysis was carried out by means of the endpoint error and warping residuals. The former is defined as the magnitude of the difference between the reference and the computed flow while the later is the difference image between the second frame and the first frame warped according to the computed flow. Standard error measures for optical flow are defined in [68] (Section 13.8.2).

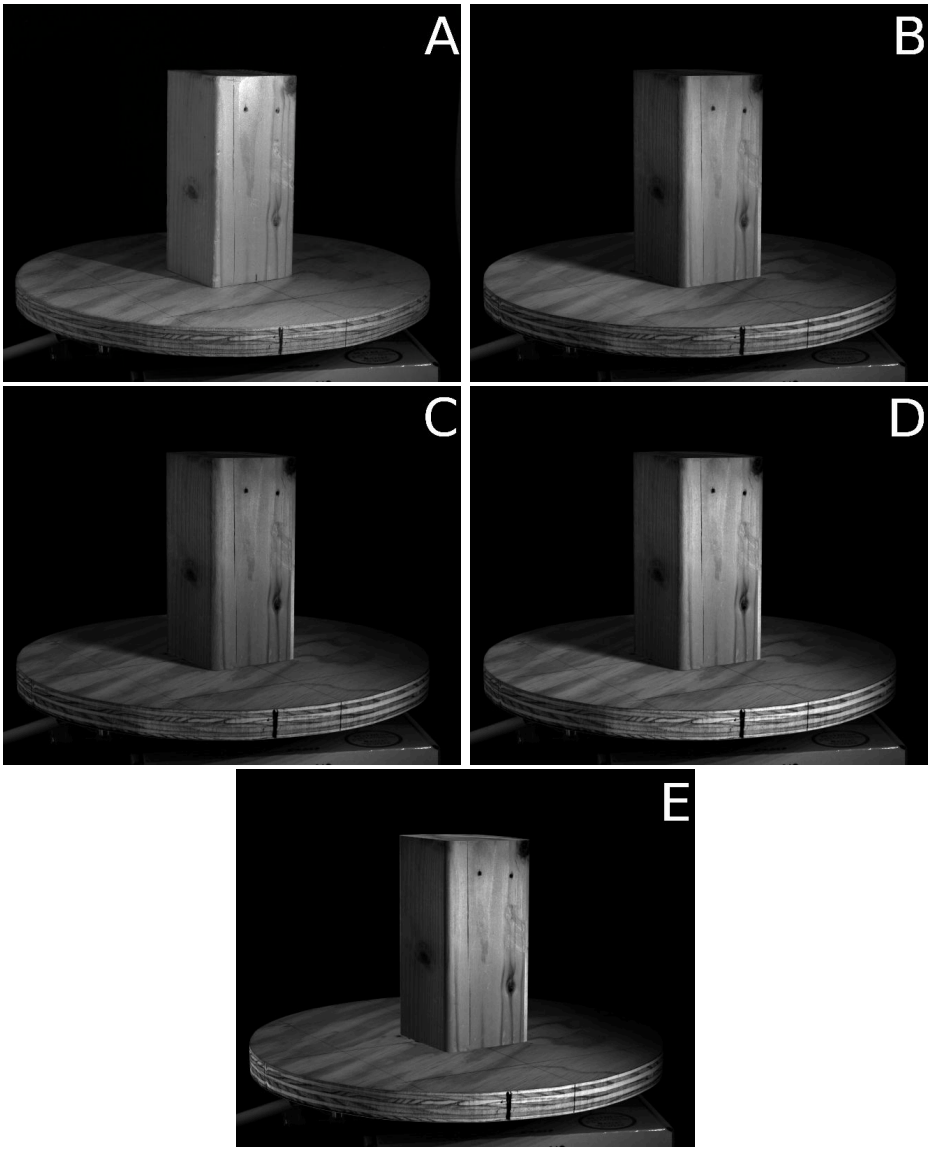


Figure 4.5: Comparison of real and rendered images: real scene(A), realistic reconstruction(B), without specularity(C),with simple diffuse shader(D), minimum realism(E)

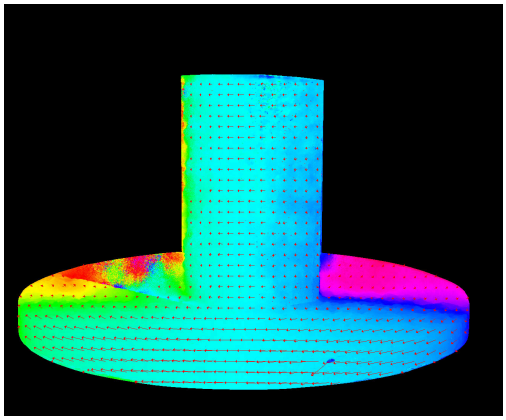


Figure 4.6: Optical flow on real sequences as HSV color overlay

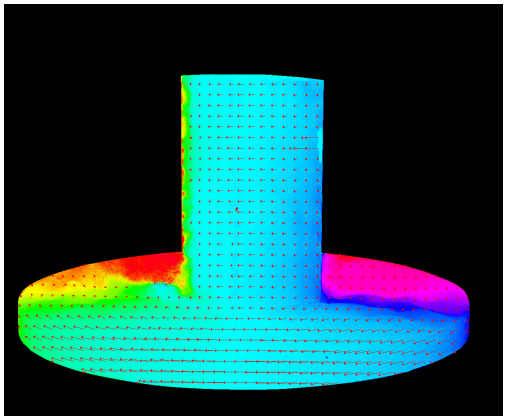


Figure 4.7: Optical flow on rendered sequence as HSV color overlay

All evaluations on the optical flow fields were performed with a regularization strength of $\lambda = 200$. For this value the tradeoff between global mean endpoint error and edge strength was minimal.

Figure 4.8 shows the endpoint errors computed for $\lambda = 200$. The images are normalized to show the same gray value for a given value of the endpoint error. Errors bigger than one pixel are pure white.

The mean endpoint errors as shown in table 4.1 were measured only where actual content was visible. The error of the real scene is the largest which can either be attributed to errors in the alignment or to deviations caused by real-world effects. One of those would be the fact that the back of the plate is poorly lit which makes flow evaluation difficult. However we are only interested in the differences between the renderings, not the absolute error.

The smallest endpoint error was reached for the most simple rendered scene with a mean of 0.29 ± 0.63 pixels (last image in Figure 4.8). This is to be expected as most realistic rendering effects that were omitted in this scene are detrimental to optical flow analysis. Deactivating shadows, for example, improves the image contrast and removes shadow edges which can appear as incorrect flow edges.

One can see that the specularity plays a minor role in this scenario as switching it off leads only to a marginal decrease of the endpoint error. However the error distribution also gets slightly narrower. The effects can best be observed at the right corner of the wooden block, where a small region with huge errors (> 10pixels) shrinks in size significantly once the specularities are reduced. For

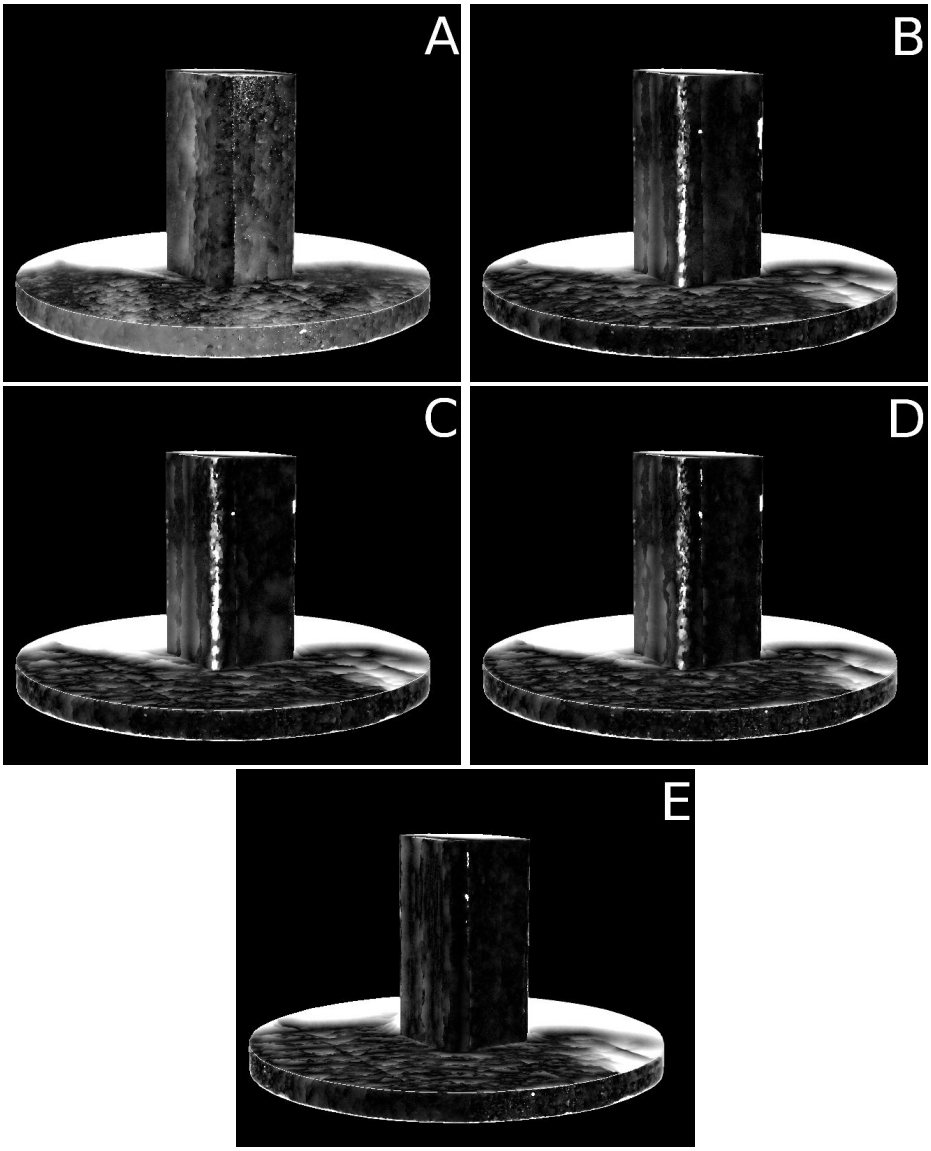
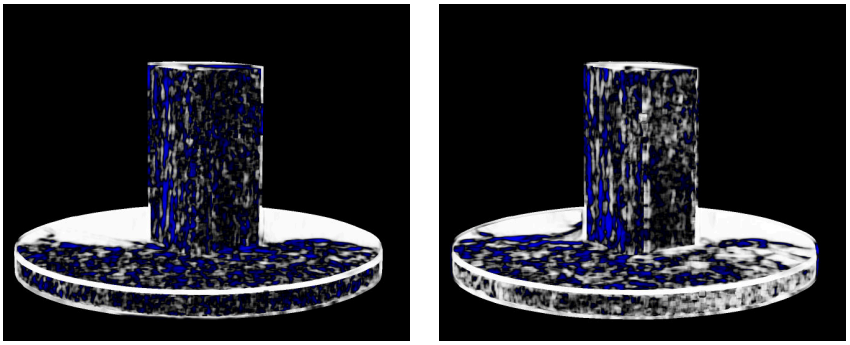


Figure 4.8: Normalized endpoint errors(Errors > 1 pixel are pure white): real(A), realistic(B), without specularity(C), simple shaders(D), minimum realism(E). Good visible are artifacts on the front and top of the block due to texture aliasing.



(a) NCC between the endpoint errors for the real and rendered sequences

(b) NCC between the endpoint errors for the realistic and simple scene

Figure 4.9: Normalized Cross Correlation (NCC). Positive values scale from zero(black) to 1.0 (white), negative values are blue.

scenes or objects where the brightness changes are larger due to distinct highlights the influence should be even stronger.

The endpoint error drops significantly once the reflectance model is changed to a pure lambertian one. There is no significant change between the Image C and D in Figure 4.8 so the error is reduced slightly over the whole image. This should be caused mostly by the increase in image contrast, as the former used Oren-Nayar shader produces a more homogeneous brightness distribution on curved surfaces.

Without shadows the large error on the front of the block vanishes nearly completely. In the renderings before, the block casts a shadow on itself. This caused an additional incorrect flow edge which is now missing.

The normalized cross correlation (Figure 4.9) between the endpoint errors of the real and the realistically rendered scene can be used as an additional spatial similarity measure. Across all surfaces there are areas with low (negative) correlations indicating large dissimilarities in the local structure of the errors. We would expect that on flat surfaces the differences between the errors would be minimal. However there seems to be a significant difference in the local structure, possibly caused by a different dynamic range and texture effects.

Optical flow results are known to depend strongly on the used algorithm and parametrization. [2] contains additional evaluations with the flow algorithm by Black and Anandan [18]. As the results are qualitatively the same, we can conclude that the observations apply to the general problem of optical flow estimation and are most likely no artifacts of this particular algorithm/scene combination.

4.3.3 Conclusions

Several conclusions can be drawn from the presented results.

First, the renderings were all very similar and except for the missing shadows it is relatively hard to distinguish between the real and synthetic ones. Surprising is, that the flow differences were significant even if the optical difference between the scenes was marginal. Therefore it is not always possible to judge the realism of images based on a subjective impression. The notion that a rendering *looks good* is not necessarily sufficient to mark a scene as realistic in the sense that it will yield the same results in image processing. This may also be valid for the inverse case. Images that look *wrong* or *unreal* in some unspecific way may deliver totally acceptable results when used for benchmarking. Validity decision should therefore generally depend on measurable quantities.

Second, the most significant differences in the flow fields were caused by effects influenced by illumination and materials. Moving shadows, highlights and reflections represent violations of the brightness constancy constraint which have direct implications for the computed flow. These effects can also represent cases were the regular optical flow definition does not apply [140]. A synthetic OF benchmark can only be considered realistic if all these effects are simulated correctly. But also effects that caused no direct BCC violation had an influence. For example, there were small but measurable differences even for relatively homogeneous regions. The most probable cause are differences in the observable texture reflection behavior or interpolation errors despite the high mesh resolution.

As already mentioned, optical flow in general is rather sensitive to small image changes. Other methods, such as stereo estimation, which is often only pixel exact, produce more coarse results. The differences between real and synthetic scenes were mostly in the subpixel range so we can assume that stereo estimation would produce comparable results for both types of images. Similar arguments apply to other vision tasks.

Robustness is often at least as important as accuracy [48], especially in practical applications. As synthetic images offer more possibilities to challenge claimed robustness we can consider them a valid method for creating reference data. But before a final verdict can be given, more examinations, possibly on very different vision tasks, should be conducted. Additionally, a form of hybrid testing, using both real and synthetic scenes could help finding discrepancies between both methods.

4.4 Reference Data for 4D Light Fields

This section is based on results first published in [8]. It presents a concrete example for a hybrid dataset that contains both synthetic as well as real world reference data. The dataset is intended for the evaluation of existing and future algorithms for light field processing. The first part will give a short introduction in the theory of lightfields, then the creation of the image and reference data will be described.

4.4.1 Introduction to Lightfields

Geometric optics uses the concept of rays to describe how light propagates in a given volume. This omits effects described by wave optics such as diffraction. A ray can be parametrized by a point in space (x, y, z), a direction (σ, ϕ) and the amount of energy transported along it. Hence, how much energy is transported in a static volume can be characterized by the 5-dimensional plenoptic function $L(x, y, z, \sigma, \phi)$ [10]. We could also say that this function contains the total information about the illumination in a scene.

Regular cameras create a sparse two-dimensional sampling of this function, characterized by the internal and external camera parameters. The light field is integrated in a small region around each pixel position and for a solid angle defined by the camera optics. As such we could call it a 2D lightfield. The 4D lightfield, which is further motivated in [57] and [101], is also a subspace of the plenoptic function, typically defined on a surface such as a plane or sphere. In practical terms the difference between both representations is that for the 4D lightfield a single pixel can contain multiple incoming light directions.

It is possible to capture a 4D lightfield by recording regular 2D images at different camera positions on this surface. Practical realizations either use multiple cameras arranged in an array or move a single camera using e.g. a motorized frame. An alternative method is the use of plenoptic cameras that use microlens arrays on the sensor to simultaneously capture light coming from slightly different directions [158].

4D lightfields have numerous applications in image processing and computational photography. They can for example be used for very exact estimations of depth and surface normals or in inverse rendering. Examples and further information on lightfields can be found in [157] or [38].

So far a few light field datasets for evaluation and benchmarking exist, for example, in [160], the UCSD/MERL Light Field Repository[4] or the Synthetic Light Field Archive [5]. Problematic is that these not always contain reference data or that the baseline between individual views of the lightfield is too large for algorithms aimed at densely sampled lightfields. The new dataset described in this section consists of both synthetic as well as real world lightfields with depth and segmentation reference data and a much higher sampling density than previous datasets.

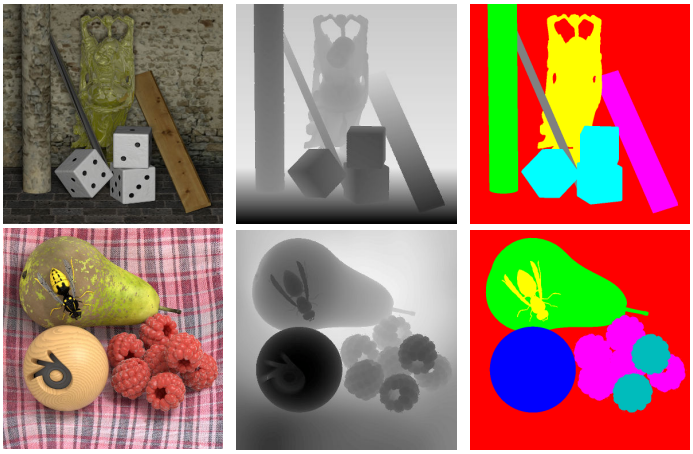
4.4.2 Real and Synthetic Light Field Data

The dataset contains a total of 13 densely sampled planar lightfields each consisting of 9-by-9 individual images. Seven of the dataset are synthetics ones, rendered with the open source software Blender [19]. The rest are real world sequences captured using a digital camera mounted on a motorized gantry. Creation and analysis of both types of lightfields as well as the synthetic reference data was performed by Sven Wanner. An example for the synthetic images can be seen in Figure 4.10, an example for the real ones in Figure 4.11.

Synthetic images are very well suited for benchmark datasets in lightfield analysis as it is necessary to create many images with slightly different view angles and positions. This kind of work can easily be automated while for real world sequences it is necessary to utilize special equipment. The renderings don't utilize full global illumination but rely instead on raytracing combined with the radiosity algorithm. Depth data was directly created from the scene description in Blender. As each visible object in the data was rendered from an individual mesh with a unique id, assigning a label for segmentation purposes was also straightforward.

Depth reference for the center view of the real world images was created using the method described in Section 2.2.1. The objects depicted in the scenes were first scanned using a structured light scanner and the relative orientation between mesh and camera was estimated using 2D-to-3D point correspondences. The depth map was then created by raycasting through each pixel of the image. Based on a reprojection error of the correspondences of 0.5 ± 0.1 pixels we estimate an error for the position of the object of $1mm$ along the cameras view axis. The relative depth error on the surface of the object is lower and in the range of the structured light scanners' accuracy. The main cause for these errors is the fact that the displayed objects don't have many distinct features that can be used for the correspondence estimation. Furthermore, the objects have a rather complex

[4] `http://vision.ucsd.edu/datasets/lfarchive/lfs.shtml`
[5] `http://web.media.mit.edu/~gordonw/SyntheticLightFields/index.php`



Center view Depth map Segmentation

Figure 4.10: Synthetic light field renderings.

Center view Depth map Depth validity mask

Figure 4.11: Real light field images

geometry which resulted in small errors and holes in the meshes. For these regions additional validity masks where created which mark incorrect sections.

4.4.3 Conclusions

We have seen that lightfield synthesis is another field were computer generated images offer great advantages. Real world devices are either expensive (in the case of camera arrays), slow (moving single camera) or sacrifice lateral resolution for angular resolution (plentoptic cameras). In synthetic setups we can simulate each individual method to decide, for example, how algorithms react to different view baselines or sampling densities.

Additionally, this dataset shows the practical limits of real-world reference data. Capturing the real lightfields themselves was time-consuming but straightforward and higher image resolutions and sampling depths could be produced relatively easily. The depth reference data on the other hand could not be made more accurately using the described method, even though high precision scanning methods were used. Creating depth data for all individual views of the lightfield data would also have taken significantly more time.

So this type of application represents a border case where sufficiently accurate reference data is quite hard to produce, especially in large amounts. The benefits of synthetic images are here clear but the impacts of using them has not been investigated in detail. The hybrid approach to dataset creation was therefore the best choice.

4.5 Creating Reference Data using Computer Games

So far we have investigated two possibilities to create synthetic reference data manually. Handling every aspect of the image creation process from object creation over scene composition to rendering gives unprecedented control over the resulting sequences. On the other hand, it limits the number of scenes that can be created in the same time, even considering parametrization and automation of certain parts. This leads back to the generalization and test case coverage problem mentioned earlier. The question is whether we can automate some or even all of the needed worksteps to create large amounts of data.

One interesting proposal is the use of a large amount of untrained workers to deal with some of the more tedious tasks of data creation. These crowdsourcing approaches were discussed by Aydemir et al.[11] or Donath and Kondermann [39]. One type of system that is used to produce image data in large amounts, however with a completely different intention, are computer games. Here the players control limited aspects of the *simulation* (such as the player character) to create new images. If it would be possible to get players to supply researchers with their regular game output, or to let them play modified versions of games with application relevant content, a potentially large source of reference datasets could be created.

There is no doubt that computer game graphics are in general of high quality. Many of the advancements in modern computer graphics were in fact made because of the gaming industries' demand for more natural and more impressive images for their products.

The wide span of available genres also suggests that the images created by games can be relevant for at least some applications. Producers of car racing games, for example, already work closely with car manufacturers to incorporate original CAD models and physically correct driving behavior to make the gaming experience more realistic.

The content of the games itself is often not publicly available, but many popular titles have active fan communities that create new content under much more permissive licenses. Newer game engines are also often build in a way that they can be scripted or modified by their user community so that additional reference data like vertex coordinates, motion etc. can be exported.

To identify possibilities as well as problems that can arise from the usage of computer games for reference data creation two game engines were investigated

more closely. [6] The first one is based on the game Crysis [36] by the German Crytek corporation[7] and the second is the Source engine by the US based Valve Corporation[8]. Both games are known to be relatively *modding friendly* as the producers have published tools for modifying content as well as graphic shaders or game logic.

Figure 4.12 shows free content for the Crysis game which, despite being already a few years old, displays impressive and realistic images. The game engine is also able to render stereo images and export depth maps in real time using the processing power of modern GPUs. A colorcoded depth map as well as a stereo estimation example using a semi-global matching approach can also be seen in the image.

The source engine was presented in 2004 and has fewer graphic capabilities, but offers more possibilities for modification as the large number of available third party products based on this engine shows.

For brevity's sake the conducted examinations were limited to the creation of depth and stereo data. While it was indeed possible to create reference data with these engines to a certain degree, many technical problems were identified.

Simultaneous rendering By programing custom shaders a user of those engines has control over what kind of data is rendered, for example, the regular scene, the depth buffer, object indices (for segmentation) etc. Both engines also have the option to render stereo images, either natively or by using a third-party plugin. But often the output is limited to a single render target. Due to this it is not possible to get all this data simultaneously. It is also not possible to render exactly the same sequence but with different types of output data repeatedly (See below). These limitations are sometimes hard constraints of the used graphics hardware and software libraries and sometimes deliberate restrictions by the engine developers. Unless the companies are willing to drop these restrictions the creation of reference data is therefore limited to specific single sets of data.

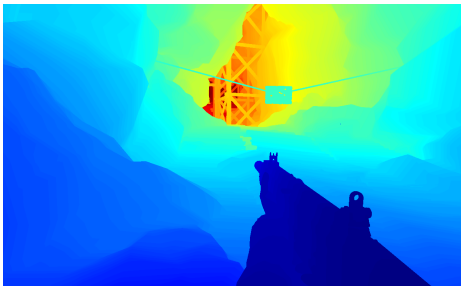
Depth buffer accuracy Nearly all computer games, not just the two investigated, use scanline render algorithms to achieve real-time performance. As such they often use the z-buffer algorithm by Catmull [29] for hidden surface removal. The z- or depth buffer value is approximated from the real distance between camera and corresponding object but the mapping is non-linear and

[6] This data was produced by Julien Stern and Maximillian Klingmann during a practical course at the HCI, Heidelberg.

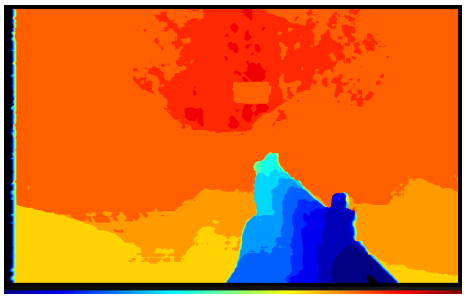
[7] `Crytek.com`,`cryengine.com`

[8] `source.valvesoftware.com`

(a) RGB Output:



(b) Reference depth from game engine



(c) Stereo estimate using Semi-global matching (Color encodings are not directly comparable)

Figure 4.12: Footage from the computer game Crysis with depth reference data. The game engine is capable of rendering realistic effects like motion blur, high dynamic range images and reflections.

in general non-reversible. Real depth values are often not computed by the engines as it is computationally expensive and not needed for rendering. The depth buffer values that are accessible are however a suboptimal supplement of real depth reference data. Additionally they may be subject to the already mentioned vertex interpolation problems. Some engines may support float precision depth maps but access to the data is not always possible using the provided library interfaces.

Randomness and repeatability Most games contain some random elements as part of their game mechanics. This clashes with the requirement that experiments should be repeatable under the same conditions. Game engines do generally not offer the possibility to control their internal state, such as the seed of the random number generator, etc. Due to the way input is handled, the resulting internal state (such as the player position and hence the camera position) can also depend slightly on factors such as CPU load or memory consumption. This is problematic if one wants, for example, to record a game situation from multiple camera angles. As there is often only one main camera observing the scene, this is only possible by repeating the exact same game sequence multiple times. If there is, however an uncontrollable random element each scene realization may be slightly (or even dramatically) different, even if the user inputs can be reproduced exactly.

The conclusion that we can draw from these results is that computer games and engines offer the technical prerequisites for reference data creation but a multitude of software restrictions make it currently unfeasible. It may be possible to get the cooperation of the developers of single games or to modify available open source games but the majority of existing games is generally unsuited for scientific purposes.

4.6 Results and Future Work

In this chapter we have identified the basic steps and problems of the creation of synthetic scenes with reference data. Once the composition of a scene is decided on, assets such as models, textures etc. must either be created or taken from other sources. Results from Section 4.3 suggest that material properties may have a significant impact on the resulting data. Section 5.5.2 of the next chapter strengthens that notion further. How sophisticated this data must be depends on the desired accuracy of the reference data.

Secondly, an appropriate rendering solution must be selected. The decision can be limited by the available raw data which may only be usable for a specific software. This choice is also influenced by the desired physical realism. Modern global illumination methods are necessary if the goal is photorealism and physical correctness. More specifics on Monte-Carlo based Global Illumination algorithms can be found in the next Chapter.

Computer graphic has the theoretical capacity to create large scale benchmark datasets, but practical aspects prevent researchers from doing so. A generic solution to create images of any desirable scene with sufficient realism and minimal experience on the side of the producer seems as unreachable as the generic optical flow algorithm that solves all problems. None the less, such data can still be created for limited application domains. Also, in situations where creating real benchmark data is not feasible it may be a realistic alternative.

Despite all efforts a purely synthetic dataset may still be considered unrealistic. Combining real and synthetic sequences in a hybrid approach may be a solution to get the advantages of both methods and may also help to bootstrap future developments in this direction.

5 Time-of-Flight Simulation

The following chapter demonstrates the problems and necessary considerations for the creation of synthetic ground truth in a specific case. The focus lies on the simulation of continuous-wave *Time-of-Flight*(*ToF*) depth cameras. These cameras use modulated infrared light sources to simultaneously measure the distance for each pixel of the sensor.

Unlike most depth imagers, such as stereo or structured light systems which measure depth using triangulation, ToF cameras measure the propagation time of a light signal to estimate the distance of objects from the camera.

Due to its unique properties, this type of image sensor has gotten much attention from researchers as well as industrial users. The capability to simultaneously capture depth as well as intensity images in the infrared range makes them useful in many fields of application, such as 3D reconstruction or human-computer-interfaces. They can also be used for creating reference data, for example using sensor fusion approaches as described in Section 2.2.2.

However, much alike other depth imaging system, these sensors show a multitude of systematic and statistical errors. These include high noise levels, depth and intensity dependent depth-errors, flying pixels or multipath interference. Development of algorithms dealing with denoising strategies, depth calibration or distortion reduction are therefore crucial for the continuous development of Time-of-Flight cameras.

Creation of reference data for these cameras can be split into two subsections. First, one can provide reference depth maps for real camera footage. This can be done using the techniques described in section 2.2. The reference depth maps that can be acquired this way are very useful for calibration purposes. The depth errors in are ToF camera a mostly non-linear in nature and multiple models to parametrize these errors have been developed. See the related work section (5.2) for details. The involved parameter spaces can have a high dimensionality and the results are in general not transferable between cameras even if they are of the same type. Hence, for error estimations and calibration the capture of multiple depth images is necessary. None the less, a few images may already be sufficient to estimate depth bias and to get more accurate results from subsequent images.

Second, on a more basic level, the raw output of the camera can be simulated using computer graphics to evaluate algorithms which operate on this raw data. Examples would be denoising algorithms or methods for motion artifact compensation.

This is an important aspect, as the different error sources present in the camera output cannot be separated easily. For a given depth image it is not clear if the deviation from the true ground truth depth was caused by intensity dependent variations, multipath effects in the scene or some other influence. A physical simulation with the possibility to change individual aspects of the scene (such as materials) and the simulation (e.g. electronic noise behavior) can help develop algorithms that deal with those effects. So far, there are multiple effects that have not been considered yet in existing ToF simulators. In the following, a special emphasis will be put on these effects, including multipath interference or ambiguous depth due to transparent materials.

The simulation of ToF cameras is a unique case, as methods from computer graphics can not be applied directly. Other depth imaging modalities for example can be simulated comparatively easy: E.g. for stereo imaging one would only need to render two images of the same scene with modified camera positions. In case of active stereo the structured light used to infer depth can as easily be imposed on a synthetic scene. ToF imaging on the other hand depends on principles computer graphics seldom has to deal with, namely the time dependent propagation of light. None the less, with minor modifications, classical global illumination algorithms can simulate such systems.

This chapter is structured as follows. Section 5.1 will summarize the theory behind Time-of-Flight cameras and list typical problems. Section 5.2 will then introduce related work and give a short overview over already existing camera simulators. Section 5.3 contains general considerations that are relevant to global illumination algorithms used in the following sections. Two different algorithms as well as examinations will be presented in Sections 5.4 and 5.5. Final conclusions and outlooks will follow in Section 5.6.

Figure 5.1: Left: PMD CamCube 2 (Image by PMD Technologies), Right: Panasonic EKL3104 (Image by Mrinnovative (CC-A-SA license))

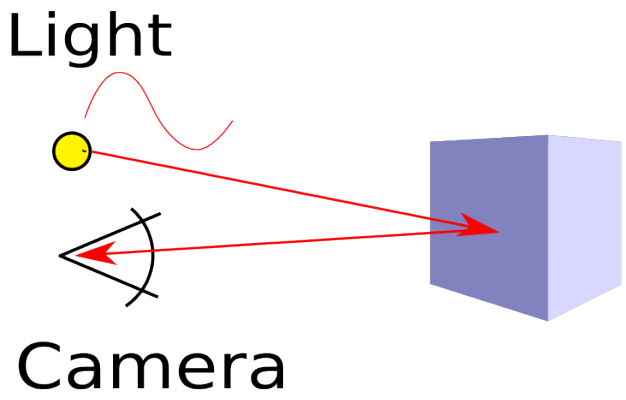


Figure 5.2: Time-of-Flight working principle. For small distances between emitter and detector the distance between object and detector is half the signal travel distance.

5.1 Time-of-Flight: Theory and Problems

The following section describes the working principle of a correlation based continuous-wave Time-of-Flight camera.

Distance measurement based on Time-of-Flight is well understood and has been in use for decades e.g. in the form of RADAR (radio detection and ranging), SONAR (sound navigation and ranging) or LiDAR (light detection and ranging) systems. All these systems are based on the same fundamentals: An emitter produces a electromagnetic (or sound) signal which is reflected from an object and finally measured by an appropriate detector. The distance d of the object can be determined by the travel time t and the propagation speed of the signal (c, speed of light for EM waves, speed of sound in the corresponding medium in case of echolocation).

$$d = \frac{t \cdot c}{2} \tag{5.1}$$

The factor two is necessary as the distance must be traveled twice by the signal.

The above mentioned systems can achieve spatial resolution of the distance field by mechanically or electronically (for example in phased array radar) sweeping a beam over an area. Sweeping the beam in two dimensions can generate a depth map and the angular resolution can only be increased by sacrificing temporal resolution (by sweeping the beam slower). Rotating terrestrial LiDAR Scanners

for example can achieve angular resolutions of 1 mrad for each axis over half a hemisphere but may need well over a minute to do so.

Time-of-Flight cameras in principle use one sensor, the image pixel, for each direction to achieve their spatial resolution and capture a whole depth field simultaneously, just like a regular CMOS or CCD sensor does for light intensity.

To do this, the scene must logically also be illuminated simultaneously. This introduces another problem to the concept. In pulse based systems, the depth resolution is directly proportional to the time resolution of the sensor. In the atmosphere, light travels the distance of 1 meter in approximately 3 nanoseconds. In this timeframe the detector must be able to collect enough light to distinguish the signal from the background intensity. Additionally, the pulse length must be in the same time domain, which makes it necessary to handle signals in the gigahertz range. This is possible but puts higher demands on the electronics and manufacturing processes. Additionally, if the camera should also capture an intensity image of the scene, the signal-to-noise ratio would be suboptimal as in this case exposure times are multiple magnitudes lower than in regular cameras. To alleviate these problems, continuous-wave Time-of-Flight cameras utilize a different principle to measure light propagation times.

Instead of measuring the signal delay directly using a pulsed signal, it is more sensible to use a periodically modulated signal of frequency f to illuminate the scene. The light returning to the sensor will have the same waveform and frequency but will be phase shifted relative to the modulation signal, that is unless multipath effects occur, in which case the waveform may change. The phase shift ϑ is then proportional to the light travel distance:

$$d = \frac{\vartheta c}{4\pi f} \tag{5.2}$$

This method has some interesting consequences. First, the achievable depth resolution is no longer dependent on the frequency of the signal. This makes it possible (and due to the following point even necessary) to modulate the signal with frequencies which are much more handable with standard electronics (≈ 20 Mhz in the case of most ToF cameras). On the other hand, for a distance d and the wavelength of the modulation signal d_m only the value $d \mod d_m$ can be determined as the possible phase values are periodic. This leads to the so called non-ambiguity range described below.

Given a known waveform and frequency, the phase of a signal can be determined. A practical realization to do so is the *photonic mixing device*.

In principle, the output of a sensor is the cross-correlation of it's own temporal sensitivity curve and the incoming signal. An ideal sensor would use a dirac pulse as its sensitivity curve, but of course such as sensor is not physically possible as the measurement process always involves the flow of energy of some form. The sensitivity curve of regular image sensors can be approximated by a box function whose width corresponds to the camera exposure time.

The photonic mixing device uses a steerable sensitivity curve to measure the phase of an image signal.

Given a scene, illuminated with a time varying light source, the incidence light intensity on a single sensor element coming from the scene can be formalized as:

$$I(t) = I_0 + A \cdot F(t) \tag{5.3}$$

Here I_0 is a constant offset induced by non-varying light sources, A is the amplitude of the signal caused by the modulated illumination and $F(t)$ is the time-modulation function of the signal.

Given a linear sensor response curve the output signal S can be described as:

$$S(t) = I'_0 + A' \cdot F(t) \tag{5.4}$$

where I'_0 and A' are the scaled intensity components. The assumption of a linear sensor does not hold in general and deviations from from it lead to some of the errors described in section 5.1.2, but it does not invalidate the basic principle.

Furthermore, we can use a sinusoidal modulation function F with frequency f, as this corresponds to the practical realization in most available ToF cameras. A general description based on arbitrary periodic signals can be found in [122].

$$S(t) = I'_0 + A' \cdot \sin(2\pi f + \varphi) \tag{5.5}$$

Here ϑ is the phase-shift as described in Equation (5.2), caused by the propagation delay. Generally, there is not one single delay value as the modulated light can take multiple paths in a scene, which arrive at the sensor at different times. The more general form would be

$$S(t) = I'_0 + \sum_i A'_i \cdot \sin(2\pi\nu + \varphi_i) \tag{5.6}$$

where i runs over all possible light paths. Although, we can continue with an effective phase shift ϑ_e, which is the weighted mean of all ϑ_i. The weight is here the corresponding amplitude A'_i, of which the one corresponding to the shortest possible distance usually dominates.

The photonic mixing device now acts like a loc-in amplifier, making it possible to maximize the output of a signal with a certain frequency. Additionally, the amplification factor can be phase shifted with a value of θ. The result is a time varying response curve for the sensor pixel which can be described with a reference signal function $R(t+\theta)$. The sensor output can then be described by the cross-correlation $C(t)$:

$$C(\theta) = \int S(t) \cdot R(t+\theta)dt \tag{5.7}$$

Using a Heaviside step function H with the same frequency as the light modulation signal we get

$$C(\theta) = \int_{t_0}^{t_1} I'_0 + A' \cdot \sin(2\pi f t + \varphi) \cdot H(\sin(2\pi\nu t + \theta))dt \tag{5.8}$$

$$= [t1 - t0] \left(\frac{I'_0}{2} + \frac{A'}{\pi} \cos(\varphi + \theta) \right) \tag{5.9}$$

Here $\Delta t = t1 - t0$ is the full integration time for the sensor. As it is only a linear factor it is possible to integrate over multiple periods of the reference signal which can reduce noise significantly.

Also there are three unknown values in this equation we are interested in. The constant intensity offset I'_0, the modulated amplitude A' and the phase φ. By sampling this function at three different phase-shifts θ, all these unknowns can be determined.

As the signal is disturbed by electronic and sensor noise, it is advisable to use more (N) samples at the positions θ_k.

The optimal solutions (in the least-squares sense) for I'_0, A' and φ as shown in [122], [164] and [125] are then:

$$I'_0 = \frac{2}{N} \sum_{k=1}^{N} \frac{C(\theta_k)}{\Delta t} \tag{5.10}$$

$$A' = \frac{2\pi}{N} \left| \sum_{k=1}^{N} \frac{C(\theta_k)}{\Delta t} e^{-i\theta_k} \right| \tag{5.11}$$

$$\varphi = arg \left[\sum_{k=1}^{N} \frac{C(\theta_k)}{\Delta t} e^{-i\theta_k} \right] \tag{5.12}$$

N should be kept as small as possible to maximize Δt for each sampling. $N = 4$ is the value used by most ToF cameras, with equidistant phase-shifts θ_k of $C_0 = C(0°), C_1 = C(90°), C_2 = (180°)$ and $C_3 = C(270°)$. We will call a full n by m pixel frame with one distinct phase-shift a raw- or phase-frame. Unless stated otherwise, the variables C_0 to C_3 will now denounce such a whole frame. The above equations can then be simplified to:

$$I'_0 = \frac{1}{2}(C_0 + C_1 + C_2 + C_3) \tag{5.13}$$

$$A' = \frac{\pi}{2} \sqrt{(C_2 - C_0)^2 + (C_3 - C_1)^2} \tag{5.14}$$

$$\vartheta = atan(\frac{C_3 - C_1}{C_2 - C_0}) \tag{5.15}$$

The effective depth can then be computed via

$$d \propto atan \left(\frac{C_3 - C_1}{C_0 - C_2} \right) \cdot \frac{c}{f} \tag{5.16}$$

In actual implementations, the atan2 function is used as it returns the phase angle without any discontinuities when the denominator in Equation (5.15) is non-positive.

There exist multiple variations to the above described correlation based ToF model. For example the modulation signal is not limited to a sinusoidal one. Multiple frequencies could be used, as shown by Conroy et al. ([33]), while Buttgen et al. even described a ToF camera using non-periodic signals ([27]).

This can effectively eliminate the below mentioned non-ambiguity range, however there exist currently no practical realisations of these principles.

5.1.1 Practical Implementation

All existing ToF cameras use at least four raw frames to compute depth, intensity and amplitude. Many of them can capture multiple raw frames with different phase shifts in parallel. This is accomplished by using so called multi-tap sensors, a sensor layout first described by Schwarte et al.[139]. Multi-tap sensors use two or more capacity wells for each pixel to accumulate charge carriers. The internal reference signal $R(t)$ can be used to steer in which capacity well the charge induced by a photon in the pixels light sensitive area will end. The PMDtech CamCube camera for example captures four pairs of two raw-frames each, simultaneously for the phase shifts 0°/180°, 90°/270°, 180°/0° and 270°/90°. This acquisition method has the benefit of decreased noise and can be used to reduce motion artifacts caused by movement of objects between adjacent raw-frames. See [99] for an example.

5.1.2 Error sources

There are multiple sources of errors which cause the result of equation (5.16) to differ from the true object-to-camera distance. Some of them are inherent problems of the measurement process, while some are caused by the electronics and noise. I will only give a short overview on these error sources and refer to the related work section (5.2) for details. A example scene with some of these effects can be seen in Figure 5.6.

Noise ToF cameras are subject to the same noise sources as regular cameras. These include poisson-distributed photon-noise, or shot noise, which is caused by the quantization of the charge carriers, as well as thermal and electronic noise in the integrated circuits. The former one is generally higher in ToF cameras as it is dependent on the number of charge carriers which is lower due to the shorter integration times. The second one is additionally temperature dependent and also higher as the complex electronics on the CMOS sensors have not yet reached the same sophistication as in other mass produced cameras. As the depth reconstruction formulas (Equation (5.15)) are also non-linear, a certain noise distribution can lead to significant depth variations.

Non-Ambiguity Range This is technically not an error but a limitation of the measurement process. Due to the signal being periodic, there is no way to distinguish between phase shifts of

$$\begin{aligned} & 0 < \vartheta < 2\pi \\ \text{or} \quad & n \cdot 2\pi < \vartheta < (n+1) \cdot 2\pi \end{aligned}$$

with $n \in \mathbb{N}$. A ToF camera based on this method can therefore only be used reliably for distances between zero and $\frac{c}{2f}$.

Flying Pixel So called *flying pixel* are the result of multiple depth cues at the same pixel. They most prominently occur at or around depth discontinuities where fore and background light signals overlap. Typically the measured resulting depth is somewhere in between the foreground and background depth unless the background depth is higher than the cameras non-ambiguity range. As the cameras sensor response curve is non-linear, other values are also possible. The results are single points that seem to float freely in space without being connected to any solid geometry. Additionally, the high camera noise causes the intermix ratio to vary wildly between consecutive frames to that the temporal depth noise around a flying pixel is even higher than on a regular surface. The mixture function that describes the measured depth with respect to the relative fore and background intensities is generally non-linear and two evaluation samples can be seen in Figure 5.3. This example assumes linear sensor response.

Non-Linearities Generally, it is assumed that the sensor response curve (the relation between incidence intensity and resulting digital signal) is linear, save for quantization effects. Experiments for different ToF cameras have shown that this is not always the case, especially for lower intensities. Deviations from a linear responsivity are also not easy to model as the true response curve is highly hardware dependent. A simple polynomial of higher order for example is not sufficient. This results in a non-linear intensity dependent error. However, as shown by Schmidt [136], deliberately using sensors with non-linear response curves can also be used to increase the dynamic range of the cameras without sacrificing depth accuracy.

Temperature dependent error Both the sensor electronics as well as the modulated light source will change multiple of their characteristics with changing temperature. This is due to both external sources as well as self-heating. Between the time of switching a ToF camera on and until it has reached a steady state the measured depth values may vary strongly.

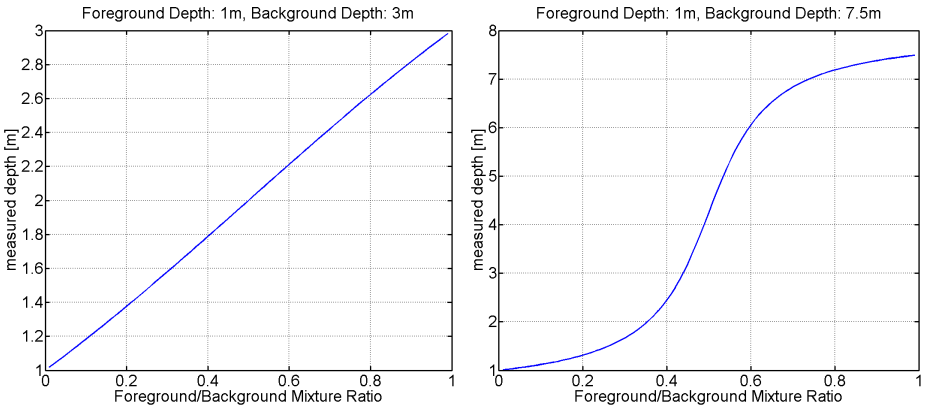


Figure 5.3: Which depth would a ToF camera report if two depth cues with different intensity were present in a pixel. A mixture ratio of zero means that the first cue has brigthness 1 while the second one has 0. The foreground depth in both plots is 1 m while the background depth is 3m in the left and 7.5m in the right image.

Wiggling Error The light modulation signal is not perfectly sinusoidal, but may contain higher harmonics. The correct waveform of the signal depends on the number of harmonics and their phase shifts. An exemplary LED signal of a CamCube 3 sensor can be seen in Figure 5.4. This results in a periodic depth deviation which is itself dependent on the true depth. See the works by Rapp [125], Erz and Jähne [42] or Schmidt [136] for specific examinations of different cameras.

Multipath The so called *multipath error* is also caused by multiple depth cues, but unlike flying pixels, it is more systematic than statistical in nature. Light does not only travel a simple path from the light source to the scene and back to the camera. This one possibility is usually the single most strong contribution for the irradiance of an observed point, but all other possible light path also do contribute. The observed depth in a point is therefore typically higher than the direct distance to the camera and also dependent on scene geometry and materials. In a computer graphics context such effects could be summarized as global illumination. The following chapters deal with the simulation of multipath effects in greater detail.

Motion artefacts ToF cameras capture multiple frames in fast succession to evaluate the phase of the modulated light signal. The result is well defined for a static scene. However, if objects in the scene move between acquisitions of the raw frames, depth errors can occur, especially at motion boundaries.

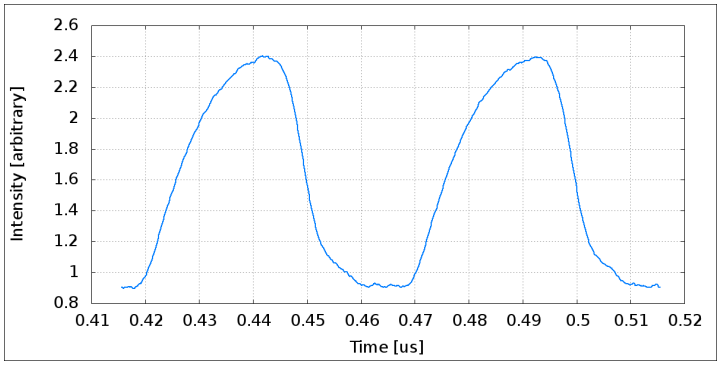


Figure 5.4: LED signal of a CamCube 3. Clearly visible are the deviations from a sinusoidal signal. (Image courtesy of Henrik Schäfer)

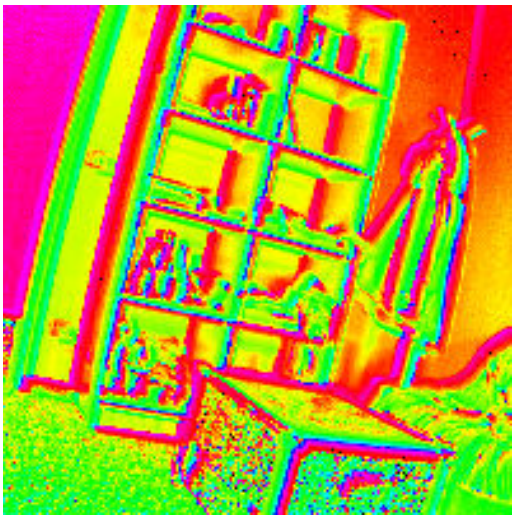


Figure 5.5: Motion artifacts caused by a moving camera. (Image courtesy of Jens-Malte Gottried.)

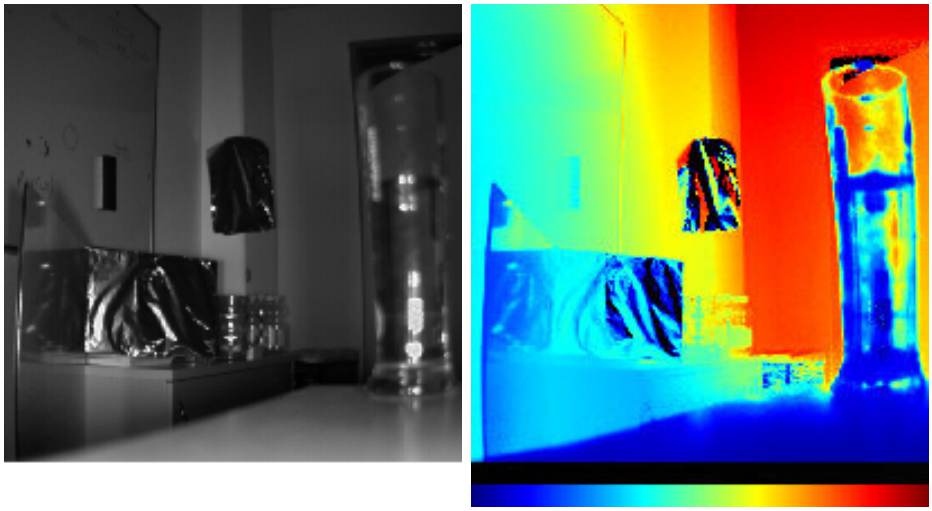


Figure 5.6: **Left:** intensity image of real example scene with problematic content. **Right:** Depth map of same scene. Depth estimations on reflecting (aluminum foil) or transparent (water glass) surfaces differ wildly.

5.2 Related Work

Literature on Time of Flight classically concentrates on characterization of typical problems, while recent works deal with methods to handle or rectify these problems. A overview over current topics regarding ground truth in time-of-flight imaging can be found in [7].

As the number of different camera and sensor manufacturers is limited, most works concentrate on one, maybe two cameras and their characteristics. The PMDtec CamCube is mentioned most often and could be considered as the typical ToF camera. This may change however when more cameras such as the Microsoft Kinect 2 [131] enter the market.

Publications that deal with ToF imaging itself and not solely with applications thereof usually fall in one or multiple of the following categories. First *Analysis*, describing the characteristics of the hardware, potentially under special conditions. This is naturally also a prerequisite to the following two, namely *Rectification*, the correction of problems inherent in the sensor and finally *Simulation*, the creation of artificial ToF data. I will list some of the most important works falling in either one of those categories.

The characteristic errors of Time-of-Flight camera haven been investigated in succession by Plaue [122], Rapp [125], Schmidt [135] and Schmidt [137, 136]. Their works characterize important error sources in Time-of-Flight imagers such as non-linear sensor response curves, non-sinusoidal illumination and different electronic noise sources. Additionally, Erz and Jähne [42] performed radiometric and spectrometric measurements and calibrations on ToF cameras.

Based on these characterizations, multiple works deal with the reduction or removal of those errors. These include calibration procedures to reduce the intensity based depth errors presented by Lindner and Kolb [104] or similar works by Falie and Buzuloiu [45].

Techniques developed for image denoising can also be applied to ToF imagers. Either directly to the resulting depth maps or even earlier in the processing pipeline by denoising the raw phase images. As depth edges are usually quite distinct, variational methods can be used to regularize the depth maps. This has been shown by Lentzen at al. [100], who use first and second order total variation approaches. Aodha et al. [107] use machine learning, especially random forests, to learn and reduce the noise behavior. Additionally, Schoner et al. [138] try to identify regions with consistent depth by means of clustering.

The estimation and compensation of multipath interference has been the focus of recent investigations. Falie and Buzuloiu [45] suggest a method based on structured light to use additional triangulation data similar to stereo systems. Fuchs [51] estimates the secondary light arriving indirectly at a point in the scene using the ToF measurements themselves. Jiménez et al [83] use a similar but iterative approach, minimizing a cost function to reduce the depth error. Their model does also make less strong assumptions about the scene radiometry than the former ones. Finally Dorrington et al. [40] repeat the raw measurement with different modulation frequencies to identify the primary light path. Most of these methods make the simplification of treating all materials in a scene as lambertian reflectors, an assumption that not always works, as pure lambertian reflectors are actually quite uncommon [17].

Scattering inside the camera casing and lens is somewhat related to multipath problems and has been investigated by Karel et al. [85]. They propose a method to iteratively measure and correct the point spread function of the system.

The reduction of motion induced artifacts has been investigated e.g. by Lefloch [99], Hussmann [76] or Lee [98]. Some of these approaches use optical flow techniques to estimate the motion between raw phase frames and warp the images back to the first one.

Simulation of ToF cameras has so far been performed by Schmidt [136] and Keller, Kolb et al. [87] [86]. The former presented a sophisticated physical sensor model based on empiric calibration data from a PMDTec CamCube. It is capable of simulating many of the sensor related errors such as electronics noise and intensity dependent depth. Although, to do so it needs a precomputed true depth map as well as reflectance values for every pixel as input. This simulation is later combined with the method described in this work.

The second simulator uses the render pipeline of modern GPUs to simulated a Time-of-Flight camera in realtime. Here a 3D scene representation based on common data formats can be used as input, which allows for fast setup of different simulation scenarios.

Simulation of ToF Data with computer graphics is unique in that time dependency is an integral part of the problem. Most render system assume a steady-state of light propagation. In computer graphics Smith et al.[143] were the first to postulate a generic time-dependent form of the rendering equation.

The usage of Monte Carlo methods to simulate multiple light scatterings is not new, as it has already been employed for the evaluation of LiDAR systems. Examples are the works of Gordon [56], describing the effect of laser reflections

on oceanic surfaces or the works by Kunkel et al. [96] who investigated multiple atmospheric scatterings.

5.3 Simulation Considerations

While other simulators for Time-of-Flight sensors do exist, they lack the ability to simulate multipath effects, which is one of the major error sources in existing depth cameras. In computer graphics, methods that simulate the propagation of light, even after multiple bounces from different surfaces or volumes, are commonly known as *global illumination* methods. As this is exactly the problem we need to solve, it seems optimal to use an existing global illumination algorithm and modify it to simulate Time-of-Flight. The fact that these algorithm are based on physical principles in contrast to e.g. scanline renderers makes it also possible to add other effects such as those caused by varying illumination, refraction, etc. An excellent overview on the theory of physically correct rendering can be found in [120].

Two different global illumination methods were considered for ToF simulation. Photonmapping and Bidirectional Path Tracing. An in-depth description of both algorithms as well as experiments and results can be found in Sections 5.4 and 5.5.

In this section I will discuss general properties and design considerations of image synthesis and render solutions. Unless noted otherwise they apply to both investigated algorithms. The actual modifications made to both algorithms that make ToF simulation possible are quite similar, but as they can be better understood in the context of the actual algorithm description they are mentioned later on.

Many render solutions and global illumination algorithms have common characteristics or parameters. Those that are important for Time-of-Flight simulation will be discussed in detail:

5.3.1 Render Equation and Render Bias

One of the fundamental problems of computer graphics is to find solutions or solvers for the *render equation*. In basic, it describes the radiant emittance, or how much light is leaving a point of a surface in a certain direction (e.g. towards a camera) given the incidence illumination. Nearly all known render algorithms can be derived from this equation. It is fundamentally based on the conservation

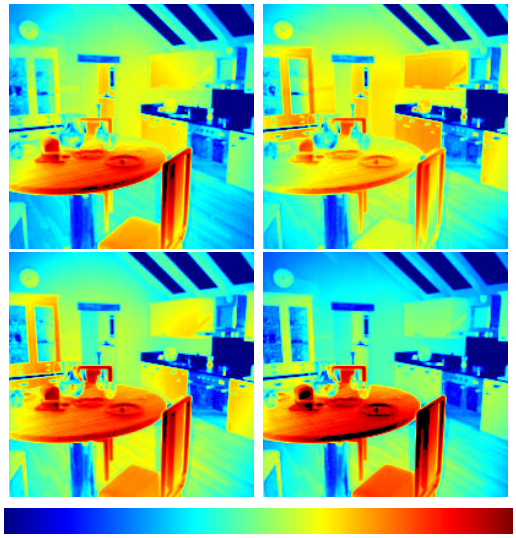


Figure 5.7: Colorcoded raw phase images of synthetic **Kitchen** scene [30]. Well visible are the intensity changes at different depths. Reflecting surfaces such as the chrome on the oven door appear very dark as they reflect only indirectly illuminated surfaces.

of energy and was formally introduced by Kajiya [84] and Immel et al. [77]. It takes the form of a integral equation, where the radiance at a point is the sum of the self-emission and a function of the irradiance:

$$I_o(\vec{x}, \omega_o) = I_e(\vec{x}, \omega_o) + \int_\Omega I_i(\vec{x}, \omega_o) \cdot b(\vec{x}, \omega_o, \omega_i) \cdot (w_i \cdot \vec{n}) d\omega_i \tag{5.17}$$

with

- I_o radiance
- $\vec{x}$ location
- ω_o outgoing direction
- I_e self emission of the surface
- Ω unit hemisphere above $\vec{x}$
- $b(\vec{x}, \omega_o, \omega_i)$ bidirectional brightness distribution function (BRDF)
- I_i irradiance
- $w_i \cdot \vec{n}$ attenuation factor due to the projection of the infinitesimal area segment

In this simple form the equation is defined on a 4-dimensional domain (on a surface with two angles for w_o). Generally, the function is additionally dependent on time and the lights wavelength λ. The *bidirectional brightness distribution function* (BRDF) b which plays a major role in the modeling of materials and surfaces can even be 10-dimensional if sub-surface scattering and luminescence are taken into consideration. However, most often models with only a few parameters based on the physical material that must be simulated are used. Examples are the lambertian, Oren-Nayar[111] or Cook-Torrance[34] models mentioned later on.

Generally, there exists no analytic solution of the render equation, so practically all render algorithms try to solve it numerically using different assumptions or simplifications. Wide spread is the use of Monte-Carlo integrators which sample the function for different points and values for incoming (ω_i) and outgoing (ω_o) angles. The two methods used in the following sections also fall under this category.

In this context the term of *render bias* is important. If modeled correctly, the numerical methods should converge on a physically correct solution. However, not all render algorithms aim at physically correctness, as artistically pleasing or at least convincing results are also often sought for. This is called render bias, as the algorithm will not converge to the correct solution even if run indefinitely. Algorithms can either be biased by design or by choosing the parameters incorrectly. The later used photonmapping algorithm for example is biased unless an infinite amount of photons is used. The influence of this is however minor and the bias can be kept small if the number of photons is sufficient. The second algorithm, path tracing, is generally unbiased unless the recursion depth is set too small.

5.3.2 Sampling and Image Filtering

Image processing, image synthesis and camera simulators all use pixels as the fundamental unit of data, be it input or output. This is somewhat problematic as the definitions of a pixel vary slightly between these fields.

As already noted by Smith [144], the most simple interpretation of a pixel as a colored square is very seldom the correct one. One of the few cases where this is correct, is when one speaks of a single display element of a TFT LCD. Here the elements are in fact square formed and lit homogeneously.

From a signal processing standpoint, a pixel is often described as a single sample of a continuous light signal on the sensor plane. To perfectly reconstruct this

signal from the samples, the *Nyquist-Shannon sampling theorem* ([80], Theorem 2.1) must be fulfilled, stating that the sampling frequency must be 2 times higher than the largest frequency present in the signal. For real world images this continuous signal is generally not bandlimited and even for synthetic scenes it is at least scale-dependent, so there is no practical rule for how dense a sampling must be. In this representation the pixel has no spatial profile which is useful from a mathematical standpoint.

Although, this does not take physical measurements into considerations. A CCD or CMOS sensor does not sample the light signal at a single point but rather over a light sensitive area. As the spatial information over this area is lost during the measurement, the characterization of a pixel as a colored square is actually not that wrong in this case. The true form of the sensitive area of course depends on the physical layout of the semiconductor and additional optical elements like microlenses that can change the effective area further. Electronic effects like crosstalk or blooming in CCDs could also be modeled into this representation, as well as quantization effects due to the conversion into a discrete signal [80].

The image signal I_{res} a real world line sensor would produce, can therefore be described as multiple convolutions of the original light signal:

$$I_{res}(x) = I(x) * P(x) * S(x) * III_T(x) \tag{5.18}$$

with

- $I_{res}(x)$: measured signal at discrete points
- $I(x)$: original continuous signal
- $P(x)$: point spread function (PSF) of camera optics
- $S(x)$: active sensor area
- $III_T(x) = \sum_k \delta(x - kT)$: dirac comb, where k sums over all pixels of the sensor with a pixel spacing of T

defined on an image domain $x \in \mathbb{R}$. Two-dimensional sensors are defined analogous on an image domain $\vec{x} \in \mathbb{R} \times \mathbb{R}$.

Image synthesis and computer graphics are either focused on simulating the original light signal that originates from a physical scene description or on the reconstruction of a visually pleasing image based on this description. Sensor simulation as it is performed here should instead focus on the physically accurate reproduction of the sensor signal. The signal convolution and sampling implicitly performed by a CCD or CMOS pixel should therefore be part of the following considerations.

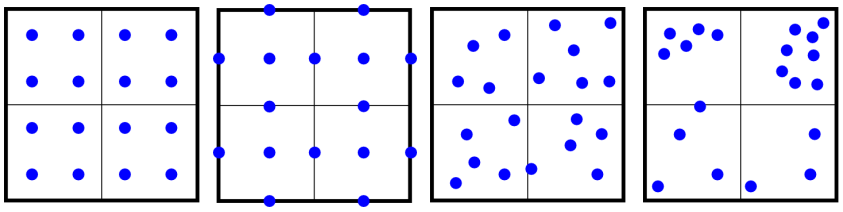


Figure 5.8: Possible distributions for *pixel samplings*: Regular spacing (can lead to aliasing), samples shared between adjacent pixels, roughly uniform random distribution, random distribution with clustering (these can be chosen deliberately or due to a bad random number generator)

Scanline renderers usually evaluate the color of a pixel only once at its center or use a very limited number of distributed samples for spatial antialiasing. Here the sample space is limited by the size of geometric structures projected onto the pixel (e.g. how much detail is contained in the pixel), although the render pipeline makes no assumption about the size of these structures. Examples of multiple sample patterns which are in general use can be seen in Figure 5.8. If the Nyquist-Shannon sampling theorem is not met, meaning if the spacing of samples is larger than the structures in the scene, aliasing artifacts occur. Non-uniform random sampling has the advantage that it converts these artifacts into noise ([120], p.334).

When the global illumination is computed, the signal space that must be sampled per pixel increases. This is the case, as we must now consider the multiple paths the light could have taken to reach the geometry visible in that pixel. Insufficient sampling manifests itself as noise and *aliasing* artifacts in the image. So, to get a clean representation the number of samples per pixel needs to be much higher than in case of scanline rendering. Probabilistic sampling methods which are aware of the current noise level or the scene structure are often employed to concentrate sampling in more important areas of the image. Examples would be Metropolis Light Transport by Veach [156] or Low Discrepancy Sampling by Mitchell [112]. Which sampling is best is often scene dependent.

When simulating a Time-of-Flight sensor, the sample space gets even larger as now the signal must also be sampled in time. Although, in the described ToF simulation algorithms this increase is somewhat alleviated as no full time-domain simulation is performed. Instead, the time sampling is replaced with a sampling over the possible distances of varying light paths. See the algorithm description under Section 5.4.1 for details.

The final pixel color is computed using an appropriate spatial filter function such

as sinc, gaussian or box filters. Their use is motivated from signal processing theory as a means to reconstruct a continuous signal from a sparse sampling. Although as the underlying continuous signal is not bandlimited, none of the available filters can be considered ideal. The effects of different filters are more prominent for low numbers of samples.

However, from a physical simulation standpoint these filters are not necessarily optimal. While from a mathematical or signal processing standpoint the sinc filter would be optimal, it also has the problem of having an infinite support, which can not be realized in actual implementations. Filter with negative lobes, such as a cutoff sinc or the often used Mitchell-Netravali filter [113] can introduce visible ringing artifacts at object boundaries .

Ideal would be a filtering with the point-spread-function of the optical system combined with the area response of the physical sensor pixel. The former can be represented by an *Airy-Disc* while the latter is probably best described by a box function. An Airy-Disc describes the diffraction pattern caused by a circular aperture which follows this form:

$$I(x) = I_0 \left(\frac{2J_1(x)}{x} \right)^2$$

with J_1 as the Bessel functions of the first kind. Generally render frameworks don't offer such a sophisticated filtering approach and in this special case it is recommended to perform simple averaging (box filter) or maybe gauss filtering on an increased number of samples as this represents the physical sensor response best.

Sampling and filtering would also influence the perceived depth of a ToF camera as well as the occurrence of flying pixels. Given two depth cues at a single pixel position, the resulting depth would lie somewhere in between the two values, depending on their relative intensities. The used sampling strategy must make sure that the distribution of accumulated depth cues matches the true distribution, according to multiple possible light paths, while depth cues from neighboring pixels due to spatial filtering may also influence the depth. The results of mixed depth cues can be seen in Figure 5.3.

5.4 Algorithm Investigation: Photonmapping

Two different global illumination methods were considered for ToF simulation, Photonmapping and Bidirectional Path Tracing. Both algorithms are used extensively for artistic and media renderings as well as for scientific visualization. Path tracing is also used in physics for example for simulating optical paths in lens system or optical fibers.

We investigate photonmapping first as it is known to have a better predictable noise behavior and allows the precomputation of the light situation in a scene. The idea is to compute the light distribution only once for four individual raw frames to save render time. While this was indeed useful for the simulation, certain properties of the algorithm proved to be difficult to manage. The most problematic part is the fact that the actual photon shooting process which approximates the light distribution in a scene has only little influence on the measured depth. The following section will explain the working principles of the photon mapping algorithm, lay out some experiments conducted with it and finally explain why the algorithm is only partially suitable for ToF simulation.

5.4.1 Method Description

Photon mapping was first introduced by Jensen [82] as a method to solve the *rendering equation* (5.17). It separates the computation of the light distribution in scene from the actual view point rendering. The working principle is visualized in Figure 5.10.

The problem is solved in two steps. First, the radiance is calculated by emitting virtual particles (photons) from each light source into the scene. Each time the photon intersects the scene geometry, its position is recorded in an appropriate data structure (here a kd-tree) and from there it can be further reflected or transmitted according to the local material properties. This is done until a sufficient number of photons are deposited in the scene. In a second step, view rays are projected from the camera similar to classical raytracing. Around each scene-ray-intersection the radiance is calculated by accumulating photons in the vicinity. If the ray hits a reflective or transparent surface it can be traced further for simulating mirror like reflections.

The following detailed algorithm description is based on the Exphotonmap Integrator, a general implementation of the photon mapping algorithm by Pharr and Humphrey [120] for the PBRT software library. The actually used implementation is of the Luxrender project [60], a fork of PBRT under the GNU General Public

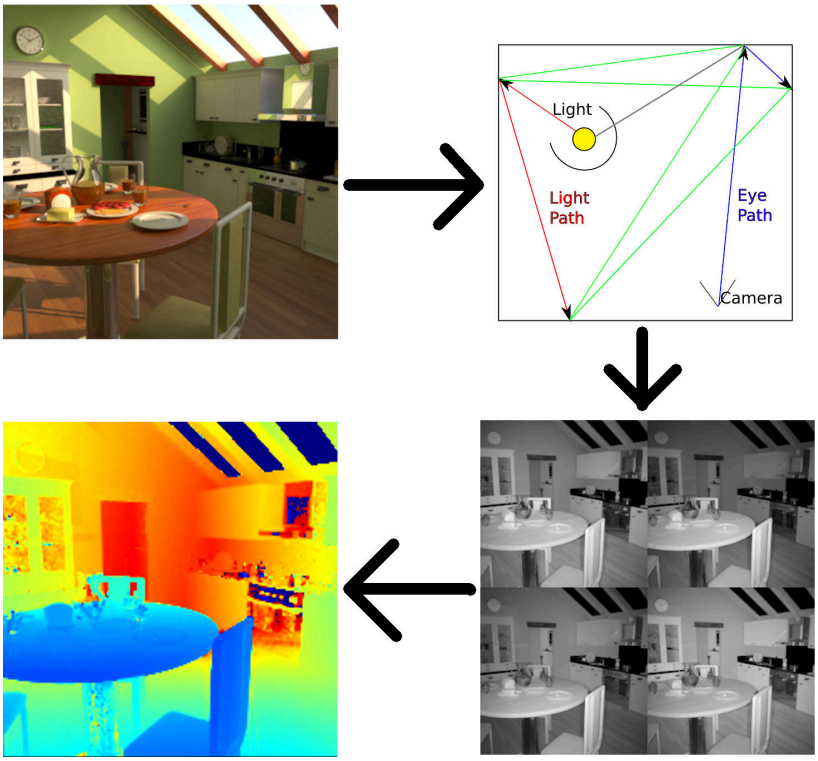


Figure 5.9: Processing Pipeline of the algorithm: Starting from a scene description global illumination is used to generate four raw phase frames, similar to a real ToF camera. The scene depth can then be recomputed from these phase images. The algorithm simulates effects based on different materials and multipath artifacts.

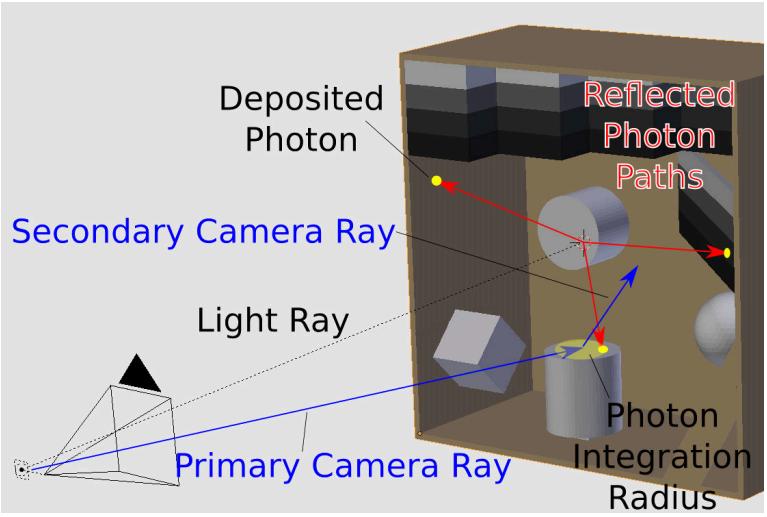


Figure 5.10: Visualization of photon mapping algorithm. Red paths are photon paths(1. step of algorithm), blue paths are view rays(2. step).

Licence [49]. The later described ToF Simulation is implemented as a modification of the luxrender code.

Detailed description The photon shooting process is implemented as follows. A light in the scene is sampled and a virtual photon is emitted, according to the light properties. If the ray describing the photon direction intersects a part of the scene geometry, its position can be recorded in a kd-tree data structure. It is also possible to use other structures, e.g. a local texture map, but the kd-tree has the advantage that it is independent of the actual geometric description (Texture maps for example would not work well for geometric primitives or volumes). This first kd-tree is also called the *direct photonmap*, as it only records photon events of the first order. The direct photon map is only used if the algorithm uses *final gathering*, a additional integration step which can speed up indirect illumination calculation later on (Note that we don't use final gathering due to implementation specific problems). Now a secondary attenuated photon is emitted from the intersection point. For transparent or mirror like surfaces the emission direction can be determined directly from Snell's law. For all other cases, such as diffuse reflection, the local BRDF must be sampled. For this either a measured or a modeled BRDF (shader) can be used. Photons resulting from direct reflections can also be called caustic photons, the ones originating from diffuse reflection are indirect photons.

Depending on the reflection type, these photons are now saved in either the *caustic photon map* or the *indirect photon map* when they intersect the scene geometry again. This commences until either, a given maximum photon depth is reached or a probabilistic termination criterion is met. Once a photon path terminates, a new photon is generated from another light source. The final photon maps now represent the distribution of light in the scene and will be evaluated in the second phase of the algorithm. A sparse rendering of an indirect photon map can be seen in Figure 5.11. Keep in mind that caustic photon map is only used to calculate the position and form of caustics due to light concentration and not for mirror projections. It usually contains less photons than the indirect map unless glass or mirror materials dominate the scene.

The second step acts much like a classical raytracer. A camera pixel is sampled and a view ray based on the lens and camera settings is traced until it hits the scene. The final color of this intersection point, as well as the originating pixel is determined by all light contributions in that point:

Direct illumination The irradiance from all lights in the scene is accumulated unless the direct line of sight between the source and the intersection point is blocked. This part can be computed directly as it only depends on the incidence angles.

Specular illumination If the hit surface is composed of a specular reflecting or transparent material, the reflection can be computed by emitting a secondary ray according to Snell's law. For a non perfect but still specular reflector the secondary ray can be sampled according to the local BRDF. The color of this secondary ray can then be computed recursively in the same way. The recursion depth is typically limited.

Indirect illumination The photons in a fixed radius around the intersection point are gathered from the direct and caustic photon maps. If *final gathering* is used, secondary rays are emitted and similar lookups are performed on the direct photon map to get a better estimate of the indirect illumination. (As the combination of final gathering and Time-of-Flight simulation has proven to be too error prone, we will omit this step.)

Special cases These can occur if the ray hits an area light source directly or if no intersection at all occurs. In the later case, the intensity can simply be set to an ambient light value or left at zero.

Usually, the described process is repeated multiple times per pixel. This is necessary as otherwise only a to small subset of all possible light paths would be considered. The final intensity of pixel is computed from individual pixel samples by means of spatial filtering, e.g. with a gauss or sinc filter as described in Section

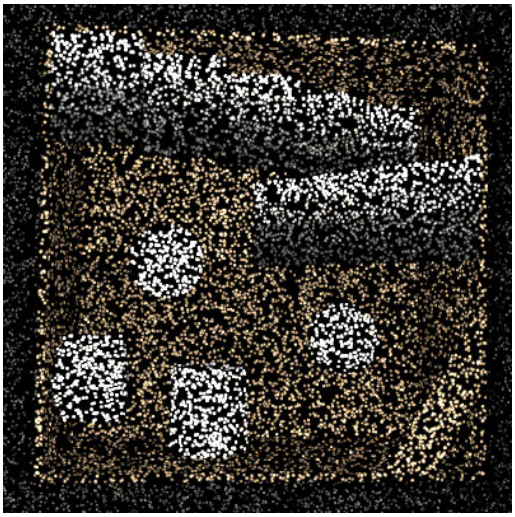


Figure 5.11: Sparse photon map of box scene. Each point is a photon event (intersection). Their density represents the radiance in that point. For visualization the number of photons has been greatly reduced.

5.3.2. The precomputed photon maps can be reused for consecutive renderings as long as only the camera view point but not the scene itself does change.

Phase Modulated Photon Mapping To simulate the time dependent illumination of a Time-of-Flight recording, the algorithm was modified in the following way: Usually, for each intersection or photon event only the intensity (possibly for different wavelengths or colors) and position of the event is saved in the photon map. Additionally we now save the full travel distance since the emission from the light source in the map. The resulting photon maps represent the radiance in the scene (through photon density and individual energy per photon) as well as a light travel distance profile.

In the second step, the individual light contributions for each pixel are accumulated to a final intensity L_r: For direct and specular illumination the lengths of the camera-scene-light paths can be computed directly by adding up the length of the individual view rays. The length of specular paths which contain multiple intersections can be added recursively. Indirect diffuse lighting as well as caustics are handled by integrating photons in an area around each intersection point. As now each photon also carries its travel distance, the integration can be modified to attenuate each light contribution according to this distance.

The final intensity L_r observed in a pixel is then:

$$L_r = \sum_i L_i \cdot (m_i + O_m) \tag{5.19}$$

where the sum is performed over all direct and indirect contributions and the modulation factor m_i for each contribution L_i defined as:

$$m_i = \left(0.5 \cos\left(\frac{2\ d\ f\ \pi}{c} + \theta_k\right) + 0.5\right) * I_m \tag{5.20}$$

- L unmodulated Intensity
- d distance
- f modulation frequency
- c speed of light
- θ_k phase shift between modulation and sensor signal
- $I_m = [0..1]$ modulation Intensity
- $O_m = [0..1]$ modulation Offset with $O_m + I_m = 1$

The light sources of a real ToF camera typically have a constant intensity offset and only a part of the light is intensity modulated. The modulation factor is modeled accordingly by introducing I_m and O_m.

Repeating this process for the whole image and the four different values for θ_k yields the four phase images C_0, C_1, C_2, C_3. Given these images, the relative phase ϑ, amplitude A' and intensity I'_0 can be reconstructed according to equation (5.15) with.

$$\vartheta = \text{atan2}\,\frac{C_3 - C_1}{C_0 - C_2} \tag{5.21}$$

which results in a phase value between $-\pi$ and π. After shifting the phase to the positive range

$$\vartheta_+ = (\vartheta + 2\pi) \bmod 2\pi \tag{5.22}$$

the final depth can be computed as:

$$d = \frac{\vartheta_+ \cdot c}{\pi \cdot f} \tag{5.23}$$

5.4.2 Simulation Considerations

Generic render settings as described in Section 5.3 also apply here, unless noted otherwise. The most important parameter of the algorithm is the number of photons to integrate over in the vicinity of each view-ray intersection point.

For artistic rendering, a few dozen photons in a rather large area are sufficient for optically convincing render results.

For Time-of-Flight simulation, one must perform a statistically sufficient sampling, not only over the light contributions at each point, but also over the various photon paths. As the algorithm is split into two parts, sufficient sampling must be conducted for both parts individually.

The Photon Mapping itself is biased in that it will not necessarily converge on a correct solution for the render equation. This is caused by the first part of the algorithm, as the number of photons is fixed and once the light distribution in the scene is computed via the photon maps, it won't change anymore. A bad initial sampling of the map can therefore influence the results negatively. A sufficiently high number of photons can alleviate this problem, especially when considering that the sample space for Time-of-Flight simulations is increased in comparison to pure artistic renderings. Progressive photon mapping [91] could be used the eliminate this problems in future implementations.

Additionally, the photon integration area must be kept small, as otherwise the distance between the intersection point and the contributing photon could introduce an error in the traveled light distance. The area around the intersection point in which photons are accumulated is usually enlarged dynamically by the algorithm until enough photons are collected. If the photon map is sparse, this distance can reach values of a few centimeters. Both these facts make it necessary to compute sufficiently dense photon maps where the number of photons should be at least one order of magnitude higher than in the default parametrization (where 1 million photons are usually enough even for complex scenes).

Furthermore, considerations apply for the used materials and material properties. Most ToF cameras use infrared light to infer depth. However, the used wavelength is typically in the range of $850nm$, which is rather close to the visible spectrum. Here we assume that the models developed for visible light do also apply under the near-infrared illumination of the ToF camera, even if the individual model parameters may change. We could observe that for example white paper with black ink has a reversed intensity response under infrared light (the paper appears darker than the ink), however the general reflection behavior (specular or diffuse) did not change drastically. Another example is glass that can be transparent in

the visible spectrum but opaque in the far infrared. So far all transparent surfaces that we have observed (with the exception of water) were still transparent when observed with a ToF camera.

Render time for a single phase image depends heavily on the used material shaders and scene geometry. The simulation of a single 200 by 200 pixel depth image at 20000 samples per pixel with the yet unoptimized program can take about two hours on a 2.4 Ghz Xeon processor. The generated photon maps can be saved between different phase renderings which reduces computation time for all images but the fist. The advantage of individual renderings is that motion blur can be simulated accurately by animating camera or scene motions for each phase image.

5.4.3 Experiments

Multiple synthetic, as well as real scene setups, were used to evaluate the algorithm. Two synthetic scenes are laid out as a proof of concept with very simple geometries. They are mostly used to investigate the influence of algorithm parameters, such as recursion depth or number of samples.

The first synthetic scene, labeled ***Shell***, is modeled as a half-shell centered around the virtual camera. In practice, the ground truth distance to the camera center is the same for every pixel. The shells surface is modeled with a pure lambertian material, as the effects of specular highlights etc. are not subject of the basic evaluations. This scene is also used later in the bidirectional path tracing experiments to test effects such as the non-ambiguity or wiggling error. A screenshot of the scene can be seen in Figure 5.12.

The second synthetic scene is labeled **Corner**. The displayed object consists of two walls with a variable opening angle (displayed in Figure 5.27 with a 90° angle). All surfaces are modeled with the same specular reflecting material, only the brightness of the materials diffuse channel does vary in vertical direction. In Luxrender, this shader (called *glossy*) is implemented as a diffuse base material combined with a specular coating. The diffuse material is modeled as a lambertian or Oren-Nayar [111] shader while the coating uses a Cook-Torrance reflectance model [34]. Details on the exact implementation of the material can be found in the LuxRender documentation.

The scene **RealCorner** is similar to the **Corner** scene, but a real target of the same geometry with a fixed 90° angle was used to obtain and compare data from a real ToF camera (Figure 5.14). The surface of the object is coated with finished paper. While recycled unprinted paper can be modeled with a lambertian reflectance model, printed or finished paper will show distinct specular reflexes

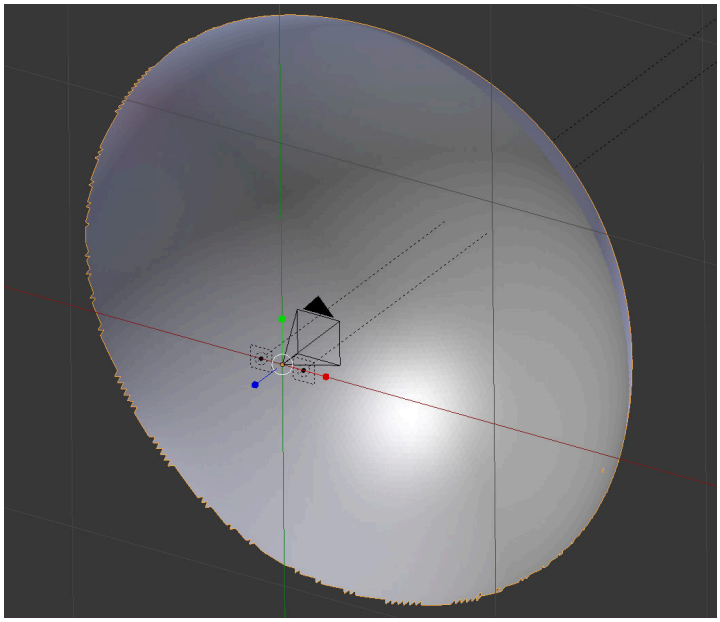


Figure 5.12: ***Shell scene***, modeled as a partial sphere with the camera and illumination at the center. For each pixel the ground truth depth is equal to the sphere radius.

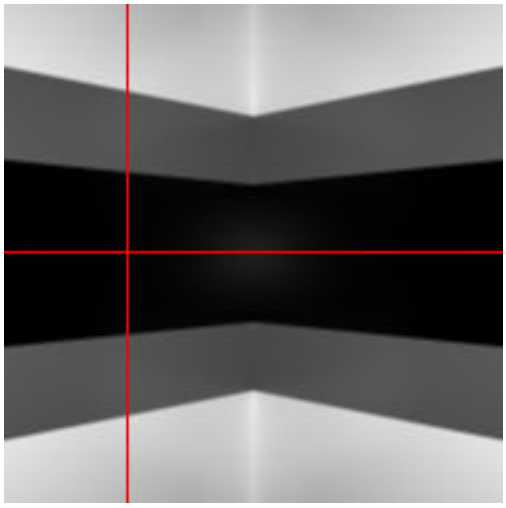


Figure 5.13: ToF Intensity Image of synthetic **Corner** scene (depth profiles along the red lines are shown in Figures 5.35 and 5.36

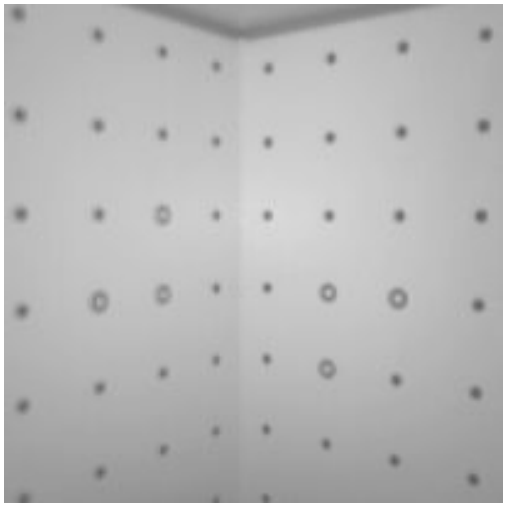


Figure 5.14: ToF Intensity Image **RealCorner** scene

under the right angles. So, for the simulation the same glossy shader as for the Corner scene was used.

Depth reference data for the scenes was either directly available from the synthetic scene description or created by using objects with known geometry as described in section 2.2.1. In this case the 2d-to-3d pose estimation and camera calibration were performed with an Levenberg-Marquardt optimization from the OpenCV image processing library [24]. The meshes have an accuracy of $\approx 1mm$ and reprojection errors of the internal and external camera calibration were below ≈ 0.5pixel. Hence, one can assume that the error in the ground truth depth maps is lower than $\approx 1mm$, which is well below the standard error ranges of most Time-of-Flight cameras. Additionally, the lens distortions of the CamCube 3 were measured using the methods described in Section 2.2. This is necessary, as the specialized optic of the CamCube 3 shows severe barrel distortions, which were subsequently removed.

Additional light sources in the scenes were not simulated, as their non-modulated light contributions can be rendered separately and added as constant offset to all phase images. However, this could be changed in the future if one wanted to simulate or evaluate background light suppression systems. The used light sources were instead modeled with a constant intensity offset similar to real ToF cameras.

To evaluate the algorithm, the generated depth maps are compared with results from a real PMDtec CamCube 3, the simulation method described by Schmidt [136], the simulator by Keller et al. [86], as well as ground truth data. The evaluations have been partially carried out using the Charon software framework [58].

Basic Evaluation

To test the basic properties of the algorithm, mostly the synthetic **Corner** and ***Shell*** scenes were used. Here the effects of varying different algorithm parameters like the number of pixel samples were investigated.

First, the radius of the half-sphere in the Shell scene was varied between 1m and 7.5m in steps of 0.1m. This radius corresponds to the non-ambiguity range for the simulated ToF modulation frequency of 20 Mhz. The difference between the radius and the depth estimated with the algorithm is plotted in Figure 5.15 with respect to the shell radius. The simulation was performed once with deactivated multi-path simulation, once with activated multi-path and once with also with activated multi-path but using a specular material instead of the previously used lambertian material. For the deactivated multi-path simulation, the measured depth is very close to the ground truth (approximately 1 cm difference), however the difference is not zero, mostly due to the minimal distance between the light source and the virtual camera. For activated multi-path simulation, the difference is slightly higher. Although, due to the used lambertian material and the scene geometry, the amount of indirect light is relatively small in this scene so the expected multi-path interference is minimal. For both simulations the error is independent of the actual depth. For true depths above the non-ambiguity range the error does raise by 7.5 as expected (not displayed here). A true depth of $8m$ for example is reported as $0.5m$.

A more drastic change in the observed depth occurs if the scene is rendered with a specular reflecting material. Here the difference is larger (between 7 and 12 cm) and depends on the ground truth radius.

Noise behavior As our goal is a physically correct simulation of the ToF camera, we also have to consider the expected noise behavior. Noise in a real camera can have different sources and distributions, such as poisson shot noise or white thermal noise (which has a gaussian distribution if band-limited). Due to the non-linear reconstruction formulas the resulting depth noise is generally not gaussian shaped. The image noise in a rendered scene can vary greatly in form,

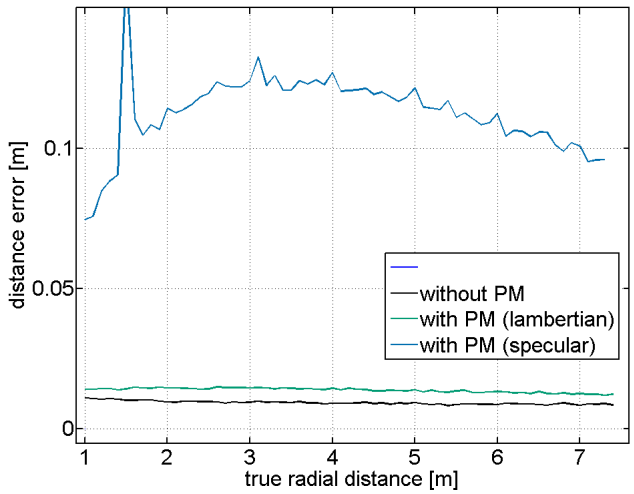


Figure 5.15: Plotted is the distance error wrt varying shell sizes. The error is here defined as the difference between the true depth an the mean depth over the whole image.

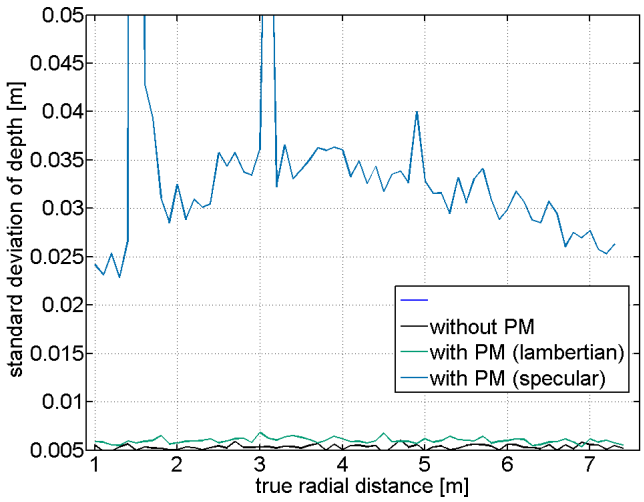


Figure 5.16: Plotted is the standard deviation of the depth over the whole image wrt varying shell sizes.

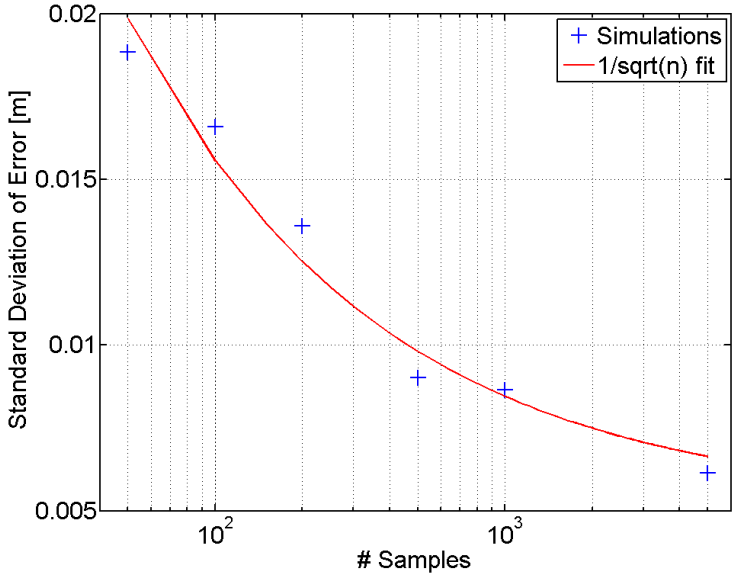


Figure 5.17: Depth noise with respect to the number of samples.

spatial distribution and amplitude, depending on the used sampling strategy. This leaves us with two options for dealing with the noise. One can either increase the number of samples until the render noise is sufficiently small and then distort the result again by adding physically correct noise. Depending on the scene this can increase the render time drastically. Alternatively, one can stop the integration when the render noise has reached a variance which is similar to the expected noise level, even if the distribution is of a different shape.

As stated by the central limit theorem, Monte-Carlo integrators converge with $\frac{1}{\sqrt{N}}$ given the number of samples N. To verify this behavior, multiple simulations with different pixel samplings were conducted. The standard deviation of the depth over the whole image for the Shell scene is plotted in Figure 5.17 with respect to the number of samples. As expected, the value decreases as the number of samples N increases, however it does not approach zero but a fixed offset imposed by the multipath effect. This is also correct as only the noise but not any systematic bias does obey the $\frac{1}{\sqrt{N}}$ law.

Influence of Scene Geometry and Materials Here the effects multipath has on scene geometry have been investigated. For that matter the corners opening angle was changed in steps in between 110 and 60 degree. Figure 5.18 shows the ground

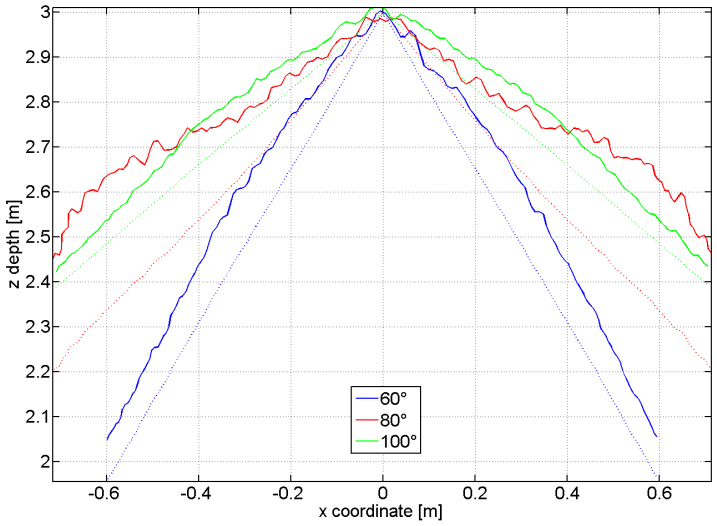


Figure 5.18: Horizontal depth profile for different corner angles (solid lines: simulated, dotted lines: ground truth). Multipath effects are most distinct for an 80° opening angle.

truth form in a top-down view as well as the simulation results. As a simple error metric the differences between the opening angles have been calculated and plotted in Figure 5.19. The simulated opening angle was measured by performing a simple linear line fit on each side of the corner. This is only partly correct for angles where the simulated corner shape was clearly not a plane anymore, however it gives a rough estimate.

For a true opening angle of 80° the simulated form deviates most strongly from the ground truth one, as in this case the angles between the light, camera and centers of the walls fulfill the reflection law almost perfectly. This can also be seen in the intensity images in Figure 5.20. The bright spot, caused by the reflection, moves into the corner and practically vanishes for angles bigger than 85°. (Note that these intensity images were created using bidirectional path tracing (Section 5.5), although this is for visualisation purposes only and there is little optical difference to the photon mapping results.) As these effects can only be explained by light/scene interactions, we can conclude that the algorithm does indeed simulate multipath effects correctly.

Similar results can also be seen in the RealCorner scenes. Multipath reflections of the modulated light between the walls cause an overestimation in the measured distance, as well as a shape change in the corner.

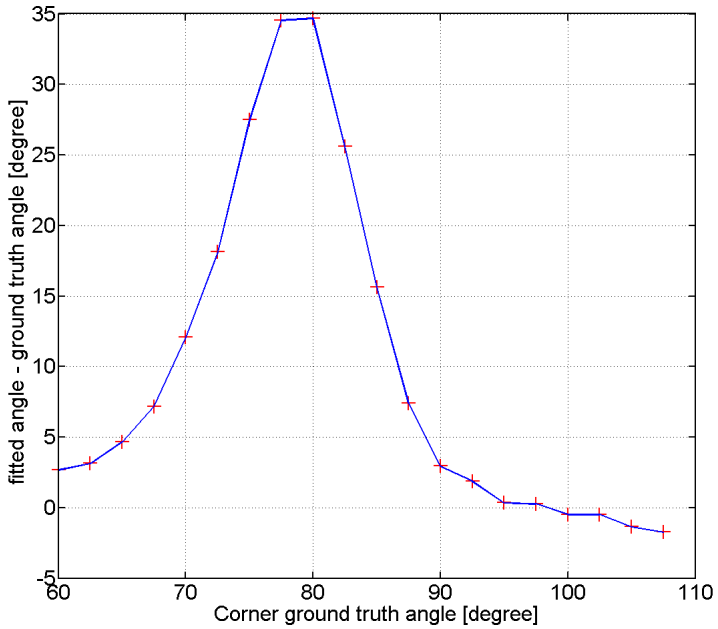


Figure 5.19: Corner scene: Difference between ground truth and fitted line wrt Ground truth corner angle. At 80° the reflection of the light source appears in the middle of the side walls which increases the distortions caused by multipath effects (See Figure 5.20).

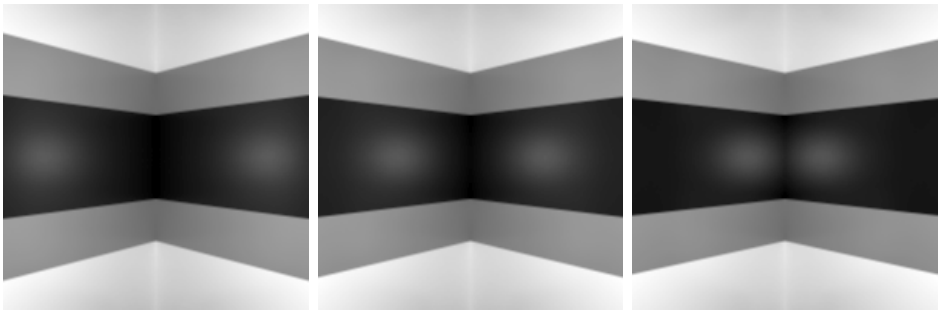


Figure 5.20: Intensity image of Corner (changed gamma) for opening angles of 75°, 78° and 85°. At this angles the reflection law is fulfilled which distorts the shape of the wall. Note that these images were created using bidirectional path tracing (Section 5.5), although there is little optical difference to the photon mapping result.

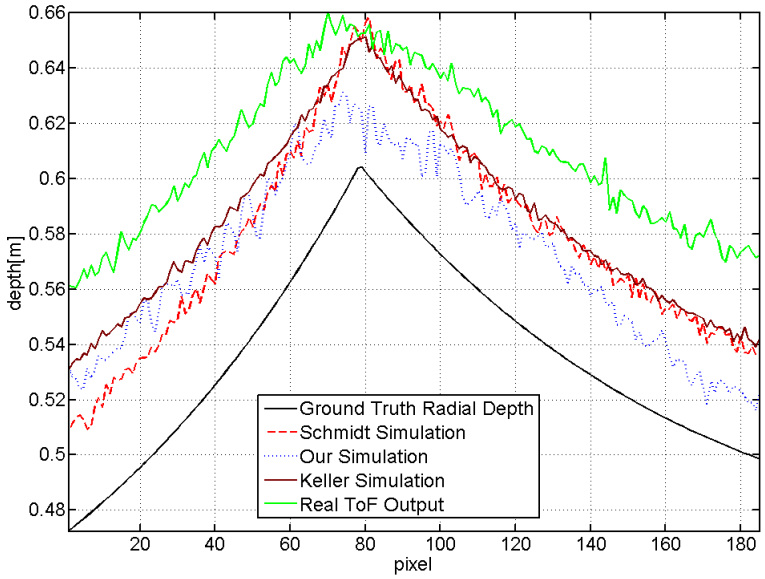


Figure 5.21: Horizontal depth profile of corner scene through the middle line of the sensor. All values are radial distances to the camera principal point. The method correctly simulated the rounded corner caused by interreflection.

The simulation results as well as the output of a real ToF camera can be seen in Figure 5.21. Shown is a cross section of the sensors middle scanline.

The real ToF cameras does produce a significant offset relative to the ground truth depth as well as distorted geometry, mostly direct in the corner. The simulators by Schmidt and Keller don't simulate multipath effects and the resulting depth therefore has no rounded corner but is mostly as sharp as the ground truth. However they partially simulate the correct depth offset. The global illumination based simulation does produce a rounded corner which is a strong indicator that multi-path effects were simulated correctly.

Closer examination of the individual light contributions did reveal that the light intensity of the diffuse indirect illumination was about two to three orders of magnitude smaller than the direct and specular indirect illumination. We can therefore assume that the influence of the indirect diffuse illumination on the measured geometry was negligible.

Figure 5.22 shows that the simulated depth response is similar to the result of a real time-of-flight camera. For visualization purposes we shifted the depth values of the simulations to the same mean as the real time-of-flight image. The noise behavior in all simulations is similar, however the noise sources are different.

	Real	Schmidt	Keller	Our
Real	1.0000	−0.4576	0.5600	**0.6301**
Schmidt	−0.4576	1.0000	−0.1162	−0.2342
Keller	0.5600	−0.1162	1.0000	0.7246
Our	**0.6301**	−0.2342	0.7246	1.0000

Table 5.1: Spearman correlation matrix for real time-of-flight data, Schmidt's method, Keller's method and our simulation. Higher values denote better correlations.

Keller's simulator does sample normal distributed noise while Schmidts simulator generates noise by simulating the sensor and electronics. The noise in the global illumination based simulation is caused by the Monte-Carlo sampling.

For a more quantitative comparison he spearman correlation between the simulated depths and the real time-of-flight signal was computed to clarify statistical similarities. See Table 5.1 for the correlation matrix. The correlation between the global illumination method and the real camera output is the highest which suggests that the method is slightly superior for this case. The results indicate that even without a sophisticated sensor electronic simulation we are able to get a more accurate approximate reproduction of the depth bias than previous methods.

One problem with the algorithm becomes obvious when the individual light contributions are investigated separately. Figure 5.23 shows a similar (although not exact the same) Corner scene where the simulation was performed once only with indirect diffuse illumination and once with specular illumination. One can see that the simulated depth is very close to the ground truth depth in the former case as the multipath effect is very small. Once we add specular illumination the depth as well as the corner shape changes. This is problematic as the indirect diffuse illumination is achieved by the photon maps which are a significant part of the algorithm. Unless there is a scene setup where this type of illumination dominates it ineffective to contribute computation time to this contribution.

5.4.4 Conclusion

The results have shown that with photonmapping one can indeed simulate the multipath effect of Time-of-Flight cameras. However, a significant step of the algorithm both in implementation complexity and computation time has little influence on the observed results. This may be caused by the special scene setups used to evaluate the algorithm but it seems that diffuse indirect illumination plays

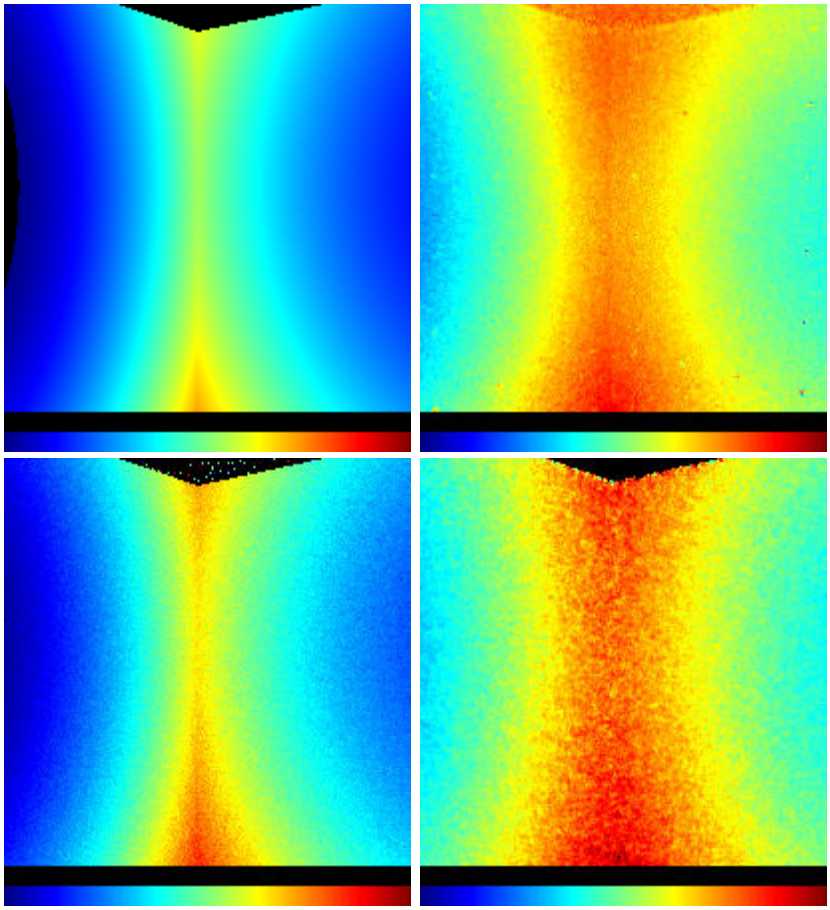


Figure 5.22: Corner scene depth maps: **ground truth**(top left), **real time-of-flight image**(top right), **Schmidt's simulation**(bottom left), **our simulation**(bottom right). All colormaps are scaled equally. The real and simulated values were shifted to the same mean depth (differences between Schmidt's and Keller's method were minimal). Our simulation is very close to the real cameras depth profile.

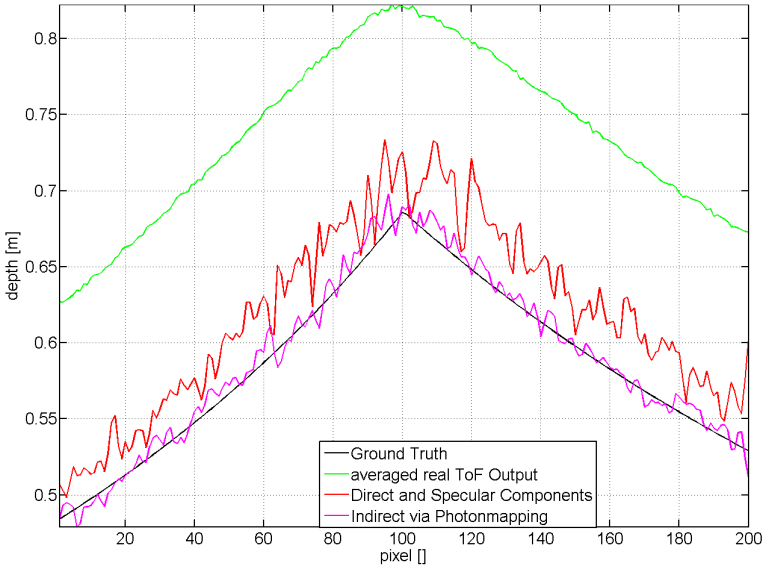


Figure 5.23: Horizontal depth profile with separated contributions. The depth variations caused by indirect lightning are negligible compared to the ones caused by specular reflections.

only a minor role in explaining the behavior of ToF depth imaging. Cases were the modulated illumination is partly obstructed may benefit from the algorithm but the goal of this examination was to develop a more generic method for creating synthetic ground truth data. Additionally, correct parametrization of the algorithm proved difficult because of the many settings that influence the render results. This also applies to the simulation of different materials which can behave quite differently under infrared illumination. With these problems in mind I investigated an alternative approach using another global illumination method.

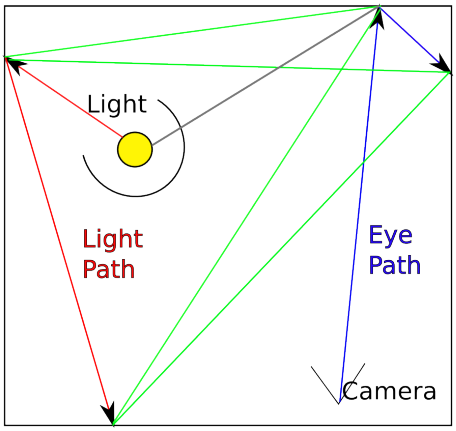


Figure 5.24: Principle of Bidirectional Path Tracing: eye paths are blue, light path are red. At each intersection point the paths are connected (green lines). Additionally, eye path intersections are connected directly with the light sources unless they are obstructed (gray path, here invalid due to shadowing).

5.5 Algorithm Investigation: Bidirectional Path Tracing

In this section I will describe an alternative approach to ToF simulation that uses *Bidirectional Path Tracing*[97, 155] for the computation of global illumination. The described ToF simulation algorithm was developed with the problems in mind that emerged during the investigations into the modified photon mapping algorithm. The working principle is similar to the second part of the photon mapping algorithm which uses raytracing to deal with specular and transparent surfaces. Otherwise most of the scene and simulation considerations discussed in the previous section also apply to Bidirectional Path Tracing. The method presented here is capable of simulating all scene dependent effects in ToF data including multipath effects, mirroring and "ghost depth images" due to transparent materials.

The results in this section have partially been published in [6].

5.5.1 Method Description and Algorithm Motivation

Bidirectional Path Tracing was introduced by Lafortune and Willems [97] and independently by Veach and Guibas [155] to solve the global illumination problem.

The following detailed description is again based on an implementation from the Luxrender project [60], which itself is based on the pbrt render engine by Pharr and Humphrey [120].

The basic principle of the algorithm is visualized in Figure 5.24. A single pixel sample is created in the following way. A view ray from the camera is sampled and traced into the scene. Once it intersects the scene geometry a secondary ray is sampled recursively according to the local bidirectional reflectance distribution function (BRDF) until a maximum recursion depth (or number of bounces) is reached. This produces a series of so called "eye-path" intersection points $e_0 \rightarrow e_1 \rightarrow\ e_2 \rightarrow ...e_n$ (Blue paths in the figure). We are interested in the light that is emitted from e_0 towards the camera.

Then, in a similar manner, a light source in the scene is selected at random (or according to some light power or importance parameter) and a light ray is sampled. This results in a light path $l_0 \rightarrow l_1 \rightarrow l_2 \rightarrow ...l_m$ (red paths in the figure).

A given view ray intersection point is then connected to all light sources and all light ray intersection points, given that these connections are not occluded by other scene geometry elements (In the figure valid connections are green, an occluded one is gray). This means the light arriving at e_n is the sum of the light coming from all l_i for $i = [0, .., m]$ plus the light coming from all light sources. The light arriving at e_{n-1} is then the light coming from e_n plus all l_i plus all light sources.

Sophisticated sampling strategies as described in Section 5.3.2 can be used to create rays directed towards more *interesting* parts of the scene. Additionally, Russian Roulette strategies may interrupt the path at random or if its light contribution is negligible.

The advantage of using paths from both directions is the faster convergence in complicated light situations. Take for example a scene that consists of a closed room, illuminated by light falling trough a small window. If only eye paths were used, very many rays would need to be cast before one of them leaves the room trough the window. If only light paths would be used, most of them would be stopped by the outside walls of them room and could not contribute to the illumination inside. By using both paths the algorithm can give priority to paths along which energy is transported through the window.

For unidirectional path tracing (a special case where the maximum light path length m is zero) the chance that a randomly sampled eye path will hit a lightsource directly is small (and even zero for point lights). The light paths are therefore also important for rendering caustics.

Not all of these scenarios are significant for typical Time-of-Flight scenarios but the usage of bidirectional tracing does impose none to little complexity impacts over unidirectional tracing. If the scene setup is suited for unidirectional tracing, the light path length can be reduced for faster convergence of the integration.

So far all ToF cameras use light sources which are mounted coaxial and in close proximity to the optics so this is a valid simplification in most cases. This however may change in future cameras or special settings and would lead to more complex light paths. For these cases, bidirectional path tracing may indeed be more suited than unidirectional tracing and it is therefore not unreasonable to directly implement the more general version.

ToF simulation is performed equivalently to the description in Section 5.4.1. The length of a path is defined as the combined lengths of a light and a eye path plus the additional ray connecting both. Note that not the final intensity of a single iteration of the algorithm is modulated but each individual contribution along the path.

Here the light modulation was additionally modified to simulate the effects of not perfectly sinusoidal illumination signals. This can be achieved by adding higher harmonics to the intensity modulation. As shown by Rapp [124], in case of the PMDtec Camcube, the intensity of higher harmonic components in the light signal drops rather fast. For multipliers higher than four, the intensity is smaller than < 0.01 relative to the intensity of the base frequency. Additionally, it has been shown by Plaue [122] that harmonic modulations with an even multiplier do not influence the measured depth.

The modulation factor from Equation (5.20) can therefore be modified as follows:

$$m_{i2} = m_i + \sum_{i=0}^{N} \cos\left(\left(\frac{(2i+1)d\ f\ 2\pi}{c} + \theta_k\right) + \phi_i\right) \cdot I_i \tag{5.24}$$

- N maximum harmonic
- ϕ_i phase shift of ith harmonic
- I_i relative intensity of i-th harmonic

Each harmonic can be phase shifted relative to the base signal (ϕ_i). This does influence the signal form and hence also the resulting depth. However, to the authors knowledge, no indepth examination of the harmonic phase shifts in ToF cameras have been performed.

Additionally, all other formulas from (5.20) to (5.23) regularly apply here.

Differences to photon mapping

There are multiple practical differences between photon mapping and bidirectional path tracing that will be mentioned here:

Uniform material handling Unlike photon mapping, path tracing does not treat diffuse and specular materials differently. Photon mapping decides based on certain flags that designate a material as specular, diffuse or transparent in which photon map a given photon is saved and whether a view ray should be followed recursively. Path tracing only uses the local BRDF to decide on the direction of a secondary rays.

Reduced memory consumption Photon mapping must hold the individual photon maps in memory during the whole render process. For a few million photon events, the memory requirements can reach hundreds of megabytes. In path tracing only the current light and eye paths need to be kept and can be discarded for each new pixel sample. This also makes parallelization easier.

Non-Biased If the noise levels for a given setting and runtime are unsatisfactory, there is no need for parameterizing of the scene. The algorithm will eventually converge to a physically correct solution given enough render time. In photon mapping it would be necessary to recalculate the photon map to reduce bias.

Easier parametrization In total path tracing has only two unique parameters that can be set (light path length and eye path length). Photon mapping in the used implementation has six parameters (number of indirect photons, number of caustic photons, photon integration count, photon integration radius, photon depth and ray recursion depth). This value is even higher when the final gathering optimization is used which is advisable for better results. With the wrong settings, photon mapping can be severely biased. Finding the optimal parameters for a scene is therefore significantly easier. Additional parameters like the number of samples, the used sampler or material settings are generally shared by both implementations.

Caching of intermediate results/faster convergence Unless the scene does change between acquisitions of the raw frames, photon mapping can be used to reduce the computational cost. The photon map must only be computed once and can be reused for different phase angles.

Noise predictability Photon mapping is often used when a predictable rendertime to noise ratio is important. For artistic movie productions it is important

that consecutive renderings have approximately the same noise characteristics and potential artifacts given the same render time. Photonmaps can be reused between different frames as long as the scene content didn't change (e.g. when only the camera has moved) and their generation time is relatively constant. Path tracing, which is a pure Monte Carlo method can give quite different results on the same scene for the same render time or number of samples per pixel. This is in fact a advantages for photon mapping as far as Time-of-Flight simulation is considered as this applies directly to the creation of the four raw phase images.

Algorithm parameters

Like in most rendering algorithms, it is possible to shift priority between speed of execution, memory consumption and physical realism. The most influential parameters of the path tracer are the maximum recursion depth for light and eye paths as well as the number of samples per pixel. At an eye recursion depth of one, no multipath effects can be observed. When the light path depth is set to zero, the algorithm is reduced to a unidirectional path tracer. The optimal value is scene dependent but as sensor and lightsource on real ToF cameras are usually very close together direct illumination dominates and path depths of approximately 8 are sufficient.

Just like most global illumination methods, path tracing is a statistical process and will converge to a correct solution as the number of samples per pixels increases. As each of the phase images is rendered independently, the statistical sampling noise can influence the produced depth maps significantly. Usually, a few hundred raytracing samples per pixel are enough for visually convincing results, but in our case a the number of samples should be one order of magnitude higher to create accurate depth maps.

Caching of intermediate results and direct computation of all required phase shifts could reduce this problem in future versions of the algorithm, although in that case it would not be possible anymore to simulate motion artifacts as each frame would be limited to exactly the same geometry.

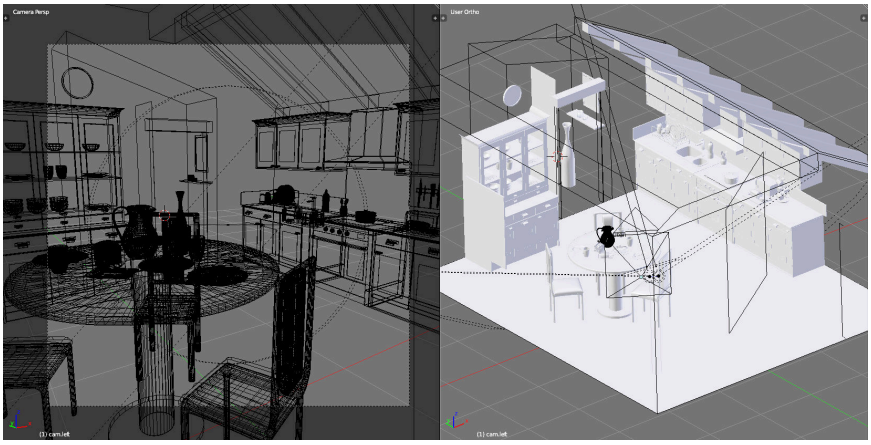


Figure 5.25: Wireframe rendering and scene layout of Kitchen.

5.5.2 Experiments

To evaluate the algorithm some of the experimental setups from the photon-mapping section (5.4.3) were reused or slightly modified. Additionally, two new synthetic setups were added to evaluate more complex light interactions.

First, a new synthetic scene, labeled **Kitchen**[30], shows a typical household kitchen with various reflecting and transparent objects. It is mainly used to demonstrate that the algorithm can deal with complex setups as it combines many different real world effects. A photorealistic rendering is displayed in Figure 5.40 and raw phase images of the scene are displayed in Figure 5.7.

The scene titled ***Reflection*** is used to evaluate the effects of transparent materials. A conceptually similar but not identical real world scene is used for a qualitative comparison. In the scene a transparent plane (acrylic glass in the real world case) is held at a roughly 45 degree angle to the camera so that a mirror reflection of another object is partially visible. The rendering and real world intensity image can be seen in Figure 5.26.

The **Box** scene (Figure 5.30) mentioned earlier was also used for comparison of simulation data with real ToF output. The styrofoam targets visible in earlier images produces additional stray light and where subsequently removed.

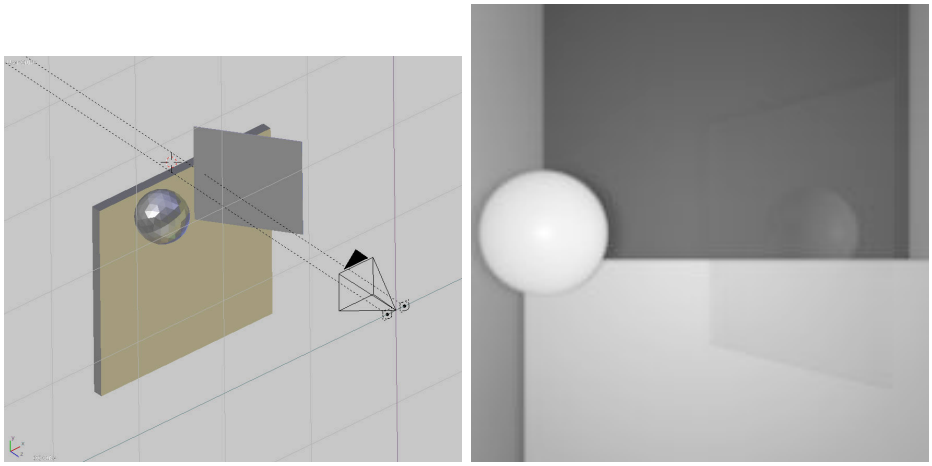


Figure 5.26: Scene Layout and rendered intensity of Reflection scene. The partially transparent plane on the right produces a reflection image of the sphere. This also effects the measured depth.

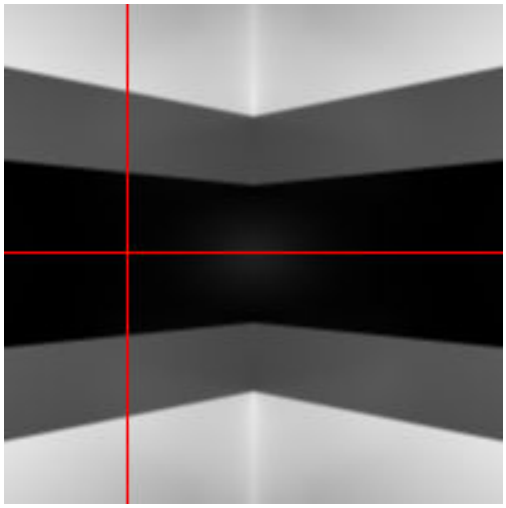


Figure 5.27: ToF Intensity Image of synthetic **Corner** scene (depth profiles along the red lines are shown in Figures 5.35 and 5.36

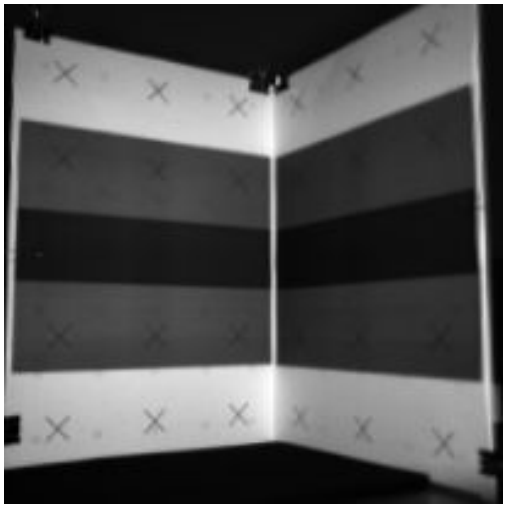


Figure 5.28: Intensity image of **RealCorner**. The black crosses where used as features for the 2D-to-3D pose estimation.

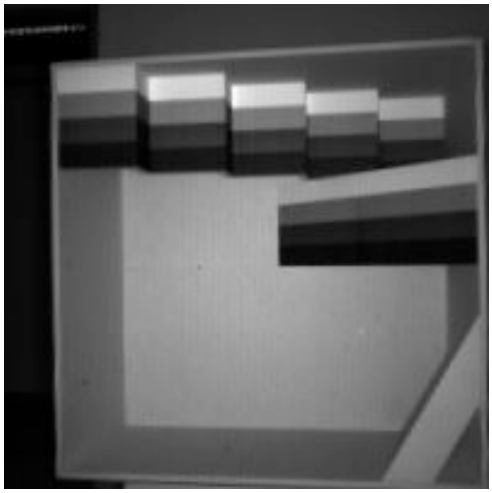


Figure 5.29: Intensity image of target ***Box*** taken with real ToF camera. Note the missing styrofoam targets

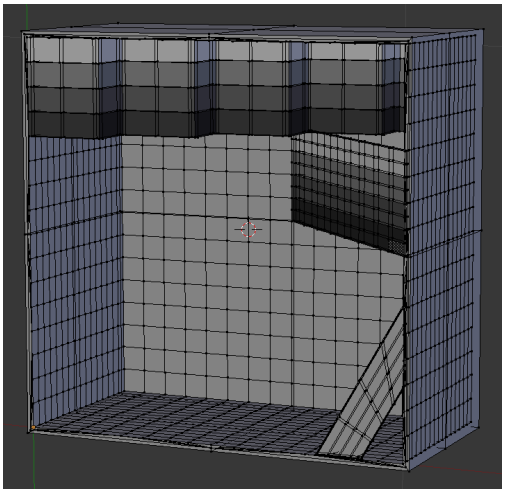


Figure 5.30: Polygon mesh of target ***Box***

Basic Evaluation

To test the basic properties of the algorithm mostly the synthetic **Corner** and ***Shell*** scenes were used. Here the effects of varying different algorithm parameters like recursion depth were investigated. In the corner scene multiple reflections of the light between both walls can cause the planes to appear curved as can be seen in the top down view (Figure 5.35). Multipath effects are also present in the shell scene, however at a smaller degree and due to the geometry they were radialsymmetric.

Noise behavior The $\frac{1}{\sqrt{N}}$ law for the error mentioned in the photonmapping section should also apply for this method. Again multiple simulations at varying N were conducted. The resulting error values (standard deviation of the depth over the whole image) are displayed in Figure 5.31. It should be noted that the resulting error values are specific for every scene. Reflecting or transparent regions may need more samples to converge, so there is no simple rule for estimating the needed number of per pixel samples. Experiments have shown that for simple geometry and materials 5000 samples is usually sufficient to reach a depth variance which is in the same range or below the results from a real ToF camera.

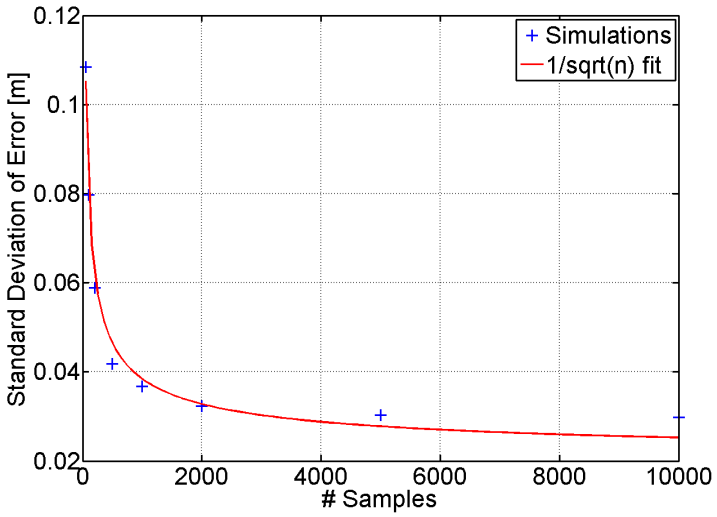


Figure 5.31: Depth noise wrt number of pixel samples. The error follows a $\frac{1}{\sqrt{N}}$ curve as expected for a Monte-Carlo method.

Influence of eye and light path length Path length denotes here the maximum number of bounces or scene intersections that are allowed for a eye or light path before it is cut off, not the physical length of the path. When the path tracing is interrupted after the first intersection no interreflection takes place and the resulting depth corresponds exactly to the ground truth depth safe for some rendering noise. For the Shell scene this can be seen in Figure 5.32 (the line 1/0) where the difference between the measured depth and the true depth is plotted. The measured depth is here defined as the spatial average over a 20x20 pixel window to eliminate effects due to rendering noise (due to the scene geometry, all pixels should have the same depth). If the combined path length is increased to two (line 1/1 or 2/0) multipath effects which change the measured depth take place. For shell sizes larger than $\approx$ 5 m the error decreases again. This could be caused by light paths which are (physically) longer than the non-ambiguity range. If these paths contribute they can effectively decrease the measured depth (A ligth path of length $10m$ would have an effective length of 2.5m for a non-ambiguity range of 7.5m). This effect takes places even earlier if the path lengths are increased further (see paths 2/1, 3/1 and 3/3). Here the maximum is reached for a true depth of $3m$. For this simple scene setup increasing the path lengths even more has little effect as light that bounced around the scene more often is to weak to have any influence.

For the Corner scene the effects are similar (See the horizontal 5.35 and vertical

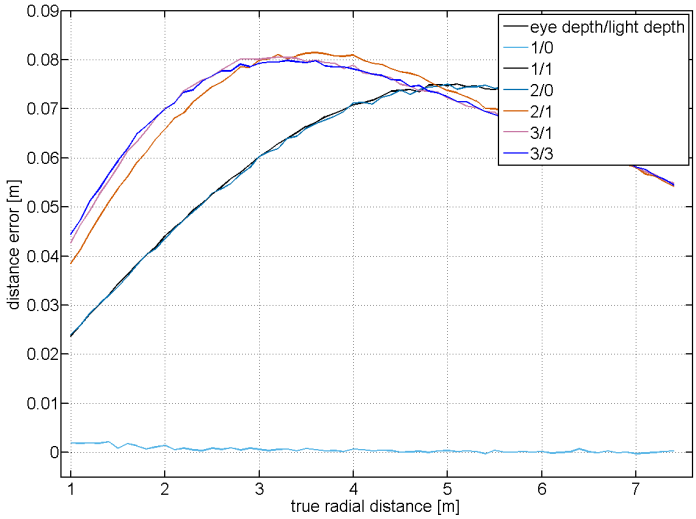


Figure 5.32: Plotted is the distance error wrt varying shell sizes at different path lengths. The error is here the difference between the true depth an the mean of a 20 by 20 pixel window.

5.36 depth profiles). Combined eye and light path lengths higher than one cause a distorted depth profile where the two walls of the corner are not flat planes any more. For an eye and light path length of one each render artifacts occur. This is visualized by the high noise for the 1/1 lines in the two depth profiles as well as the increased depth variance in Figure 5.34. Path lengths higher than 5 would only be necessary in complex scenes where the illumination is purely indirect and increasing this value has little to no effect in typical ToF setups.

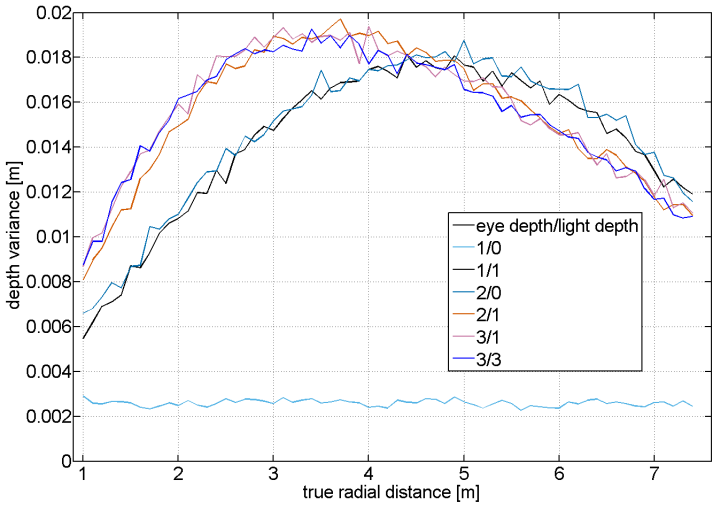


Figure 5.33: Plotted is the standard deviation wrt varying shell sizes at different path lengths. The deviation is taken over the same image window as in Figure 5.32.

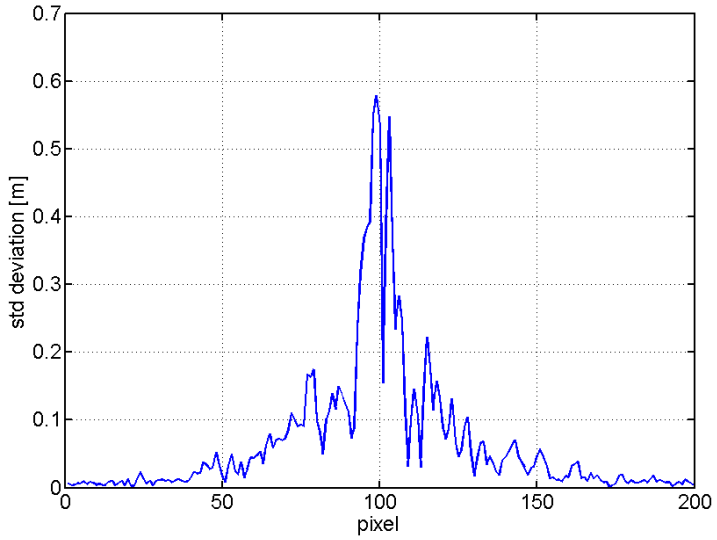


Figure 5.34: Depth std deviation for simulation of **Corner** scene with path length 2. The deviation was computed along the vertical scanlines. Rendering artifacts in the center suggest that this path length is insufficient.

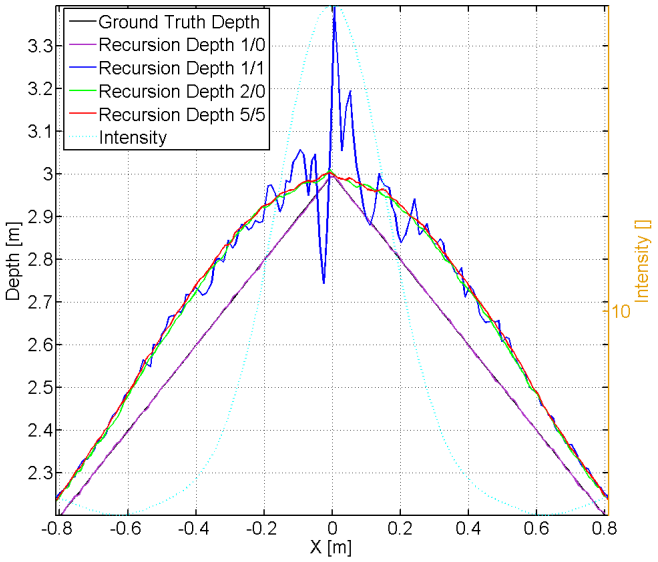


Figure 5.35: Horizontal depth profiles along the lines in Figure 5.27 for different path lengths. For lengths of one, no multipath effects occur, lengths of two show strong render artifacts in the center.

Influence of Scene Geometry Here the experiment of Section 5.4.3 concerning the variable opening angle of the corner was repeated. Figure 5.37 shows the ground truth form in a top-down view as well as the simulation results. The difference between the opening angle and a linear fit is again plotted in Figure 5.38. The data shows similar behaviour which suggest that multipath simulation also works in this case and that there are no significant differences for specular materials.

Influence of Material Properties

Apart from regular reflection and multireflection, different materials can have different brightness levels in the resulting ToF images. ToF cameras are known to exhibit a intensity dependent depth offset which is mainly caused by the non-linear photo response function of the sensor ([136]). Examples can be seen in the vertical depth profile of the **Corner** scene (Figure 5.36) and the slope in the **Box** scene (region A in Figure 5.39). The vertical slice in the former shows a higher measured depth in the center, where dark material was used, while the

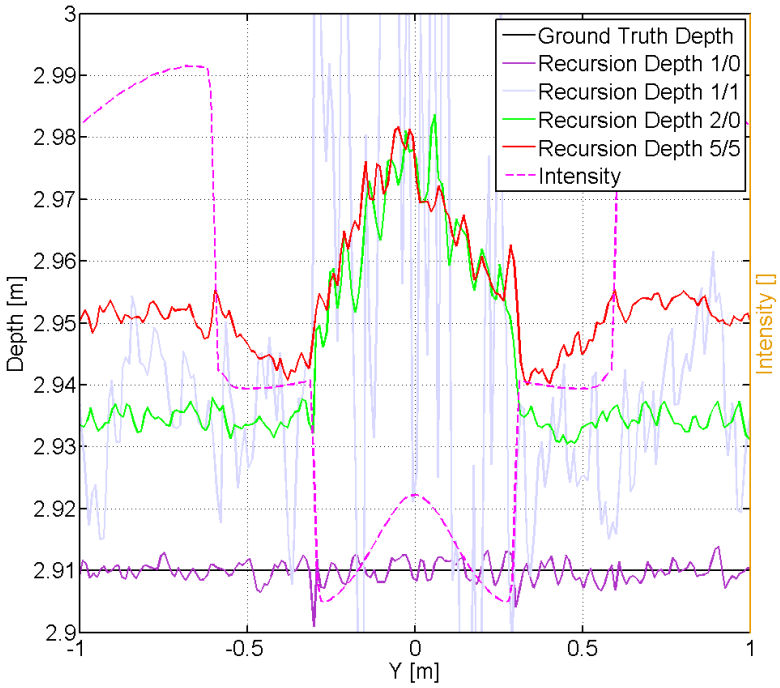


Figure 5.36: Vertical depth profiles along the lines in Figure 5.27 for different path lengths, The measured depth changes with the observed intensity. Multipath effects are strongest in the center as the relative contribution of direct light is weaker.

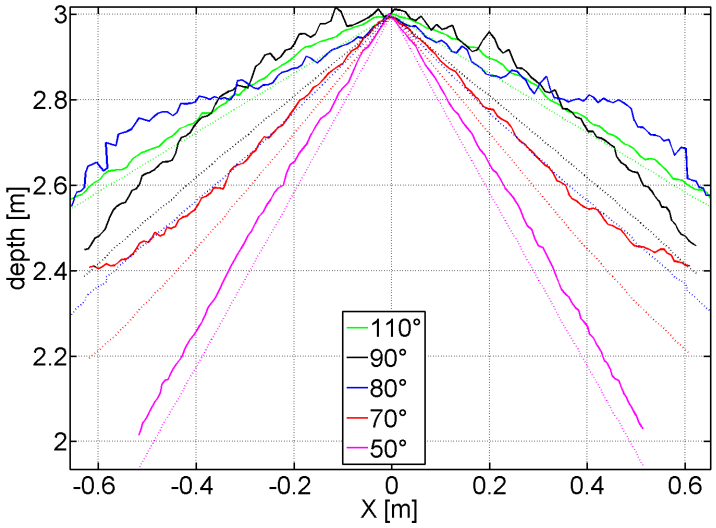


Figure 5.37: Horizontal depth profile for different corner angles (solid lines: simulated, dotted lines: ground truth). Multipath effects are most distinct for an 80° opening angle.

darker part of the slope in the latter rises at a more shallow angle. In these regions the light which was reflected from brighter regions and has taken a longer path has a higher relative contribution over the direct light.

Both our method, as well as Schmidt's method reproduce this effect to a certain degree, although based on different physical principles. For Schmidt's simulator the offset is mostly caused by simulating the non-linear pixel response curves, which changes the relative intensity differences between the raw phase images. In our method multipath effects are more distinct in dark areas as here the relative intensity contribution between direct and indirect light is different than for brighter materials. Additional material based effects are for example the bulge in region C of Figure 5.39. This is probably caused by the fact that many materials appear different under low angles of deflection. In these cases the simple models fail as interactions between microscopic surface layers (e.g. due to coating) etc. become dominant. The unfinished wood can show strong reflection behavior near the gracing angle. The depth behavior in this region could not yet be reproduced reliably by any method, however better material shaders or measured BRDFs could do so.

The **Kitchen** scene shows examples of more effects present in more complex scene setups and real life conditions. This however only a qualitative comparison. Clearly visible are artifacts caused by reflections or transparent materials such as

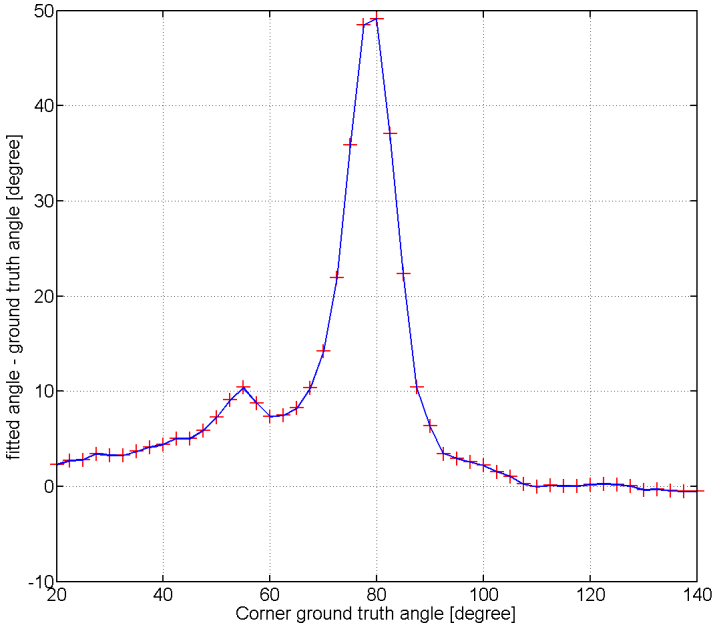


Figure 5.38: Corner scene: Difference between ground truth and fitted line wrt Ground truth corner angle. At 80° the reflection of the light source appears in the middle of the side walls which increases the distortions caused by multipath effects (See Figure 5.20).

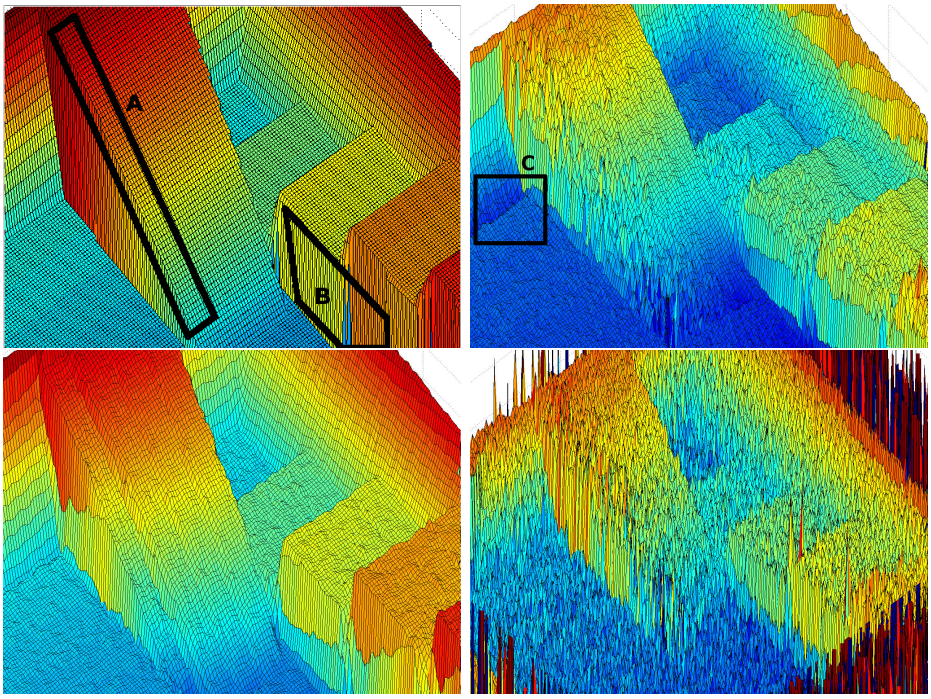


Figure 5.39: Surface plots of target box. **Top left:** Ground truth; **Top right:** real ToF output; **Bottom left:** our simulation; **Bottom right:** simulation by Schmidt. Visible are the intensity dependent slop (A), flying pixel (B) and the bulge in the real image caused by material properties (C).

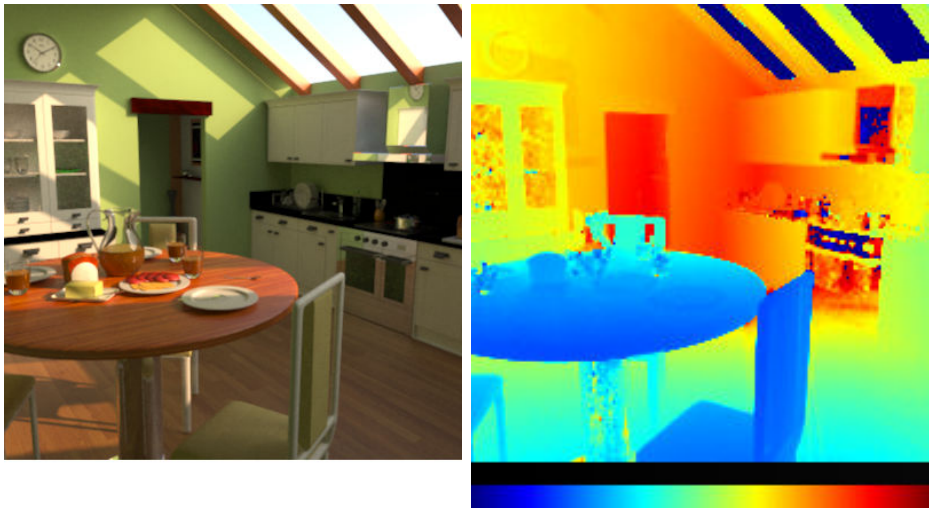


Figure 5.40: **Left:** Photorealistic rendering of scene. **Right:** Simulated ToF depth map. Depth range from 0 m(blue) to 7.5 m(red). Materials like glass or chrome create artifacts in ToF depth images. Examples are the oven door or the glass cabinet.

the glass cups or the chrome surfaces on the oven or the table-leg. The cooker hood for example does reflect parts of the back wall and the side of the white cupboard. The cupboard side is not illuminated directly by the ToF illumination (which is coaxial to the camera) and is therefore darker (See also raw phase images in Figure 5.7). Some similar effects can be seen in the real ToF scene displayed in Figure 5.6.

Comparison and combination of different simulation methods

To compare our method to existing ToF simulators we primarily used the **Real-Corner** and **Box** scenes (Figures 5.41 and 5.30). For the visualization we shifted the depth values to their common mean. This is justified as all ToF cameras and simulation methods introduce a different bias into the data, however these can be nullified by known depth calibration techniques.

The method of Schmidt et al. can simulate depth offsets caused by different intensities while Keller's method is real-time capable due to a sophisticated GPU implementation (the bright dot is caused by partial overexposure). We were also able to combine methods by using the depth maps of our algorithm as input for

	Real	Schmidt	Keller	Our	Combi
Real	1.00	0.25	−0.20	**0.19**	**0.42**
Schmidt	0.25	1.00	−0.78	−0.38	0.67
Keller	−0.20	−0.78	1.00	0.38	-0.61
Our	**0.19**	−0.38	0.38	1.00	0.09
Combi	**0.42**	0.67	-0.61	0.09	1.00

Table 5.2: Spearman correlation matrix for real time-of-flight data, Schmidt's method, Keller's method, our simulation and the combi-method. Higher values indicate a simulation result closer to the real camera output.

Schmidt's method. The result is a simulation with included multipath artifacts combined with intensity based offsets and a physical noise model.

To perform a statistic comparison of the algorithm results we first calculated the error images between the ground truth and the real ToF output as well as the difference between ground truth and each individual simulation. We then computed the spearman correlation between each pair of error images (Table 5.2). The closer the correlation between the real ToF output and a given simulation method, the higher the realism of the method. Our method (bold) is slightly worse than Schmidt's method (green), mainly because the later one simulates the intensity based offset more accurately. Combining both methods (blue) significantly improves the results as now nearly all existing physical effects are handled.

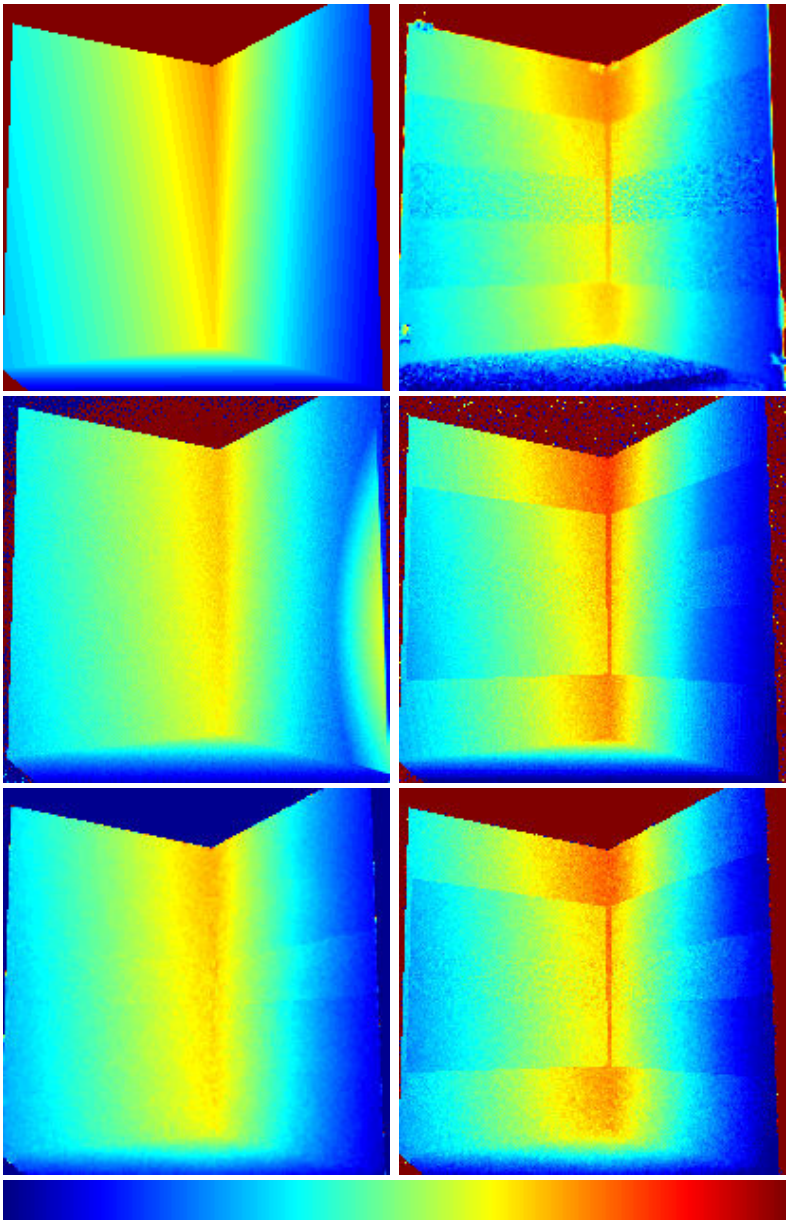


Figure 5.41: Colorcoded Depth Maps for **RealCorner** scene, **top left:** ground truth, **top right:** real ToF camera, **middle left:** Keller's method, **middle right:** Schmidt's method, **bottom left:** our method, **bottom right:** combination of our and Schmidt's method. All depth values were shifted to the same mean depth: 0.5 m (blue) to 0.95 m (red).

5.6 Conclusion and Future Work

Based on a modification of the bidirectional path tracing algorithm, a physically correct simulation of Time-of-Flight cameras, including multipath interference, has been performed. We have shown that the method can correctly simulate flying pixels, depth errors due to multiple interreflections as well as transparent or strongly reflecting materials based on the scene geometry. Furthermore, it is able to simulate Time-of-Flight depth distortions caused by different materials in the scene. The simulation is so far the only one which accounts for all these effects, although it does not yet address all problems like intensity dependent offsets or wiggling errors. However, we have shown that our simulation method can be combined with other sophisticated sensor simulation methods to cover a wider field of ToF related problems. By taking into consideration the physical and geometric setup of a scene it is possible to create more realistic images and depth maps for statistical analysis, evaluation, denoising and test of Time-of-Flight cameras. This enables researchers and practitioners to create synthetic test datasets for various environmental conditions, scenes or resolutions without the need of a real camera setup.

However, not all characteristics of the simulator are yet known and multiple future improvements are possible.

First of all, setting all possible material properties to match real world materials is a time consuming process and is only possible with some experience in Luxrender or at least general rendering solutions. The resulting depth maps of the simulation are very sensitive to the used models and not all real world materials can be represented adequately by the available shaders. Examples are wood, rugs or plants which may show various effects like subsurface scattering or brightness strongly dependent on the incidence angle. This is complicated even more by the fact that materials may act different under infrared illumination and available or measured parameterizations are often only available for visible light. It can therefore be expected that the quality of the simulation can be improved significantly by using measured BRDF data. New, compact representations of such data, as for example presented by Ruiters and Klein [132], could help in dealing with the large amounts of data involved in this process.

Second, simulation speeds of the method are currently significantly longer than those of other Time-of-Flight simulators. Improvements are possible for example by using GPU acceleration of the ray shooting process. This is already implemented in Luxrender but not yet utilized in the ToF simulation. Additionally, the render time could be cut to one fourth if the simulation of individual raw frames is omitted in favor of faster results.

The experiences made during the development of this simulator support the findings that synthetic reference data can be used for evaluation purposes but the effort to create them is still significant. Unless depth ground truth can not be created by other means, it is in many cases preferable to perform test on real world data. Exceptions are also situations where not yet existing cameras must be tested or when explicit control over the raw frames is required, for example for testing motion compensation solutions.

6 Conclusion

In this work reference datasets have been investigated as an important tool for algorithm testing and performance evaluation in image processing. Three types of reference data could be identified: Real world reference data with ground truth, real world reference data without ground truth and synthetic reference data. For each of these types basic creation and usage considerations were examined and concrete examples presented.

6.1 Summary

In Chapter 1, we introduced the concepts of performance evaluation, benchmarking and ground truth. One main argument is, that engineering practices should be used in the creation process of reference data as only with a clear definition of algorithm domains and requirements the data can be used to its full potential. Additionally, we presented a few best practices for dataset creation, handling and publishing.

Chapter 2 describes the basics of real world datasets with ground truth for dense correspondence problems. Solving a correspondence problem is an intermediate step in many image processing applications, both low as well as high level. We have described how dense correspondence data can be created from just a few manually selected sparse correspondences, using known 3D meshes and camera calibration techniques. Our results indicate that for high accuracy requirements on scenes or objects with sizes of less than two or three meters, structured light scanning is currently the best scanning method. For scenes of larger size, (terrestrial) laser scanning is a valid alternative. Furthermore, we identified and evaluated the KinectFusion algorithm as a method for creating large amounts of reference data with lower accuracy at significantly reduced scanning efforts. We found that the resulting meshes have accuracies between 10 and 80 mm, depending on scene size. Correspondence reference data acquired with these meshes can reach accuracies of well below one pixel.

In Chapter 3 we investigated the significance of datasets without or with only weak ground truth. Many test scenarios don't need accurate reference data or

are based on qualitative comparisons. E.g., testing of fault cases or also machine learning can be performed on unannotated data. To emphasize the usefulness of such data two stereo datasets aimed at automotive applications have been presented.

Chapter 4 examines the concept and problems of synthetic reference datasets. Advancements in computer graphics as well as photorealistic productions from the entertainment industry suggest that synthetic images can be used in cases were real ground truth imagery is hard to obtain. To that matter we investigated the results of optical flow computation on both a synthetic and a real sequence. Furthermore, we created a hybrid dataset for the evaluation of light field algorithms. The results indicate, that the realism of current graphics algorithms is sufficient for evaluation purposes, but creating synthetic sequences with ground truth is neither fast, nor easier than creating real world sequences. Synthetic data is therefore best suited for cases where real world reference data is very hard to obtain or where the possibility to change smaller test scene characteristics is important for the evaluation process.

Finally, in Chapter 5, we demonstrated how computer graphics can be used to create reference data for the non-standard case of Time-of-Flight imaging. By simulating the time-dependent light propagation in a scene and combining our method with an existing ToF simulator we could reproduce all relevant error sources of a ToF camera. By separating these error sources it is possible to create test data for different aspects of the imaging system to evaluate existing and future ToF cameras.

6.2 Consequences and Recommendations

We can draw multiple universal conclusions from the experiments in this work.

Image processing has reached a certain level of matureness in that both its theories and tools are solid and well established. What must be improved is practicality and robustness, as established evaluation practices are regularly criticized by practitioners and researchers. Experiences and best practices from engineering sciences and software development such as formal requirements engineering can help identify problems in existing methods. To apply these methods, testing on reference data is still the best method as it delivers comparable and repeatable results.

First, any type of reference dataset is useful, and the more data it contains the better. Highly accurate ground truth is not always necessary, as for example

engineers can draw important conclusions about the performance of an algorithm even without quantitative analysis simply by means of eyeballing. This doesn't mean that we should stop creating high accuracy datasets but we should als keep the requirements of other fields in mind. A combination of methods and accuracies could produce the best results as we can make assumptions about the behavior of a algorithm based on many inaccurate and a few accurate samples. What is important, is that we know the properties of our reference data. It is a bit underwhelming when a dataset is labeled "*including ground truth*" while a closer examination reveals that the *ground truth* is indeed sparse and its variance is larger than three pixels. This mustn't make the data less impressive but tests and experimental design based on it must be adapted to these properties.

Having large amounts of data is definitely advantageous as it allows for more robust error statistics and helps in covering more broader application and input domains. For machine learning, object recognition and detection, large image databases have always been used. However only recently did the idea gain traction, that also other applications, for example dense correspondence problems, could benefit from large datasets. More and more methods that can create such large datasets are now becoming available. Some of them, for both real and synthetic images, have been presented here. What we are still lacking are methods and infrastructures to use this data to its full potential. When a scientist publishes a new algorithm, tests are often only performed on a few images or a small database, often manually (This thesis is in itself also guilty of this shortcoming). Performing thorough examinations is difficult and time consuming as we lack the tools to select appropriate test cases and to automate performance benchmarking.

That there is no standard for publishing data also doesn't help the fact. Accumulating all existing datasets into a large database with a unified interface is currently not practicable and also probably not useful, but there a few steps that can help making datasets more accessible. Proper documentation of the data and the experiments as well as the inclusion of source code is a first step. We have seen that well laid out data can be used beyond the original scope for which it was created.

The use of synthetic images for performance evaluation has proven difficult, but not necessarily for the reasons initially assumed. Synthetic image sequences are often criticized for being unrealistic which makes the results obtained on these sequences not transferable to real world problems. The experiments conducted in this thesis however indicated that with careful scene and experimental design synthetic scenes can be used for evaluation purposes. What is limiting their usability is the fact that they are not easier or faster to create than real world images, unless setups can be reused or the same scene is to be produced with only

small variations. The different skill sets and assets needed to create synthetic good quality reference data make this tasks a own domain within image processing and computer graphics.

A division of responsibilities as is for example performed in physics with theoretical, experimental and applied physics could also be applied to image processing. The discipline as a whole could benefit from researchers focusing solely on creation of both real and synthetic test data and the evaluation of existing methods.

6.3 Outlook and Alternate Methods

The experiments and methods described in this work are in no way exhaustive.

A point that has been mentioned but not laid out in detail is the creation of reference data via crowdsourcing. Usually ground truth data is created by either setting up the scenes in such a way that the data is known beforehand or it is determined afterwards or in parallel with another measurement method. For some types of data these methods are not practicable, either because they take to much time or because they can't be automated. There are, for example many tasks where evaluation and labeling must still be performed by human experts. An expert can be anything from a highly trained radiologist labeling tumor cells to a layman selecting animals in a zoo picture. For many low-level tasks, such as edge detection, or object recognition, the *expert knowledge* can be obtained in a very short times making ground truth produced by humans at least equal to most automated algorithms [39]. Letting researchers perform such annotations however makes for a poor cost-effectiveness ratio.

It can be assumed that, encouraged by the spread of personal computing devices such as smartphones, reference data creation tasks can be broken down and distributed into pieces manageable by many untrained workers. First attempts using the Amazon Mechanical Turk Platform[1] or mobile computer games such as Cerberus[2] or the soon to be released Geoglyph[3] are so far promising. This could represent a method to solve the processing problem associated with the aforementioned amounts of reference data.

[1] `www.mturk.com`
[2] `http://www.cerberusgame.com`
[3] `www.geoglyph.net`

List of Figures

List of Tables

Previous Publications

[1] Kai Berger, Stephan Meister, Rahul Nair, and Daniel Kondermann. A state of the art report on kinect sensor setups in computer vision. In *German Conference on Pattern Recognition (GCPR), 2013 International Conference on, Workshop on Imaging New Modalities*, 2013.

[2] Stephan Meister. A study on ground truth generation for optical flow. Master's thesis, Heidelberg Collaboratory for Image Processing, University of Heidelberg, 2010.

[3] Stephan Meister and Daniel Kondermann. Real versus realistically rendered scenes for optical flow evaluation. In *Proceedings of 14th ITG Conference on Electronic Media Technology*, 2011. ISBN 978-1-4577-1269-2.

[4] Stephan Meister, Shahram Izadi, Pushmeet Kohli, Martin Hämmerle, Carsten Rother, and Daniel Kondermann. When can we use kinectfusion for ground truth acquisition? In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, Workshops & Tutorials*, 2012.

[5] Stephan Meister, Bernd Jähne, and Daniel Kondermann. Outdoor stereo camera system for the generation of real-world benchmark data sets. *Optical Engineering*, 51, 2012.

[6] Stephan Meister, Rahul Nair, and Daniel Kondermann. Simulation of time-of-flight sensors using global illumination. In *Vision, Modeling and Visualization (VMV), 2013 International Workshop on*, 2013.

[7] Rahul Nair, Stephan Meister, Martin Lambers, Michael Balda, Hannes Hoffmann, Andreas Kolb, Daniel Kondermann, and Bernd Jähne. *Ground Truth for Evaluating Time of Flight Imaging*, chapter 4. LNCS. Springer, 2013.

[8] Sven Wanner, Stephan Meister, and Bastian Goldluecke. Datasets and benchmarks for densely sampled 4d light fields. In *Vision, Modeling and Visualization (VMV), 2013 International Workshop on*, 2013.

Bibliography

[9] S. Abraham and T. Hau. Towards autonomous highprecision calibration of digital cameras. In *Videometrics V, Proceedings of SPIE Annual Meeting*, volume 3174, pages 82–93. Citeseer, 1997. 34, 35, 65

[10] E.H. Adelson and J.R. Bergen. The plenoptic function and the elements of early vision. *Computational models of visual processing*, 1, 1991. 97

[11] A. Aydemir, D. Henell, P. Jensfelt, and R. Shilkrot. Kinect@ home: Crowdsourcing a large 3d dataset of real environments. In *2012 AAAI Spring Symposium Series*, 2012. 101

[12] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 92(1):1–31, 2011. 17, 27, 28, 54, 80

[13] Alexander Barth, Jan Siegemund, Annemarie Meissner, Uwe Franke, and Wolfgang Förstner. Probabilistic multi-class scene flow segmentation for traffic scenes. In M. Goesele et al., editor, *DAGM*, LNCS 6376, pages 513–522. Springer Heidelberg, 2010. 60

[14] Florian Becker, Frank Lenzen, Jörg Hendrik Kappes, and Christoph Schnörr. Variational Recursive Joint Estimation of Dense Scene Structure and Camera Motion from Monocular High Speed Traffic Sequences. In *Proceedings of ICCV 2011*, 2011. to appear. 66, 69

[15] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *Visualization and Computer Graphics, IEEE Transactions on*, 5(4):349–359, 1999. 50

[16] Paul J Besl and Neil D McKay. A method for registration of 3-d shapes. *IEEE Transactions on pattern analysis and machine intelligence*, 14(2): 239–256, 1992. 42

[17] Anak Bhandari, Børge Hamre, Øvynd Frette, Lu Zhao, Jakob J Stamnes, and Morten Kildemo. Bidirectional reflectance distribution function of spectralon white reflectance standard illuminated by incoherent unpolarized and plane-polarized light. *Applied Optics*, 50(16):2431–2442, 2011. 121

[18] M.J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *Proceedings of the Fourth International Conference on Computer Vision (ICCV'93)*, pages 231–236, 1993. 95

[19] Blender. Blender Foundation, 1994–2013. `www.blender.org`. 36, 98

[20] James F. Blinn. Models of light reflection for computer synthesized pictures. *Proceedings of the 4th annual conference on Computer graphics and interactive techniques*, 1977. 90

[21] Wolfgang Boehler, M Bordas Vicent, and Andreas Marbs. Investigating laser scanner accuracy. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(Part 5):696–701, 2003. 38

[22] Andrea Bonarini, Wolfram Burgard, Giulio Fontana, Matteo Matteucci, Domenico Giorgio Sorrenti, and Juan Domingo Tardos. Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets. In *In proceedings of IROS'06 Workshop on Benchmarks in Robotics Research*, 2006. URL `http://www.robot.uji.es/EURON/en/iros06.htm`. 29

[23] Jean-Yves Bouguet. Camera calibration toolbox for matlab. 2004. 34

[24] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 34, 137

[25] Duane C Brown. Decentering distortion of lenses. *Photometric Engineering*, 32(3):444–462, 1966. 33

[26] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *Computer Vision–ECCV 2012*, pages 611–625. Springer, 2012. 80

[27] B Buttgen, M'H-A El Mechat, Felix Lustenberger, and Peter Seitz. Pseudonoise optical modulation for real-time 3-d imaging with minimum interference. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 54(10):2109–2119, 2007. 114

[28] M. M. Bronstein C. Strecha, A. M. Bronstein and Pascal Fua. LDAHash: Improved matching with smaller descriptors. In *EPFL-REPORT-152487*, 2010. 28

[29] Edwin Catmull. A subdivision algorithm for computer display of curved surfaces. Technical report, DTIC Document, 1974. 102

[30] Cenobi, blendswap community. Kitchen scene. Distributed under the Creative Commons Attribution-ShareAlike 3.0 license `http://www.blendswap.com/blends/view/62683`, accessed 15. Okt. 2013, 2013. 123, 152

[31] Simone Ceriani, Giulio Fontana, Alessandro Giusti, Daniele Marzorati, Matteo Matteucci, Davide Migliore, Davide Rizzi, Domenico G. Sorrenti, and Pierluigi Taddei. Rawseeds ground truth collection systems for indoor self-localization and mapping. *Autonomous Robots*, 27(4):353–371, 2009. doi: 10.1007/s10514-009-9156-5. URL `http://www.springerlink.com/content/k924032g72818h53/`. 29

[32] Andrew I Comport, Ezio Malis, and Patrick Rives. Accurate quadrifocal tracking for robust 3d visual odometry. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 40–45. IEEE, 2007. 61

[33] Richard M Conroy, Adrian A Dorrington, Michael J Cree, Rainer Künnemeyer, and Brian Gabbitas. Shape and deformation measurement using heterodyne range imaging technology. In *Proceedings of the 12th Asia-Pacific Conference on Non-Destructive*, 2006. 114

[34] Robert L Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics (TOG)*, 1(1):7–24, 1982. 124, 135

[35] P Courtney and NA Thacker. Performance characterisation in computer vision: The role of statistics in testing and design. *Imaging and Vision Systems: Theory, Assessment and Applications. NOVA Science Books*, 2001. 25

[36] Crytek GmbH (Developer), Electronic Arts Corporation (Publisher). Crysis (computer game), 2007. 102

[37] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 303–312. ACM, 1996. 40

[38] Maximilian Diebold and Bastian Goldluecke. Epipolar plane image refocusing for improved depth estimation and occlusion handling. In *Vision, Modeling and Visualization (VMV), 2013 International Workshop on*, 2013. 97

[39] Axel Donath and Daniel Kondermann. Is crowdsourcing for optical flow ground truth generation feasible? In *Computer Vision Systems*, pages 193–202. Springer, 2013. 71, 80, 101, 172

[40] A.A. Dorrington, J.P. Godbaz, M.J. Cree, A.D. Payne, and L.V. Streeter. Separating true range measurements from multi-path and scattering interference in commercial range cameras. In *IS&T/SPIE Electronic Imaging*, pages 786404–786404. International Society for Optics and Photonics, 2011. 121

[41] Michael Erz. *Charakterisierung von Laufzeitkamerasystemen für Lumineszenzlebensdauermessungen*. PhD thesis, University of Heidelberg, 2011. 64

[42] Michael Erz and Bernd Jähne. Radiometric and spectrometric calibrations, and distance noise measurement of tof cameras. *Dynamic 3D Imaging*, pages 28–41, 2009. 117, 120

[43] Michael Erz and Bernd Jähne. Optimierte kameraauswahl für maschinelles sehen durch standardisierte charakterisierung der bildgebenden systeme. In *Forum Bildverarbeitung*, page 155. KIT Scientific Publishing, 2010. 64

[44] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, June 2010. 17

[45] D. Falie and V. Buzuloiu. Distance errors correction for the time of flight (tof) cameras. In *Imaging Systems and Techniques, 2008. IST 2008. IEEE International Workshop on*, pages 123–126. IEEE, 2008. 120, 121

[46] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007. 17, 60

[47] Wolfgang Förstner. 10 pros and cons against performance characterization of vision algorithms. In *Proceedings of the ECCV Workshop on Performance Characteristics of Vision Algorithms*, pages 13–29, 1996. 20, 22, 25

[48] Uwe Franke. The quest for robustness and accuracy - mission impossible? In *Unsolved Problems in Pattern Recognition and Computer Vision, DAGM Workshop on, 35th German Conference on Pattern Recognition (GCPR 2013)*, 2013. URL `http://www.mpi-inf.mpg.de/conferences/up2013/`. 58, 80, 96

[49] Free Software Foundation, Inc. Gnu general public license, version 3. `http://www.gnu.org/licenses/gpl-3.0.html`, 2007. 130

[50] Jannik Fritsch, Tobias Kuehnl, and Andreas Geiger. A new performance measure and evaluation benchmark for road detection algorithms. In *International Conference on Intelligent Transportation Systems (ITSC)*, 2013. 28, 38, 61

[51] S. Fuchs. Multipath interference compensation in time-of-flight camera images. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 3583–3586. IEEE, 2010. 121

[52] Dorian Garcia, Jean-José Orteu, Laurent Robert, Bertrand Wattrisse, Florian Bugarin, et al. A generic synthetic image generator package for the evaluation of 3d digital image correlation and other computer vision-based measurement techniques. *PhotoMechanics 2013*, 2013. 81

[53] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR2012*, Providence, USA, June 2012. 17, 28, 38, 61, 62, 63

[54] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 28, 38, 61, 62, 63

[55] C.M. Goral, K.E. Torrance, D.P. Greenberg, and B. Battaile. Modeling the interaction of light between diffuse surfaces. In *ACM SIGGRAPH Computer Graphics*, volume 18, pages 213–222. ACM, 1984. 91

[56] Howard R Gordon. Interpretation of airborne oceanic lidar: effects of multiple scattering. *Applied Optics*, 21(16):2996–3001, 1982. 121

[57] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54. ACM, 1996. 97

[58] Jens-Malte Gottfried and Daniel Kondermann. Charon suite software framework. *Image Processing Online (IPOL)*, 2012. 138

[59] Frank Gray. Pulse code communication, March 17 1953. US Patent 2,632,058. 37

[60] Jean-Philippe Grimaldi, Jean-Francois Romang, Tom Bech, Peter Bienkowski, and David Bucciarelli et al. Luxrender: Gpl physically based renderer, 2010–2013. URL `http://www.luxrender.net`. 128, 148

[61] The HDF Group. Hierarchical data format. `http://www.hdfgroup.org/`, 2013. 24

[62] Gaël Guennebaud and Markus Gross. Algebraic point set surfaces. In *ACM Transactions on Graphics (TOG)*, volume 26, page 23. ACM, 2007. 56

[63] Ralf Haeusler and Reinhard Klette. Benchmarking stereo data (not the matching algorithms). In *Pattern Recognition*, pages 383–392. Springer, 2010. 25

[64] Ralf Haeusler and Daniel Kondermann. Synthesizing real world stereo challenges. In *Pattern Recognition*, pages 164–173. Springer, 2013. 80

[65] Vladimir Haltakov, Christian Unger, and Slobodan Ilic. Framework for generation of synthetic ground truth data for driver assistance applications. In *GCPR*, September 2013. 81

[66] Ghassan Hamarneh and Preet Jassi. Vascusynth: Simulating vascular trees for generating volumetric image data with ground truth segmentation and tree analysis. *Computerized Medical Imaging and Graphics*, 34(8):605–616, 2010. doi: 10.1016/j.compmedimag.2010.06.002. 81

[67] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. 34

[68] H. Haussecker and H. Spies. Motion. In B. Jähne, H. Haussecker, and P. Geissler, editors, *Handbook of Computer Vision and Applications*, volume 2, chapter 13. Academic Press, 1999. 91

[69] D. Heeger. Model for the extraction of image flow. *Journal of the Optical Society of America*, 4(8):1455–1471, 1987. 80

[70] Peter Henry, Dieter Fox, Achintya Bhowmik, and Rajiv Mongia. Patch Volumes: Segmentation-based Consistens Mapping with RGB-D Cameras. In *3D Vision 2013 (3DV), International Conference on*, 2013. 41

[71] Francisco Heredia and Raphael Favier. Point cloud library developers blog, kinfu large scale(june 18, 2012). http://www.pointclouds.org/blog/srcs/, 2012. 41

[72] Simon Hermann and Reinhard Klette. Hierarchical scan-line dynamic programming for optical flow using semi-global matching. In *Computer Vision-ACCV 2012 Workshops*, pages 556–567. Springer, 2013. 70

[73] Simon Hermann and Reinhard Klette. Iterative semi-global matching for robust driver assistance systems. In *Computer Vision–ACCV 2012*, pages 465–478. Springer, 2013. 70

[74] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):328–341, 2007. 68, 70

[75] Heiko Hirschmüller. Stereo processing by semiglobal matching and mutual information. *IEEE PAMI*, 30:328–341, 2008. ISSN 0162-8828. doi: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2007.1166. 53

[76] Stephan Hussmann, Alexander Hermanski, and Torsten Edeler. Real-time motion artifact suppression in tof camera systems. *Instrumentation and Measurement, IEEE Transactions on*, 60(5):1682–1690, 2011. 121

[77] D.S. Immel, M.F. Cohen, and D.P. Greenberg. A radiosity method for non-diffuse environments. In *ACM SIGGRAPH Computer Graphics*, volume 20, pages 133–142. ACM, 1986. 123

[78] Cesar Isaza, Joaquin Salas, and Bogdan Raducanu. Rendering ground truth data sets to detect shadows cast by static objects in outdoors. *Multimedia Tools and Applications*, pages 1–15, 2013. 81

[79] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. 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*, UIST '11, pages 559–568, 2011. ISBN 9781450307161. doi: 10.1145/2047196.2047270. 38, 40

[80] B. Jähne. *Handbook of Computer Vision and Application*, volume 2, chapter 2. Academic Press, 1999. 125

[81] Stjepan Jecić and Nenad Drvar. The assessment of structured light and laser scanning methods in 3d shape measurements. In *Proceedings of the 4th International Congress of Croatian Society of Mechanics*, pages 237–244, 2003. 37, 38, 39

[82] H.W. Jensen. Global illumination using photon maps. *Rendering Techniques*, 96:21–30, 1996. 128

[83] D. Jiménez, D. Pizarro, M. Mazo, and S. Palazuelos. Modelling and correction of multipath interference in time of flight cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 893–900. IEEE, 2012. 121

[84] J.T. Kajiya. The rendering equation. *ACM SIGGRAPH Computer Graphics*, 20(4):143–150, 1986. 123

[85] Wilfried Karel, Sajid Ghuffar, and Norbert Pfeifer. Modelling and Compensating Internal Light Scattering in Time of Flight Range Cameras. *The Photogrammetric Record*, 27(138), 2012. ISSN 1477-9730. 121

[86] M. Keller and A. Kolb. Real-time simulation of time-of-flight sensors. *Simulation Modelling Practice and Theory*, 17(5):967–978, 2009. 121, 138

[87] M. Keller, J. Orthmann, A. Kolb, and V. Peters. A simulation framework for time-of-flight sensors. In *Signals, Circuits and Systems, 2007. ISSCS 2007. International Symposium on*, volume 1, pages 1–4. IEEE, 2007. 121

[88] 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 *3D Vision 2013 (3DV), International Conference on*, 2013. 41

[89] K. Khoshelham. Accuracy analysis of kinect depth data. In *ISPRS Workshop Laser Scanning*, volume 38, page 1, 2011. 43

[90] Reinhard Klette, Norbert Kruger, Tobi Vaudrey, Karl Pauwels, Marc Van Hulle, Sandino Morales, Farid I Kandil, Ralf Haeusler, Nicolas Pugeault, Clemens Rabe, et al. Performance of correspondence algorithms in vision-based driver assistance using an online image sequence database. *Vehicular Technology, IEEE Transactions on*, 60(5):2012–2026, 2011. 60, 62

[91] Claude Knaus and Matthias Zwicker. Progressive photon mapping: A probabilistic approach. *ACM Transactions on Graphics (TOG)*, 30(3):25, 2011. 134

[92] Daniel Kondermann. Ground truth design principles: an overview. In *Proceedings of the International Workshop on Video and Image Ground Truth in Computer Vision Applications*, page 5. ACM, 2013. 16

[93] Daniel Kondermann, Steffen Abraham, Gabriel Brostow, Wolfgang Förstner, Stefan Gehrig, Atsushi Imiya, Bernd Jähne, Felix Klose, Marcus Magnor, Helmut Mayer, Rudolf Mester, Tomas Pajdla, Ralf Reulke, and Henning Zimmer. On performance analysis of optical flow algorithms. In Frank Dellaert, Jan-Michael Frahm, Marc Pollefeys, Laura Leal-Taixé, and Bodo Rosenhahn,

editors, *Outdoor and Large-Scale Real-World Scene Analysis*, volume 7474 of *Lecture Notes in Computer Science*, pages 329–355. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-34090-1. doi: 10.1007/978-3-642-34091-8_15. URL http://dx.doi.org/10.1007/978-3-642-34091-8_15. 25

[94] Daniel Kondermann, Anita Sellent, Bernd Jähne, and Jochen Wingbermühle. Hci robust vision challenge. http://hci.iwr.uni-heidelberg.de/Static/challenge2012/, 2012. 69

[95] K Kraus. Die katasterphotogrammetrie im praktischen einsatz. *F., Ackermann (Ed.), Numerische Photogrammetrie, Sammlung Wichmann, Neue Folge. Wichmann Karlsruhe*, 1973. 15

[96] Kenneth Edward Kunkel and JA Weinman. Monte carlo analysis of multiply scattered lidar returns. *Journal of Atmospheric Sciences*, 33:1772–1781, 1976. 122

[97] Eric P Lafortune and Yves D Willems. Bi-directional path tracing. In *Proceedings of CompuGraphics*, volume 93, pages 145–153, 1993. 147

[98] Seungkyu Lee, Byongmin Kang, James DK Kim, and Chang Yeong Kim. Motion blur-free time-of-flight range sensor. In *Proceedings of the SPIE Electronic Imaging*, 2012. 121

[99] Damien Lefloch, Thomas Hoegg, and Andreas Kolb. Real-time motion artifacts compensation of tof sensors data on gpu. *Proc. SPIE 8738, Three-Dimensional Imaging, Visualization and Display*, 2013. doi: 10.1117/12.2018140. 115, 121

[100] F. Lenzen, H. Schaefer, and C. Garbe. Denoising time-of-flight data with adaptive total variation. In *Advances in Visual Computing*, volume 6938 of *LNCS*, pages 337–346. Springer, 2011. ISBN 978-3-642-24027-0. 120

[101] M. Levoy and P. Hanrahan. Light field rendering. In *Proc. SIGGRAPH*, pages 31–42, 1996. 97

[102] Colin Levy (Director), Ton Roosendaal (Producer), and Esther Wouda(Writer). Sintel movie, 2010. URL http://www.sintel.org/. copyright Blender Foundation, durian.blender.org. 81, 85

[103] Derek D Lichti. Error modelling, calibration and analysis of an am–cw terrestrial laser scanner system. *ISPRS Journal of Photogrammetry and Remote Sensing*, 61(5):307–324, 2007. 38

[104] M. Lindner and A. Kolb. Calibration of the intensity-related distance error of the pmd tof-camera. In *Proc. SPIE, Intelligent Robots and Computer Vision*, volume 6764, page 67640W, 2007. 120

[105] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM Siggraph Computer Graphics*, volume 21, pages 163–169. ACM, 1987. 40

[106] O. Mac Aodha, G.J. Brostow, and M. Pollefeys. Segmenting video into classes of algorithm-suitability. In *CVPR2010*, pages 1054–1061, 2010. 80

[107] Oisin Mac Aodha, Neill DF Campbell, Arun Nair, and Gabriel J Brostow. Patch based synthesis for single depth image super-resolution. In *12th European Conference on Computer Vision*. ECCV, 2012. 120

[108] Mark Maimone and Steven A Shafer. A taxonomy for stereo computer vision experiments. In *ECCV workshop on performance characteristics of vision algorithms*, pages 59–79. Citeseer, 1996. 25

[109] Sarah Martull, Martin Peris, and Kazuhiro Fukui. Realistic cg stereo image dataset with ground truth disparity maps. In *In proceedings of The 3rd International Workshop on Benchmark Test Schemes for AR/MR Geometric Registration and Tracking Method (TrakMark2012), Pattern Recognition (ICPR), 2012 21st International Conference on*, 2012. 29, 81

[110] B. McCane, K. Novins, D. Crannitch, and B. Galvin. On benchmarking optical flow. *Computer Vision and Image Understanding*, 84(1):126–143, 2001. 80

[111] Shree K. Nayar Michael Oren. Generalization of lambert's reflectance model. *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 1994. 90, 124, 135

[112] D Mitchell. Ray tracing and irregularities of distribution. In *Proceedings of the Third Eurographics Workshop on Rendering*, pages 61–69, 1992. 126

[113] Don P Mitchell and Arun N Netravali. Reconstruction filters in computer-graphics. In *ACM Siggraph Computer Graphics*, volume 22, pages 221–228. ACM, 1988. 127

[114] Sandino Morales and Reinhard Klette. Ground truth evaluation of stereo algorithms for real world applications. In *Computer Vision–ACCV 2010 Workshops*, pages 152–162. Springer, 2011. 28

[115] Yuichi Nakamura, Tomohiko Matsuura, Kiyohide Satoh, and Yuichi Ohta. Occlusion detectable stereo-occlusion patterns in camera matrix. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on*, pages 371–378. IEEE, 1996. 28

[116] Andreas Nüchter and Kai Lingemann. Robotic 3d scan repository. `http://kos.informatik.uni-osnabrueck.de/3Dscans/`, 2011. Institute of Computer Science, Knowledge-Based Systems Research Group, University of Osnabrueck. 59

[117] Carlos E. Ochoa, Myriam B.C. Aries, and Jan L.M. Hensen. State of the art in lighting simulation for building science: a literature review. *Journal of Building Performance Simulation*, 5(4):209–233, 2012. doi: 10.1080/19401493.2011.558211. URL `http://www.tandfonline.com/doi/abs/10.1080/19401493.2011.558211`. 86

[118] Gaurav Pandey, James R. McBride, and Ryan M. Eustice. Ford campus vision and lidar data set. *International Journal of Robotics Research*, 30 (13):1543–1552, November 2011. 61

[119] Martin Peris, Sara Martull, Atsuto Maki, Yasuhiro Ohkawa, and Kazuhiro Fukui. Towards a simulation driven stereo vision system. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1038–1042. IEEE, 2012. 29, 81

[120] M. Pharr and G. Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2010. 86, 122, 126, 128, 148

[121] Bui Tuong Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975. 90

[122] M. Plaue. Analysis of the pmd imaging system. Technical report, Interdisciplinary Center for Scientific Computing,University of Heidelberg, 2006. 112, 113, 120, 149

[123] Marcel Prastawa, Elizabeth Bullitt, and Guido Gerig. Synthetic ground truth for validation of brain tumor mri segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2005*, pages 26–33. Springer, 2005. 81

[124] Holger Rapp. Experimental and theoretical investigation of correlating TOF-Camera systems. Master's thesis, University of Heidelberg, `http://katalog.ub.uni-heidelberg.de/titel/66407691`, 2007. 149

[125] Holger Rapp, M. Frank, Fred Hamprecht, and Bernd Jähne. A theoretical and experimental investigation of the systematic errors and statistical uncertainties of time-of-flight-cameras. *International Journal of Intelligent Systems Technologies and Applications*, 5(3):402–413, 2008. 113, 117, 120

[126] Fabio Remondino and Clive Fraser. Digital camera calibration methods: considerations and comparisons. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5):266–272, 2006. 34

[127] Ralf Reulke, Andreas Luber, Mathias Haberjahn, and Björn Piltz. Validierung von mobilen stereokamerasystemen in einem 3d-testfeld. *Presented at: 3D-NordOst*, 10:11, 2009. 63

[128] M. Reynolds, J. Dobos, L. Peel, T. Weyrich, and G.J. Brostow. Capturing time-of-flight data with confidence. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 945–952. IEEE, 2011. 38

[129] Seyed Hamid Rezatofighi, W Pitkeathly, Stephen Gould, Richard Hartley, Katarina Mele, W Hughes, and J Burchfield. A framework for generating realistic synthetic sequences of total internal reflection fluorescence microscopy images. In *Proc. ISBI*, 2013. 81

[130] Henry Roth and Marsette Vona. Moving volume kinectfusion. In *British Machine Vision Conf.(BMVC),(Surrey, UK)*, 2012. 41

[131] Peter Rubin. Xbox one revealed. *Wired, http://www.wired.com/gadgetlab/2013/05/xbox-one/*, 2013. 120

[132] Roland Ruiters and Reinhard Klein. A compact and editable representation for measured brdfs. Technical Report CG-2010-1, University of Bonn, December 2010. 166

[133] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47:7–42, 2001. 17, 22, 28

[134] Konstantin Schauwecker, Sandino Morales, Simon Hermann, and Reinhard Klette. A comparative study of stereo-matching algorithms for road-modeling in the presence of windscreen wipers. In *Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IEEE IV '11)*, pages 7–12, June 2011. doi: 10.1109/IVS.2011.5940392. 60

[135] Martin Schmidt. *Spatiotemporal Analysis of Range Imagery*. PhD thesis, University of Heidelberg, 2008. 120

[136] Mirko Schmidt. *Analysis, Modeling and Dynamic Optimization of 3D Time-of-Flight Imaging Systems*. PhD thesis, University of Heidelberg, 2011. 116, 117, 120, 121, 138, 158

[137] Mirko Schmidt and Bernd Jähne. A physical model of time-of-flight 3d imaging systems, including suppression of ambient light. In *Dynamic 3D Imaging*, volume 5742 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin / Heidelberg, 2009. ISBN 978-3-642-03777-1. URL `http://dx.doi.org/10.1007/978-3-642-03778-8_1`. 120

[138] H Schoner, B Moser, Adrian A Dorrington, Andrew D Payne, Michael J Cree, B Heise, and F Bauer. A clustering based denoising technique for range images of time of flight cameras. In *Computational Intelligence for Modelling Control & Automation, 2008 International Conference on*, pages 999–1004. IEEE, 2008. 120

[139] Rudolf Schwarte, Horst-Guenther Heinol, Zhanping Xu, and Klaus Hartmann. New active 3d vision system based on rf-modulation interferometry of incoherent light. In *Photonics East'95*, pages 126–134. International Society for Optics and Photonics, 1995. 115

[140] Anita Sellent, Daniel Kondermann, Stephan Simon, Simon Baker, Goksel Dedeoglu, Oliver Erdler, Phil Parsonage, Christoph Unger, and Wolfgang Niehsen. Optical flow estimation versus motion estimation. Technical report, Universitätsbibliothek der Universität Heidelberg, 2012. 96

[141] Peter Shirley, R Keith Morley, Peter-Pike Sloan, and Chris Wyman. Basics of physically-based rendering. In *SIGGRAPH Asia 2012 Courses*, page 2. ACM, 2012. 86

[142] R Simpson, J Cullip, and J Revell. The cheddar gorge data set. Technical report, Technical Report, Advanced Technology Centre, BAE Systems (Operations) Ltd, Sowerby Building, FPC 267, PO Box 5, Filton, Bristol, BS34 7QW, UK, 2010. http://www. openslam. org, 2011. 61, 62

[143] Adam Smith, James Skorupski, and James Davis. Transient rendering. Technical report, School of Engineering, University of California, 2007. 121

[144] Alvy Ray Smith. A pixel is not a little square, a pixel is not a little square, a pixel is not a little square! (and a voxel is not a little cube). Technical report, Microsoft Cooperation, 1995. 124

[145] Mike Smith, Ian Baldwin, Winston Churchill, Rohan Paul, and Paul Newman. The new college vision and laser data set. *The International Journal of Robotics Research*, 28(5):595–599, 2009. 61

[146] Jörg Stückler and Sven Behnke. Multi-resolution surfel maps for efficient dense 3d modeling and tracking. *Journal of Visual Communication and Image Representation*, 2013. 41

[147] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the IEEE Int. Conf. on Intelligent Robot Systems (IROS)*, pages 573–580, 2012. 29

[148] Deqing Sun, Stefan Roth, J. P. Lewis, and Michael J. Black. Learning optical flow. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 83–97, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-88689-1. doi: http://dx.doi.org/10.1007/978-3-540-88690-7_7. 60

[149] Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2432–2439. IEEE, 2010. 68, 70

[150] John Robert Taylor. *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements*, pages 128–129. University Science Books, 1999. 17

[151] Neil A. Thacker, Adrian F. Clark, John L. Barron, J. Ross Beveridge, Patrick Courtneye, William R. Crum, Visvanathan Ramesh, and Christine Clark. Performance characterization in computer vision: A guide to best practices. *Computer Vision and Image Understanding*, 109(3):305–334, March 2008. 26

[152] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: a large database for non-parametric object and scene recognition. *IEEE PAMI*, 30 (11):1958–1970, November 2008. 60

[153] R.Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, pp. 364-374.*, 1986. 90

[154] T. Vaudrey, C. Rabe, R. Klette, and J. Milburn. Differences between stereo and motion behaviour on synthetic and real-world stereo sequences. In *Image and Vision Computing New Zealand, 2008. 23rd International Conference*, pages 1–6, 2008. ISBN 978-1-4244-3780-1. 80, 88

[155] Eric Veach and Leonidas Guibas. Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques*, pages 145–167. Springer, 1995. 147

[156] Eric Veach and Leonidas J. Guibas. Metropolis light transport. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 65–76, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. doi: http://doi.acm.org/10.1145/258734.258775. 126

[157] S. Wanner and B. Goldluecke. Variational light field analysis for disparity estimation and super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013. 97

[158] S. Wanner, J. Fehr, and B. Jähne. Generating EPI representations of 4D light fields with a single lens focused plenoptic camera. *Advances in Visual Computing*, pages 90–101, 2011. 97

[159] Thomas Whelan, Michael Kaess, Maurice Fallon, Hordur Johannsson, John Leonard, and John McDonald. Kintinuous: Spatially extended kinectfusion. Technical Report MIT-CSAIL-TR-2012-020, CSAIL Technical Reports, 2012. URL `http://hdl.handle.net/1721.1/71756`. 41

[160] Bennett Wilburn, Neel Joshi, Vaibhav Vaish, Eino-Ville Talvala, Emilio Antunez, Adam Barth, Andrew Adams, Mark Horowitz, and Marc Levoy. High performance imaging using large camera arrays. *ACM Transactions on Graphics*, 24:765–776, July 2005. 98

[161] Alexander Wilkie, Andrea Weidlich, Marcus Magnor, and Alan Chalmers. Predictive rendering. In *ACM SIGGRAPH ASIA 2009 Courses*, page 12. ACM, 2009. 82, 86

[162] Simon Winkelbach, Sven Molkenstruck, and Friedrich Wahl. Low-cost laser range scanner and fast surface registration approach. *Pattern Recognition*, pages 718–728, 2006. 39

[163] J. Wulff, D. J. Butler, G. B. Stanley, and M. J. Black. Lessons and insights from creating a synthetic optical flow benchmark. In A. Fusiello et al. (Eds.), editor, *ECCV Workshop on Unsolved Problems in Optical Flow and Stereo Estimation*, Part II, LNCS 7584, pages 168–177. Springer-Verlag, oct 2012. 85

[164] Z. Xu, R. Schwarte, H. Heinol, B. Buxbaum, and T. Ringbeck. Smart pixel–photonic mixer device (pmd). In *Proc. Int. Conf. on Mechatron. & Machine Vision*, 1998. 113

[165] C. Yang and G. Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992. 40

[166] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv- l 1 optical flow. In *29th Annual Symposium of the German Association for Pattern Recognition (DAGM'07)*, pages 214–223, 2007. 91

[167] Ming Zeng, Fukai Zhao, Jiaxiang Zheng, and Xinguo Liu. A memory-efficient kinectfusion using octree. In *Computational Visual Media*, pages 234–241. Springer, 2012. 41

[168] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000. 34