Lukas Gruber

Photometric Registration for Augmented Reality in Dynamic Environments

Dissertation

to achieve the university degree of
Doktor der technischen Wissenschaften

submitted to

Graz University of Technology

Supervisor

Univ. Prof. Dr. Dieter Schmalstieg
Institute for Computer Graphics and Vision

Referee

Univ. Prof. Dr. Tobias Höllerer
University of Santa Barbara California

Graz, Austria, March 2015

To Disco

Tages Arbeit! Abends Gäste!
Saure Wochen! Frohe Feste!

"Der Schatzgräber", Johann Wolfgang von Goethe
(1749 - 1832)

Abstract

In Augmented Reality, digital information is overlaid on the real-world, bringing the virtual world and real-world closer together. This new intuitive and natural interface concept enables new solutions for a variety of different applications, such as assisted personal navigation, Augmented Reality gaming or e-commerce. Since Augmented Reality is mainly known as a display technology, the quality of the rendered virtual content matters. Especially for application scenarios with a high demand on rendering quality, e.g., home shopping, convincingly embedding virtual objects into the real-world can improve the Augmented Reality experience of the user. This important key feature is also known as visually coherent rendering, which aims for blending the real-world and the virtual together, so they become indistinguishable from each other. This involves several different techniques, where applying real-world lighting and shadowing to the virtual objects and vise versa is one of them.

This demands measuring the real-world lighting through photometric registration. Common approaches in Augmented Reality so far provided good results, employing special hardware, so called light probes. Although the quality of the results can be satisfying, most light probes have to be inserted into to the scene and hence are visually disturbing and not applicable for consumer-oriented applications. In this thesis, we investigated the design and implementation of an Augmented Reality system which is capable of estimating the real-world lighting without special light probes and employing the estimated lighting in real-time photorealistic Augmented Reality rendering. In particular, the work in this thesis introduces a novel approach for probeless photometric registration by relying only on input from a color and depth camera. Moreover we present various acceleration techniques to achieve real-time performance working in greater mid-sized workspaces.

Kurzfassung

In Augmented Reality werden digitale Informationen über die reale Welt überlagert, womit die virtuelle Welt und die reale Welt näher zusammengebracht werden. Dieses neue intuitive und natürliche Interface-Konzept ermöglicht neue Lösungen für eine Vielzahl verschiedener Anwendungen, wie zum Beispiel Navigation, Augmented Reality Spiele oder E-Commerce. Da Augmented Reality hauptsächlich als Anzeigetechnologie verwendet wird, ist die Qualität der graphischen Inhalte ausschlaggebend. Speziell für Anwendungsszenarien mit einem hohen Anspruch an die Graphikqualität, wie zum Beispiel "Home-Shopping", kann eine überzeugende Einbettung der virtuellen Objekte in die reale Welt die Augmented Reality Erfahrung des Anwenders verbessern. Dieses Schlüsselmerkmal wird auch als visuell kohärentes Rendering bezeichnet, dessen Ziel die Vermischung der realen Welt und der virtuellen Welt ist, so dass sie sich voneinander optisch nicht unterscheiden. Dabei werden verschiedene Techniken benoetigt, wovon eine die Anwendung reeler Beleuchtung und Schattenbildung auf virtuelle Objekte ist.

Dies erfordert die Messung der realen Beleuchtung durch photometrische Registrierung. Allgemeine Ansätze in Augmented Reality erzielten bislang gute Ergebnisse mit speziellen Messobjekten, so genannten "light probes". Obwohl die Qualität der Ergebnisse zufriedenstellend ist, muessen die meisten Messobjekte in die reale Szene platziert werden und sind optisch störend und daher nicht einsetzbar in kommerziellen Anwendungen. In dieser Arbeit untersuchten wir die Konzeption und Umsetzung eines Augmented Reality Systems, welches in der Lage ist die Schätzung der realen Beleuchtung ohne spezielle Messobjekte durchzuführen und weiters das geschätzte Licht für fotorealistische Augmented Reality Graphik einzusetzen. Insbesondere stellt diese Arbeit einen neuartigen Ansatz für photometrische Registrierung ohne specielle Messobjekte dar und beruht auf die alleinige Eingabe von einer Farb- und Tiefenkamera. Darüber hinaus stellen wir verschiedenen Beschleunigungstechniken vor, um Echtzeitperformance in größeren Arbeitsbereichen zu erzielen.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

The text document uploaded to TUGRAZonline is identical to the presented doctoral thesis.

Place	Date	Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Dissertation identisch.

Ort	Datum	Unterschrift

Acknowledgments

I would like to thank everybody without whom this dissertation would not have been possible: First of all, I sincerely thank my teacher and adviser Dieter Schmalstieg for introducing me to the exciting field of Augmented Reality with his lectures, and for supporting, guiding, and pushing me throughout the last years in Graz. I want to thank my referee, Tobias Höllerer, for giving me the opportunity to pursue my research at his lab at the University of California, Santa Barbara. I want to thank all the members and former members of the Institute for Computer Graphics and Vision (ICG). I especially thank Daniel Wagner for being an inspiring research lead and late hours companion at the office. I thank Gerhard Reitmayr for his support during his time as a Professor in Graz. I thank the entire team of the Christian Doppler Laboratory for Handheld Augmented Reality. Each one of them was a great support at all times. I am much obliged to my friend Tobias Langlotz, with whom I shared an office and many other things, which hopefully will stay in the office. I also want to thank Jonathan Ventura for his help and valuable input during his time in Graz. I want to thank Manuela Waldner for sharing deep thoughts, late at night on the bridge, and Albert Walzer for last minute video editing. I appreciate the cooperation with Qualcomm, which helped me to grow into industry. Furthermore, I want to thank the entire Four Eyes research Lab I met in Santa Barbara and especially Pradeep Sen for his support and inspiration regarding my work. I am grateful to the people from Bongfish, and especially Thomas Richter Trummer for his collaboration and inspiration.

I am grateful to my parents who never asked annoying questions, and who supported me in my studies, and life in general. I also want to thank my entire family for supporting me at all times. I want to thank Bri, who made life at Inffeld magic. I especially want to thank Mara, Virginia and HannaH for their love and interruptive behavior, and not always tolerating my working hours. I thank Bea for our long friendship. Last but not least, I want to thank all my friends, Houseverbot, MfM, disko404, and many many more I met in the first rows.

Contents

List of Figures

List of Tables

1 Introduction

Contents

1.1 Augmented Reality

With the development of handheld devices, ubiquitous computing became reality for the masses. Smart phones or tablets can be used now everywhere, providing more flexibility in terms of usage and interaction compared to desktop PCs. Bringing compute power closer to the physical world naturally raises the demand for more intuitive user interfaces. A user should be assisted by some sort of easy-to-operate user interface to solve complex tasks, such as navigation in unfamiliar environments.

Over the last decades, user input technologies advanced from purely text based systems to more interactive systems which support the mouse and other tactile interfaces. System output methods have evolved from simple printouts to colored display monitors, which support graphics-based user interfaces to fully immersive virtual reality environments. While common input and output methods are sufficient for certain tasks, mobile computing demands new interfaces which bridge the gap between humans, computers and the environment.

Augmented Reality is an interface concept which addresses natural human computer interaction. The main idea is that digital information is visualized or embedded into a representation of the physical world. More precisely, information is displayed in a spatial relationship to the real-world. Visualizing the important information in place by solving the connectivity between digital information and the real-world is a particularly complex task, which is often left to the user. In classical map navigation, for example, the user has

to understand the relationship between the drawn information on the map and the real-world by himself. An Augmented Reality driven navigation would overlay the important information onto the user's view of the real-world. Therefore Augmented Reality brings the virtual world and real-world closer together, enabling more intuitive and natural interfaces.

In 1968, Ivan Sutherland [111] built the first Augmented Reality system, already featuring an optical see-through display and six degrees of freedom (6DOF) pose tracking. In 1992, the term Augmented Reality was introduced for the first time by Tom Caudell and David Mizell [13], who proposed a system for overlaying computer processed information in an industrial scenario. Paul Milgram and Fumio Kishino [79] presented in 1994 the Reality-Virtuality Continuum. The Reality-Virtuality Continuum defines the transition from the real to the virtual medium, comprising four major positions: Real Environment, Augmented Reality, Augmented Virtuality and Virtual Reality. Augmented Reality resides closer to the Real Environment, while Augmented Virtuality is located closer to the Virtual Environment. A more specific definition for Augmented Reality, which has been acknowledged for many years in the research community, has been introduced by Ronald Azuma [4] in 1995. Azuma states three conditions Augmented Reality applications must fulfill:

- combining the real and virtual,
- being interactive in real-time, and
- registering information in the real-world in 3D.

From these definitions, we can derive that Augmented Reality systems can become very complex in terms of hardware and software. First, combining the real and the virtual on a common display implies that the real-world has to be sensed and modeled. A very common method today, is video-see through Augmented Reality (e.g., on smart phones), where the real-world is captured with a camera device and displayed on a monitor. Virtual augmentations are then rendered on top of the video feed using computer graphics algorithms. Another method is see-through Augmented Reality (e.g., Microsoft Hololens), in which the real-world is perceived through a transparent medium, on which the augmentations are displayed.

Second, the entire system including hardware and software has to run in real-time to provide the requested level of responsiveness and interaction. This is an especially challenging goal for mobile commodity platforms, which have significantly less computing power than stationary systems. Achieving real-time performance is especially challenging for robust and general registration in 3D and high quality augmentations targeting photo-realism. Furthermore, the efficient processing of multiple sensor data, ranging from video to global positioning system (GPS) data and depth sensor data, is a challenging task itself, also because the energy consumption increases.

With the evolution of mobile hardware such as smart phones or tablets, Augmented Reality has been introduced to the mass market. Simultaneously, the quality of video cameras (more pixels) and display technology (higher resolution) has improved. As a consequence, Augmented Reality games or e-commerce are becoming more popular, increasing the need for better and more convincing graphics in Augmented Reality. Today's

expectations for the quality of graphics in AR are high, since they are usually compared to the quality of Virtual Reality games or even classical computer graphics (CG) in movie productions. In the example of e-commerce (e.g., home shopping), products are presented virtually embedded into the real-world. Obviously, the presentation of the virtual objects plays a substantial role, and convincing graphics can positively influence the shopping experience.

Furthermore, Augmented Reality enables the possibility to place virtual products into the personal environment. A prominent use case is furniture shopping, where the user interactively places new items such as a couch into her own living room. In this scenario, the most important concepts of Augmented Reality become evident. The automated 3D registration of the virtual couch in the real-world environment enables the user to figure out if the couch will fit into the given real-world space. Through an appropriate user interface, the user might be able to move the couch around in space and find a suitable spot. The appearance, including texture and lighting of the virtual couch, is created by the Augmented Reality rendering algorithm. Regarding visual quality, the goal is a photorealistic impression, such that the virtual couch is perceptually plausibly integrated into the real-world. This implies the correct estimation and application of real-world lighting onto the virtual objects. Applying real-world lighting involves modeling various lighting effects, such as shadows from real to virtual and vice verse, but also adding indirect illumination. The process of light estimation is known as photometric registration and contributes to the field of visually coherent rendering, where virtual objects are seamlessly blended into the real-world environment.

In this thesis, we argue that there is a high demand for visually coherent rendering solutions for Augmented Reality, where the application of real-world lighting to virtual objects is an important key feature. That can improve the embedding of virtual objects into the real-world, potentially improving the Augmented Reality experience of the user. Photometric registration is a challenging task, and common approaches in Augmented Reality so far have provided good results, but were bound by many restrictions. Most of the presented Augmented Reality solutions conducted photometric registration by employing special hardware, so called light probes. Although the quality of the results can be satisfying, most light probes have to be inserted into to the scene and hence are visually disturbing and not applicable for consumer-oriented applications like the aforementioned furniture scenario. The challenge attacked in this thesis is to overcome these and many other restrictions, providing a general and robust photometric registration solution for practical Augmented Reality.

1.2 Photometric Registration in Augmented Reality

Visual artists were concerned about the correct handling of light in their paintings over centuries. While the media evolved from paint on canvas to photography and moving pictures, the key question still remained: Where does the light come from? Especially in still life painting, where one goal is the realistic representation of the real-world, artists deeply studied the effects of lighting to create convincing depictions. Today, we use computers to simulate physically correct lighting for image synthesis. In this case, the light sources

are given. The artist has been substituted by an algorithm which automatically estimates where the light is coming from. In computer vision, the reconstruction of the light situation from a single image has been studied extensively in the last decades, mostly to remove lighting effects from the image. Another prominent application field is the intersection of modern computer graphics and film production. Pioneering work in merging real video images with computer generated images was published in the early nineties by Bajura et al. [5] or Fournier et al. [26]. The main distinguishing aspect between Augmented Reality and film applications is that visually coherent renderings must now be generated in real time and registered in the environment with a limited amount of preparation. As already mentioned, realistic light interaction between real and virtual objects is a key factor for photorealistic Augmented Reality. Previously, Augmented Reality systems mostly computed photometric registration using special measurement devices such as light probes, and little effort was focused on light estimation solely from the actual real-world scene. Computing the interaction of light between real and virtual from arbitrary scene geometry requires estimation of real-world lighting (photometric registration), estimation of scene geometry (3D reconstruction) and finally the possible radiance transfer computation between virtual and real objects. Since the first two - real-world lighting and real-world geometry - can change dynamically, and radiance transfer depends on both, all three must be computed in real time.

For practical Augmented Reality systems, we face sensor limitations, which will necessarily introduce errors and imperfections in the 3D reconstruction and photometric registration. Handling such errors robustly limits the choice of techniques. Consequently, existing approaches have usually concentrated on a certain aspect of the problem and made simplifying assumptions for the remaining aspects. Concerning geometry, scenes may be assumed to be small and have simple geometry, so the computational effort can be bounded [45, 51, 61, 85]. This can be combined with an assumption that scenes are static, with the exception of explicitly tracked objects. Therefore, offline 3D reconstruction or manual modeling can deliver largely error-free scene geometry, greatly simplifying photometric registration and radiance transfer computation. Arbitrary online geometric changes, such as a user's moving hand in the scene, are not supported. Concerning photometric registration, one simplification is to assume that real-world lighting is static, so photometric registration can performed in a pre-process [45, 51, 78, 85] and online illumination changes, such as a user casting a shadow on the scene, are not supported. Alternatively, online photometric registration can use an invasive light probe, such as an omni-directional camera or reflective sphere [51, 61, 85] placed in the scene, to replace computing photometric registration by direct measurement. Finally, global illumination effects can be restricted to virtual light sources [Lensing and Broll], avoiding photometric registration altogether.

In particular, the work in this thesis introduces a novel approach for probeless photometric registration by relying only on input from a common RGBD camera to compute both dynamic geometry and the environment light considering occlusions. However, the need for simultaneous reconstruction, photometric registration and radiance transfer computation is not easily accommodated in real time. In particular, robust radiance transfer computation based on dense ray-tracing provides good quality but at high costs, and thus requires optimization. Moreover we target greater mid-sized workspaces such as shown in

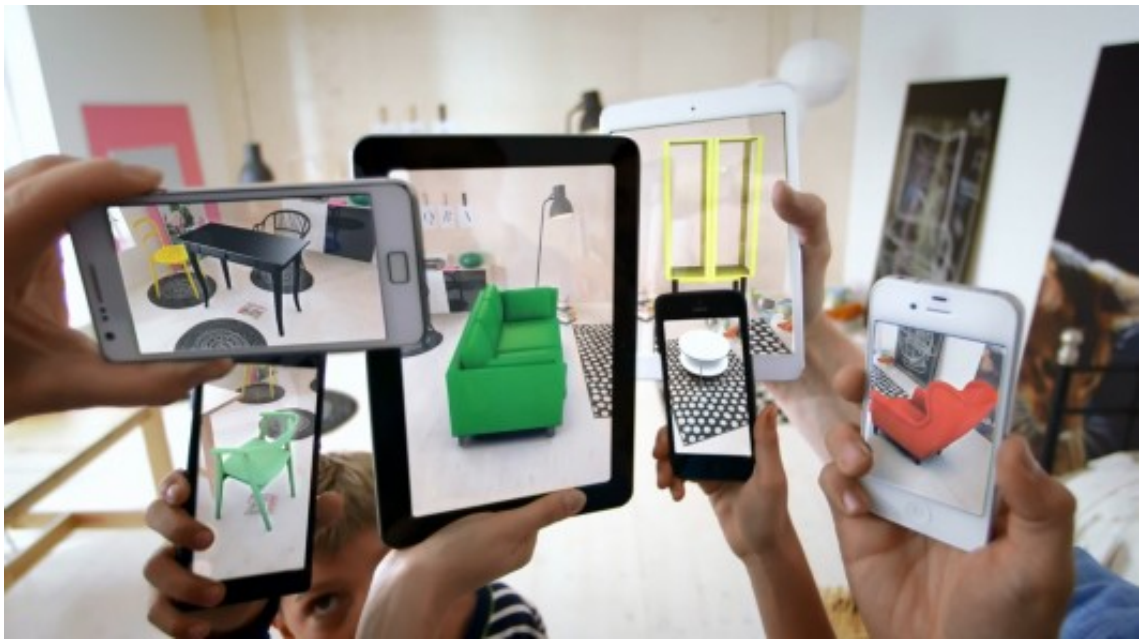


Figure 1.1: Example of an Augmented Reality "home shopping" scenario created by IKEA. Users perceive the real-world captured by a camera through the display of their mobile devices. Over the camera image virtual furniture is rendered. Due to camera tracking technology the virtual object stays in place even if the devices move, creating the illusion of being attached to the real-world. Photometric registration is not handled in this solution. If the lighting condition of the room would change, the appearance of virtual objects would stay the same.

Figure 1.1, which is a "home shopping" example created by IKEA *.

1.3 Goals

This thesis investigates the design and implementation of an Augmented Reality system which is capable of estimating real-world lighting without special light probes and employing the estimated lighting in real-time photorealistic Augmented Reality rendering. We believe that successful Augmented Reality applications are systems which are practical and simple to operate. To build such a system, we define in the following our goals, which are examined throughout the remainder of this document:

- **Probe-less real-world light estimation support:** Real-time probe-less photometric registration from dynamic geometry is a main goal of our work. We aim for a practical solution for photometric registration, which is easy to use and robust against noisy input data, while working in mid-sized environments.

*http://www.gizmag.com/ikea-augmented-reality-catalog-app/28703/

- **Minimal constraints on input data:** In our work, we aim for a solution for photometric registration and rendering which only requires a minimal set of input data. Our system setup is limited to a RGB camera and some sort of depth information. The depth data can originate from an additional device residing next to the RGB camera - e.g., a depth sensor or a stereo camera system - or be computed algorithmically from the RGB stream. The scene geometry is allowed to change dynamically, and no pre-processing steps are required to reconstruct the scene geometry.

- **Amount of realism:** The necessity for photorealistic rendering in Augmented Reality is especially needed where visual aesthetics play a substantial role. The Augmented Reality system has to provide perceptively plausible lighting results, supporting global illumination effects such as direct lighting, shadows, near-field ambient occlusion and indirect illumination in dynamic environments, where the lighting condition and the geometry can change.

- **Agile user interaction:** The processing of our system has to be in real time to allow a high level of interaction for the user. The user has to be able to move the camera and objects in the scene and change the real-world lighting situation dynamically. The speed of the system can also influence the level of light interaction between virtual objects and real-world objects. The user has to be able to interact with virtual objects by producing shadow effects and indirect illumination effects on the virtual objects.

- **High level of automation:** The acquisition of the required input data should be as simple as possible. This leads to the requirement for a single compact hardware setup, which can be operated by a single user, preferably a handheld device. Additional external cameras or artificial props placed within the working space would break this requirement. Also additional user input such as manual registrations are not allowed. In our work, we describe the results of a system where only camera movements and application specific input (e.g, game controls) are necessary for operation.

- **Low costs and high accessibility:** The goal is to build a system from off-the-shelf commodity hardware that is accessible to a broader community, even running on mobile hardware such as tablets.

1.4 Contributions

This thesis contributes to the field of Augmented Reality in the area of photometric registration and visually coherent rendering. It introduces the real-time photometric registration pipeline (see Figure 1.2) for Augmented Reality, which is capable of estimating real-world lighting from previously unknown, unprepared and dynamically changing environments. Furthermore, it presents global illumination based on real-world lighting for Augmented Reality rendering in real-world environments. Our findings support real-world Augmented Reality scenarios that target non-expert users and demand no additional man-

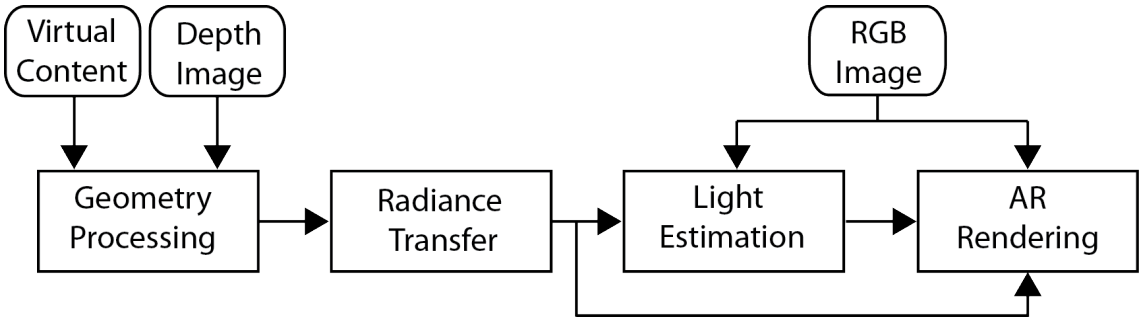


Figure 1.2: General overview of the most important blocks of our photometric registration and rendering pipeline. A key block is the radiance transfer method which computes the possible light transfer between the real geometry and the real and virtual geometry.

ual input, with the exception of operating the camera. The contributions in particular are:

1. a hybrid geometry reconstruction pipeline capable of handling fast, dynamically changing geometry,

2. a real-time photometric registration solution, which handles dynamically changing geometry,

3. methods to improve the robustness of the photometric registration,

4. acceleration methods suitable for photometric registration and rendering,

5. an Augmented Reality global illumination rendering pipeline based on real-world lighting, and

6. a systematic evaluation including the design and implementation of a physical stage set and test environments.

1.4.1 Hybrid Dynamic Geometry Reconstruction Pipeline

Reconstructing the real-world geometry is a central part of our photometric registration and rendering pipeline. Geometry information about the real-world serves as input for computing the possible light transfer of the scene, which we compute in the radiance transfer step. In this thesis, we present a hybrid scene geometry representation which captures changing geometry at the speed of the depth sensor and keeps reconstructed geometry in memory for further global illumination rendering. We integrated the geometry reconstruction into our system [37] and describe the entire evolution of our geometry processing methods in Chapter 4.

1.4.2 Real-time Photometric Registration in Dynamic Environments

Photometric registration means measuring real-world lighting, which is needed for visually coherent rendering in Augmented Reality. Commonly, photometric registration in Augmented Reality has been solved through sensing artificial light probes, such as mirror balls, or by using additional cameras with fish-eye lenses. Although these methods can provide high quality results, they have major drawbacks. For example, using additional cameras increases the cost of the Augmented Reality system, and passive light probes such as mirror balls are not always desired to be visible in the final Augmented Reality view. Such limitations can lower the user experience. In this thesis, we present a novel, non-invasive photometric registration pipeline, which is designed to work in dynamic environments and requires only the minimum amount of user input. Based on our geometry reconstruction, we recover the real-world diffuse environment lighting from dynamically changing scene geometry and support occlusion for every single frame. We integrated our pipeline in a single camera system [36] and multi camera setup [49]. Details on the photometric registration pipeline is covered in Chapter 2.2.

1.4.3 Robust photometric registration in dynamic environments

Photometric registration from unknown environments naturally depends on the quality of the input data. The raw primary input data are the RGB and depth camera streams, which suffer from noise. Moreover, the real-world geometry is not always ideal for performing light estimation. The ideal surface for estimating the environment illumination would be a sphere, which provides surface normal vectors pointing in all possible directions. Unfortunately, such a uniform distribution of surface normal vectors is not always naturally given in the real-world. In practice, the scene often consists of a large planar surface (e.g., a table) with a single dominant surface normal vector. Naturally, samples taken from this surface will have a large impact on the final estimation. Other normal directions will not be sufficiently represented. For example, samples taken from smaller objects in the scene, which have a smaller pixel coverage in the video frame, will not yield a comparable number of samples. In this thesis, we discuss several methods how to improve the robustness and quality of the light estimation. We smooth the input image using a bilateral noise filter, canceling out noise, high frequency textures and small specular highlights. We use object space filtering on the reconstruction over time to improve the geometry and the estimated surface normal vectors. We discard certain samples from the light estimation, which are in areas which suffer from too much occlusion. We analyze the surface vector normal distribution of the reconstructed geometry and weight the samples according to the deviation from the normal distribution. To prevent flickering in the photometric registration result, we apply a moving average filter over the light estimation results. In this thesis, we present these robustness improving methods integrated in our photometric registration pipeline. Details on robust photometric registration are covered in Section 1.4.3.

1.4.4 Accelerating Radiance Transfer Computation

Speed is an important factor in Augmented Reality, and real-time performance is a basic requirement. Augmented Reality systems can get complex very quickly. Each component such as tracking, reconstruction, rendering or interaction can become a serious performance bottleneck. Although 6DOF pose tracking and reconstruction from small workspaces has been become more reasonable, global illumination is still considered challenging. However, there are several approaches for improving the performance of a system. In general, we distinguish between algorithmic improvements, e.g., ray-tracing approximation in image space and hardware-friendly software design such as GPGPU techniques. In this thesis, we focus on improving the performance of the radiance transfer computation, addressing two problems at the same time: photometric registration and rendering. From the algorithmic point of view, we study several sampling techniques in image or object space and exploit several caching possibilities. These techniques are known from real-time rendering. We were te first to evaluate these techniques for photometric registration in conjunction with rendering in AR. The acceleration techniques have to work on imperfect input data from the geometry reconstruction and be robust against moving cameras, dynamically changing geometry, and real-world lighting. Moreover, we exploit the GPU and implement most steps of our photometric registration and rendering pipeline on the GPU using CUDA and OpenGL GLSL shader languages [1][2][4]. More detailed information is given in Chapter 1.4.4.

1.4.5 Global Illumination in Dynamic Environments for Augmented Reality

After 3D registration, rendering is one important aspect of Augmented Reality. In this thesis, we focus on video see-through systems, where the virtual object is rendered into a RGB camera image stream, which represents the real-world. The overall goal is to achieve visually coherent augmentations, where it is hard to distinguish between virtual and real objects. We find prominent application cases for the demand of this quality of rendering in Augmented Reality advertising or gaming. Convincingly embedding virtual furniture into a real-world living room might be one example. However, visual coherent rendering can have many aspects, and applying real-world lighting to the virtual object is one of them. Another important aspect is the quality of the rendering itself, where photorealism is the higher goal. Photorealistic rendering and compositing in real time is achieved by employing global illumination algorithms in combination with differential rendering [19]. Global illumination can be simply explained as computing the shading for each point on the surface by accounting for direct illumination, but also for occlusions/shadows and indirect illumination which incorporates the surrounding geometry. The real-world is not static, but consists of dynamic geometry and dynamic lighting. People and objects move, and lighting changes, as the sun moves across the sky or dynamic objects cast shadows. Therefore, different from classical computer graphic applications, pre-computation is not always possible. This makes global illumination for Augmented Reality even harder than for virtual reality, where the environment is more predictable and pre-computation more feasible. Furthermore, global illumination also covers the simulation of more complex light interactions such as the simulation of caustics. In this thesis, we investigate real-time global

illumination algorithms for Augmented Reality which work in dynamically changing environments with a work space size exceeding desktop setups. Furthermore, we are discussing in this thesis the confluence of photometric registration and global illumination rendering algorithms. We implemented global illumination, with support for shadowing and indirect lighting in dynamic real-world environments, and discuss the details in Chapter 7.

1.4.6 Systematic Evaluation

The Augmented Reality system built in this thesis allows for the first time to estimate real-world lighting in reasonably-sized workspaces without the usage of artificial light probes. In the course of this thesis, several evaluation metrics were developed to assess the quality of the light estimation and the rendering. The evaluation results give also important insights on probe-less light estimation. For the evaluation in this thesis, we present a systematic evaluation pipeline for performance, visual quality, and light estimation quality [35–37]. Furthermore, we designed and implemented a physical and virtual model of an imaginary urban scene - the 'City of Sights' - [34] that can serve as a backdrop or 'stage' for a variety of Augmented Reality research problems. We argue that the Augmented Reality research community benefits from such a standard model dataset, which can be used for evaluation of topics such as tracking systems, modeling, spatial AR, rendering tests, collaborative Augmented Reality and user interface design. More specifically, we used this model to change the scene properties such as the surface texture, while preserving the scene geometry, to evaluate the photometric registration pipeline. Furthermore, we built dedicated test environments to improve the repeatability of our experiments. We introduce our different evaluation tools and test environments in Chapter 3 and explain our evaluation metrics in the according evaluation section of each chapter.

1.5 Collaboration statement

This work builds on publications which have been conducted in collaboration between various researchers from various institutions. In the following, we list the publications and the collaborators who were involved.

- **Lukas Gruber**, Steffen Gauglitz, Jonatan Ventura, Stefanie Zollmann, Mmanuel Huber, Michael Schlegel, Gudrun Klinker, Dieter Schmalstieg, and Tobias Hoellerer, "'The City of Sights: Design, construction, and measurement of an Augmented Reality stage set,"' in 2010 9th IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2010, pp. 157-163. *The author of the paper was involved through all stages of the work in designing, building and creating the data set. Steffen Gauglitz, Jonathan Ventura and Tobias Hoellerer from UCSB were actively involved in designing and improving the imaginary urban stage set. Stefanie Zollmann from TUG was actively involved in building the physical stage set and creating the video sequences with the Robot Arm. Furthermore we got assistance by Matthias Ruether and Martin Lenz from TUG for operating and calibrating the Robot Arm. Manuel Huber and Michael Schlegel from TUM assisted in creating the calibrated video sequences with the FARO arm and ART tracking system.*

- **Lukas Gruber**, Thomas Richter-Trummer, and Dieter Schmalstieg, "Real-time photometric registration from arbitrary geometry," in 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2012, pp. 119-128.
  The author of the paper is responsible for the design and implementation of the real-time photometric registration and Augmented Reality rendering pipeline on GPU hardware. Thomas Richter-Trummer was actively involved in implementing a first prototype and a reference system for SH rendering and inverse rendering based on a CPU raytracer. Gerhard Reitmayr provided a primary implementation of the 3D scene reconstruction part based on Kinect Fusion.

- Bernhard Kainz, Stefan Hauswiesner, Gerhard Reitmayr, Markus Steinberger, Raphael Grasset, **Lukas Gruber**, Eduardo Veas, Denis Kalkofen, Hartmut Seichter, and Dieter Schmalstieg. "'OmniKinect: Real-Time Dense Volumetric Data Acquisition and Applications"', Proceedings of the 18th ACM Symposium on Virtual Reality Software and Technology, page 25–32. New York, NY, USA, ACM, (2012)
  The author of this thesis was actively involved in designing the lighting environment for the OmniKinect setup and adding photometric registration to the system. The main author of this paper, Bernhard Kainz, designed and implemented the actual OmniKinect setup, consisting of multiple RGBD depth sensors (Kinect). In collaboration with Stefan Hauswiesner, Gerhard Reitmayr and Markus Steinberger, the 3D reconstruction algorithm based on KinectFusion has been extended to support multiple RGBD depth sensors. Raphael Grasset, Eduardo Veas and Denis Kalkofen were contributing by implementing application scenarios.

- **Lukas Gruber**, Pradeep Sen, Tobias Höllerer and Dieter Schmalstieg, "Acceleration methods for radiance transfer in photorealistic augmented reality," Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on , pp.255,256, 1-4 Oct. 2013
  The evaluation of different acceleration methods for radiance transfer computation for photorealistic augmented reality has been conducted by the author of this paper. Pradeep Sen and Tobias Hoellerer from UCSB and Dieter Schmalstieg from TUG were actively involved in designing the problem space and supervision.

- **Lukas Gruber**, Tobias Langlotz, Pradeep Sen, Tobias Höllerer, and Dieter Schmalstieg. Efficient and Robust Radiance Transfer for Probeless Photorealistic Augmented Reality. In Proc. IEEE Virtual Reality 2014, Minnesota, MN, USA, March 2014.
  The author of this paper was responsible for designing, implementing and evaluating an acceleration and caching technique for photometric registration and Augmented Reality rendering. Tobias Langlotz from TUG was involved in the evaluation step. Pradepp Sen and Tobias Höllerer from UCSB and Dieter Schmalstieg from TUG were actively involved in designing the problem space and supervision.

- **Lukas Gruber**, Jonathan Ventura, and Dieter Schmalstieg. Image-Space Illumination for Augmented Reality in Dynamic Environments. In Proc. IEEE Virtual

Reality 2015, Arles, Camarque, France, March 2015.
The author of this paper was responsible for designing, implementing and evaluating a hybrid real-time geometry reconstruction algorithm for fast dynamically changing geometry and a global illumination algorithm for Augmented Reality rendering operating in image-space. Dieter Schmalstieg from TUG and Jonathan Ventura were actively involved in designing the algorithms and supervision.

The ideas and research results in the former listed publications were transformed and extended to several patent applications elaborated in cooperation with our project partner Qualcomm:

- **Lukas Gruber**, Thomas Richter-Trummer, Dieter Schmalstieg, "Photometric registration from arbitrary geometry for augmented reality" US 20130271625 A1
- **Lukas Gruber**, Dieter Schmalstieg, "Efficient Radiance Transfer for Light Estimation"
- **Lukas Gruber**, Jonathan Ventura, Dieter Schmalstieg, "Augmented Reality Lighting with Dynamic Geometry"

2 Related Work

Contents

In 2008, Zhou et al. [122] presented a survey about Augmented Reality trends in research, using a metric based on publications and citations. The survey states that *tracking* was and probably still is the most prominent and substantial problem in AR research, followed by *interaction* and *calibration*. *Rendering* resided on the last position, far behind *visualization*. However, since 2008, the interest in higher quality Augmented Reality rendering has noticeable increased in the research community. This trend might be driven by the growing number of commercial Augmented Reality applications, which have a need for high quality rendering to address the user expectations. Moreover, compute power increased, and the quality of the displays improved too. The focus of Augmented Reality rendering research mostly lies in exploring algorithms which improve the visual coherence between virtual content and real content. The most challenging goal is to embed virtual objects into the real-world, so they become indistinguishable from it. This ranges from correct occlusion handling to the application of virtual shadows on real objects and vice versa. This thesis can be primarily located in the field of Augmented Reality rendering, with a focus on real-world light estimation, also known as photometric registration and photorealistic rendering. Important prior art originates from research in inverse rendering and real-time global illumination. We furthermore believe that interesting Augmented Reality rendering problems lie in developing robust algorithms, which estimate parameters such as real-world light sources, and efficiently process these in a rendering pipeline, which can deal with imperfect input data. This is the main difference to problems in Virtual Reality, where the input data is precisely defined in most cases.

We further discuss different photometric registration and photorealistic rendering approaches for Augmented Reality, comparing our work to others, and conclude this chapter with a general summary.

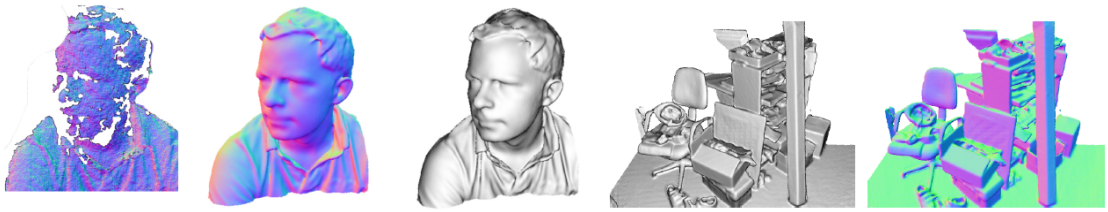


Figure 2.1: This figure illustrates intermediate steps of the Kinect Fusion algorithm [98]. The left most image shows the effects of the imperfect data from the depth sensor. Over time, more observations or depth maps are added to the reconstruction volume, and the quality of the surface reconstruction improves.

2.1 Real-time Geometry Reconstruction

Surface geometry reconstruction is a wide research field, dealing with the problem of digitizing the real-world geometry. Geometry reconstruction in general, can be distinguished by the type of sensor system, which has been employed to scan the environment. The most common reconstruction systems are based on vision camera [104], employing photogrammetric algorithms, laser range finders [47], which determine the distance to the geometry by measuring the return time of the reflection of laser beams, also known as time of flight (TOF) [25], or projector-camera systems, which project structured light patterns against the geometry, using computer vision for analysis. The common output of all geometry reconstruction algorithms is a point cloud in camera space, which has to be further processed to create a useful surface representation in object space. An extensive overview on such reconstruction algorithms is given by Berger et al. [8].

In our work, we are especially interested in interactive real-time reconstruction methods, which are capable of reconstructing dynamic geometry. This excludes a vast amount of photogrammetric algorithms [1, 43] based on structure-from-motion [112], which process a huge amount of images in batch mode to compute a consistent reconstruction of the environment. If the support of dynamically changing geometry is a requirement, as in our work, we cannot consider monocular simultaneous localization and mapping (SLAM) [24, 60, 81]. These methods provide good results for static geometry, but not for dynamically changing geometry.

For capturing fast dynamic changes, a sensor system which provides updates at a high frame rate is necessary. The MS Kinect V1 provides depth maps at 30Hz. It is based on structured light. Because of its high spatial resolution (640x480) of the depth map and its low costs, it has become very popular for research. The limitation of the MS Kinect lies in the high level of noise, increasing with the distance from the sensor. Additionally, highly reflective surfaces cannot be captured by the sensor and create holes in the depth map. Moreover, the disparity between the camera and the infrared projector creates a shadow in the depth map due to geometry occlusion. An example of the quality of the depth map is shown in the left most image in Figure 2.1. As discussed in Chapter 3, we chose the MS Kinect to capture the geometry.

To actually compute a consistent surface representation from such a depth map or

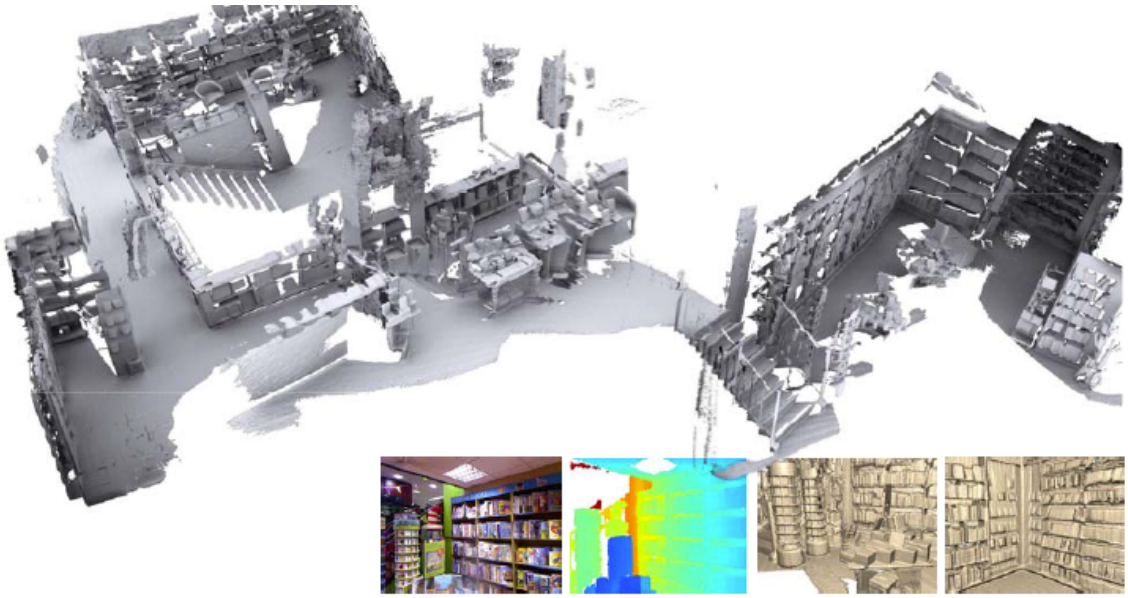


Figure 2.2: The enhanced KinectFusion algorithm by [15] is a volumetric reconstruction algorithm capable of processing large environments. The images shows the phong shaded rendering of the entire library. At the bottom, the input camera image and depth map are shown, plus two renderings of the current view.

dense point cloud, several algorithms have been presented in the past. One is object-space filtering, which continuously improves the reconstructed scene model by integrating depth images over time and space into a volumetric model [80](see Figure 2.1), a 3D point cloud [58] or a 3D map composed of depth keyframes [59]. The main advantage of object-space filtering is the global scene model, which can represent geometry outside the current camera FOV, also at relatively large scale [15] (see Figure 2.2). The main disadvantage is the inability to pick up fast scene changes, because robust integration requires time. Fast scene changes are classified as outliers and disregarded. Image-space filtering processes only the current depth image and uses the result directly as a scene model for light estimation [Lensing and Broll] and rendering [98]. The advantage of image-space filtering is that it instantaneously captures the current state of the scene. However, image-space filtering is more susceptible to sensor artifacts, cannot capture the scene outside the current FOV and requires additional effort for 3D camera tracking.

In this work, we combine object-space and image-space filtering to produce a scene model which is both smooth and dynamic and explain this more in detail in Section 4.2.2. This idea is related to scene change detection from depth sensors, which has, for example, been proposed for model-based object tracking [44] and telepresence [73]. However, these works aim at object-space results, while we use the object-space only as an intermediate step and compute global illumination in image-space.

Figure 2.3: Exemplary results for solving the intrinsic image decomposition problem (cf. Chang et al. [14]). The left image is the input image of a figure with line drawings as texture. The middle figure shows the result of the illumination component estimation, while the most right image shows the results of the reflectance or texture component estimation.

2.2 Photometric Registration in Augmented Reality

2.2.1 Inverse Rendering

Inverse rendering is the counterpart to the more commonly known forward rendering. Instead of creating an image from known geometry and lighting, inverse rendering deals mainly with the estimation of lighting from an image. Ramamoorthi and Hanrahan [95] presented a mathematically consistent framework for inverse rendering. They describe the reflected light field as a convolution of lighting and bidirectional reflection distribution function (BRDF). Moreover, they represent the light field as a product of spherical harmonics (SH) coefficients of the BRDF and the lighting. The same authors also presented an efficient way of rendering environment illumination using SH [94]. A main finding of this work is that a limited number (9 to 16) SH coefficients is sufficient for creating perceptually valid renderings. For a broader overview of inverse rendering problems, we refer to Patow et al. [87], Stephen Robert Marschner [75], and to the survey by Jacobs and Loscos [46] on illumination methods for mixed reality. In the following sections, we discuss several methods for photometric registration. We categorized these methods by the type of main input.

2.2.2 Photometric Registration From Static Images

In this section, we discuss several methods which take a static RGB image as input for photometric registration. Intrinsic image decomposition is one of them and deals with separating the lighting effects from texture or reflectance in single images. It has a long tradition in computer vision research, going back to the seminal work of Land et al. [66] from the 1970s, which introduces the Retinex algorithm. A great volume of algorithms has been published based on the principles of the Retinex algorithm [9, 29, 30, 106]. The Retinex algorithm is based on the assumption that light changes less frequently than texture. Considering, for example, an object with line drawings on it, we can observe that there are two sets of distinguishable image gradients. One set of image gradients belongs to the lines, and changes at high frequency. The other set is the effect of illumination reflected on the object, which changes at low frequency. An example is illustrated in Figure 2.3, which is taken from Chang et al. [14]. They propose a novel intrinsic image

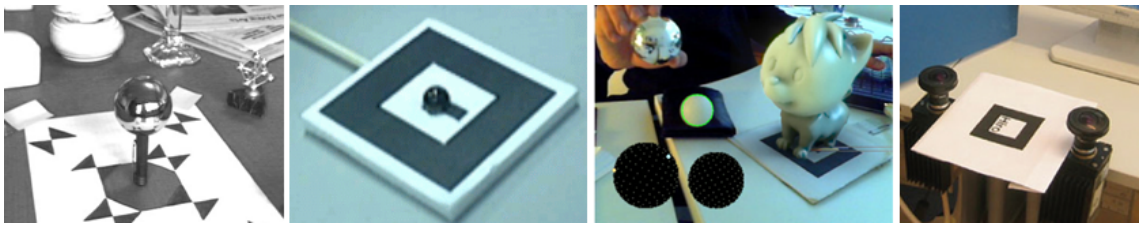


Figure 2.4: From left to right. Specular light probe [19] for capturing the environment lighting for image-based rendering. Small black sphere on a tracking target to filter ambient illumination for better specular highlight detection and light source estimation [52]. Ping-pong ball on black background for better detection [2]. Two HDR cameras with fish eye lenses employed in [33].

decomposition algorithm based on a generative, probabilistic model. A recent trend in intrinsic image decomposition is using the larger availability of depth images [6, 16, 67], which allows to derive assumptions on the image generation process incorporating surface properties. This approach is very close to the work discussed in Section 2.2.7 about photometric registration from arbitrary geometry. A further interesting work in the area of intrinsic image decomposition has been published by Bell et al. [7]. They created a dataset from real-world images, using croudsourcing to provide manually generated ground truth. Reflectance points have been classified manually by many different users. This is an important step for improving intrinsic image decomposition algorithms to work better on real-world data. A common objective for intrinsic image decomposition is to remove the lighting effects and to obtain the pure reflectance. More work on estimating the actual light source from single images has been published [70, 71, 83, 88]. The basic idea is to detect the contours of objects in the image. A contour provides a good cue to compute an estimate of the surface normal vector on the contour, which can then be used to trace back the position of the light source. Lopez et al. [70, 71] use this approach for relighting images, hence, removing the lighting effects from the image and applying an artificial one.

The solution presented by Karsch et al. [54] renders virtual objects into images. This approach is not real-time and demands manual user input to define proxy geometry for the real-world scene and annotating light sources in the image. They improve they approach [55] by automatic geometry estimation from images in combination with machine learning algorithms for reflectance and light estimation. They achieve compelling rendering results using a physically based rendering software (LuxRenderer *).

In the work of Van den Hengel et al. [113], the user has to specify the geometry through manual input. The appearance model of the geometry is estimated and transferred to other newly inserted virtual objects. However, the input of this work is an offline video sequence, which can be seen as a series of static images, with the advantage of using the knowledge of preceding and succeeding frames.

*http://www.luxrender.net

2.2.3 Light Probes

In this section, we discuss photometric registration based on active and passive light probes. Most common passive light probes are spherical objects. A sphere has the important characteristic of a surface which uniformly reflects the environment light from all directions. The reflection is captured with a video camera and further processed for light estimation. If the sphere has a highly reflective surface material (mirror balls), an environment map can be created directly from it. In general, we cathegorize passive light probes as objects with known geometry and surface material properties, which are inserted into the scene and not part of the actual application scenario. Geometric and surface material properties may vary and are not restricted to spheres only. Active light probes are secondary camera systems with a wide field of view (fish eye lenses), which capture the incident lighting directly, providing a panoramic image from the environment. In Figure 2.4 we show examples for passive and active light probes.

Passive light probes: The principles of using passive light probes for photometric registration can be traced back to Paul Debevec [22]. This early work shows how to recover high dynamic range irradiance maps from multiple photographs capturing a mirror ball. Subsequently work Debevec et al. [19] and Yu et al. [119] demonstrated how to correctly light virtual objects inserted into the real scene with measured scene radiance and global illumination. However, this work was not been implemented for real-time applications. Powell et al. [93] used three mirror balls to compute the direction of point lights by triangulation. In the following, we discuss only real-time methods for photometric registration, organized by how the environment light is observed. Kanbara and Yokoya [52] presented a method how to estimate the light environment in real-time for Augmented Reality using a fiducial marker for geometric registration and a black mirror ball for filtering low frequency lighting to detect specular highlights better. They recover dominant light source directions, assuming that all viewing vectors are parallel to the optical axis of the camera. Yusaku et al. [84] recover HDR images from spherical reflective light probes. In contrast, recent work by Aittala [2] captures diffuse lighting using a diffuse light probe, in this case a ping-pong ball or a rotated planar marker. The interesting aspects of the ping-pong ball light probe is the diffuse reflectance characteristics. Aittala solves the relationship between a general light source model and the light intensity observations from the camera images using L1-regularized least squares minimization. This allows to robustly estimate the dominant light sources from the diffuse environment map.

Active light probes: Active light probes are preferable if high quality light estimation is desired at little computational costs. Grosch et al. [33], Knecht et al. [61] and Kan et al. [51, Kan and Kaufmann] use special camera sensors for capturing the environment lighting. Their approach is based on a high dynamic range camera with a fish-eye lens. Compared to mirror balls, which can cover a field of view (FOV) of 300 degrees, the fish-eye lens provides only a FOV of about 60 degrees. Moreover, fish-eye HDR cameras can be expensive and are not always available in casual Augmented Reality scenarios.

2.2.4 Light-Field Reconstruction

Passive and active light probes provide the light-field of the environment directly. In the following, we discuss methods, which acquire the light-field from the environment by reconstruction, using a monocular camera system. Pilet et al. [91] propose a fully automated method for geometric and photometric calibration from an arbitrary textured planar pattern, by reconstructing a diffuse lightfield. In this work, the observed geometry is constrained to a planar surface, and could be seen as a passive light probe. Their photometric registration process is offline and involves waving the planar surface under different angles in front of a camera. This simulates the characteristics of a sphere, and reflections from different light directions are captured over a sequence of frames.

Jachnik et al. [45] also assume a planar surface, which has half-diffuse, half-specular material properties, as it is very common for books with glossy covers. This could also be seen as another interesting special case of a passive light probe. Since the light probe provides only one surface direction, the user has to reconstruct the incident light-field by sampling the light probe with a camera around a hemisphere. This has the same effect as waving the book in front of a static camera under different angles [91]. Using visual tracking, the position of the camera is estimated, and the camera images are projected and merged into a common light-field map. Using visual feature based tracking requires salient feature points in the camera image. Therefore, this approach does not work with fully-specular material properties, such as a mirror.

The creation of panoramic images of the scene falls in the category of actively sensing the light-field, a panorama, instead of reconstructing it from reflections, as discussed in the two previous examples. Compared to active light probes with a dedicated second panoramic camera, these approaches use a single camera system and reconstruct the panorama beforehand. For example, DiVerdi et al. [23] show a system for online creation of panoramic images using vision base tracking. The panoramas can be used for image based lighting of virtual content rendered into the camera image. This method projects the panorama to a cylindrical model, which is a very simplified assumption of the scene geometry.

Meilland et al. [78] take this approach a step further, by online computing a reconstruction of the scene geometry using a RGBD based tracking system. Moreover, they estimate high dynamic range texture maps for the surface geometry. This improves discriminating light sources in the acquired textures from scene geometry, because light sources also appear in the higher spectrum of the light range. However, because of their single camera approach, both methods are not able to react on dynamically changing geometry or lighting. For example, if the light situation would change from one frame to the other, the change would only be registered if the camera points to the light source directly.

2.2.5 Photometric Registration Using Shadow Cues

Another method for estimating light sources is to observe the shadows in an image. The principal method is based on the partial knowledge of the geometry of the shadow caster and the correct classification and measurement of the shadow appearance in the image. In practice, this means detecting the shadow and its contour in the image. Surface points on the contour are traced back to the object geometry boundary of the shadow caster,

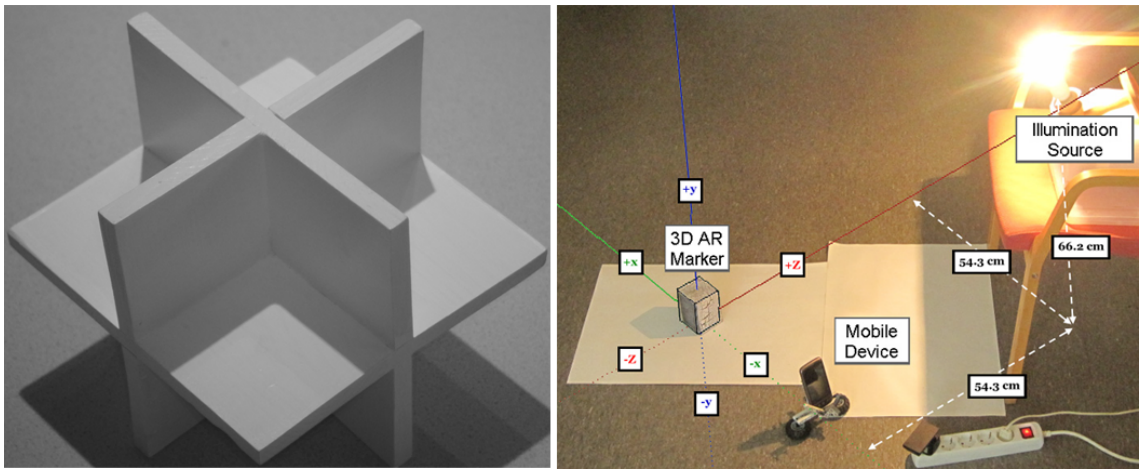


Figure 2.5: The left image shows the shadow catcher form Haller et al. [38], which is designed to create visible shadows from any camera view point. On the right image is the setup for shadow based light source estimation on mobile phones illustrated [3].

which also has to be in the FOV. From there, the light source direction can be directly estimated. For example, Haller et al. [38] utilize a light probe with special geometry characteristics - "shadow catcher" - which reliably captures shadows from various directions (see Figure 2.5, left). Wang et al. [116] presented a method for estimating multiple directional light sources from known geometry. Arief et al. [3] demonstrate the same concept of light source estimation from cast shadows on a mobile phone. Their approach has only been applied to static images and combines a shadow based method and a shading based method (see Figure 2.5, right). A major challenge in photometric registration from shadows is actually the problem of detecting the shadows. Depending on the quality of the image and the textureness of the scene, this can become a hard problem. In this section, we covered methods which solely compute the lighting from shadows and not from the reflectance of the geometry. In the next section, we cover photometric registration methods from arbitrary geometry, using the reflectance or the reflectance and shadowing in combination as input.

2.2.6 Photometric Registration From Arbitrary Geometry

In this section, we particularly focus on photometric registration algorithms, which take the camera image and real-world scenes with arbitrary geometry, different from spheres or planes, as input. A further distinction can be made if shadows are incorporated into the estimation.

In 1999 Jürgen Stauder [110], presented one of the first systems capable of autonomously estimating a single distant point light and ambient light from estimated scene geometry. The application scenario was video conferencing with virtual augmentations in the video stream. The light estimation was constrained to scenes with non-occluding, only rigidly moving geometry and static uniform backgrounds. The geometry reconstruction and tracking itself was based on the segmentation of moving

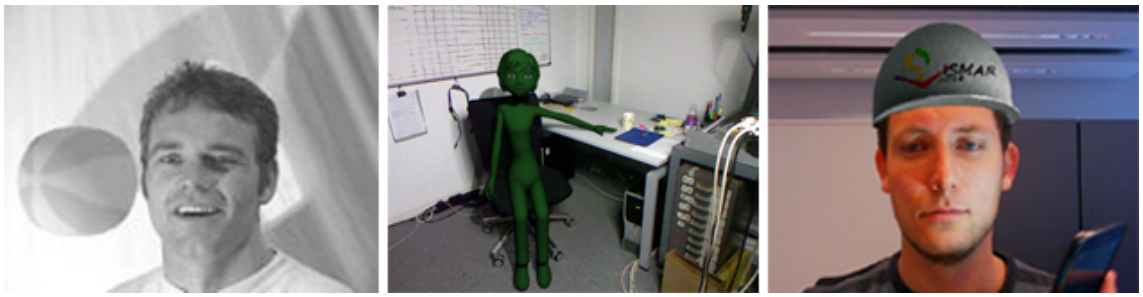


Figure 2.6: Left: Early work of Stauder [110] showing autonomous light estimation for augmented video conferencing. Middle: Augmented Reality rendering of Boom et al. [10]. Right: Light estimation from human faces by Knorr et al. [63].

geometry from static background and approximating the geometry with ellipsoid like 3D models. At the other side, the light estimation was solved by observing the intensity values on the reconstructed geometry of two consecutive images. This allows to separate the ambient lighting from directional diffuse lighting. The actual light estimation was computed by minimizing a cost function for the unknown lighting parameters, which are assumed to model the observed intensity values. In the left most part of Figure 2.6, the result of a rendering is shown, including a cast shadow.

A method for estimating multiple directional light sources using a single image with known geometry has been presented by Wang et al. [116]. They use the brightness variation of the shading on the object in combination with the brightness variation of the shadow generated by the object. Using shadow information as an additional cue improves the robustness of the estimate from shading, if the geometry does not provide enough different surface normal vectors. This can be the case, if the geometry of the object does not have to have a shape similar to a sphere. The geometry itself has been reconstructed in a preparation step using a laser scanner. Similar work has been published by Ikeda et al. [42], computing the illumination from object shadows and incomplete object shape information, using a RGBD camera for reconstructing the geometry.

Weber et al. [117] proposed a framework to reconstruct a geometric model, the real-world lighting and the Lambertian surface reflection parameters from a collection of images. The objective of their work is to create a digital model of the object with known material properties, which can be used for relighting from different views and with different light sources. In their work, they place objects on a turn-table and compute a reconstruction of the geometry using space carving methods. The geometry is, represented as a voxel model.

Boom et al. [10] presented a system which estimates a single point light source from an arbitrary scene geometry using an RGBD sensor (Kinect). They assume a Lambertian reflectance model for the entire scene. The approach is based on segmenting the RGB image by color into different sets of pixels, assuming that each set has the same albedo. The albedo is estimated by taking an arbitrary light source as input. The result is a uniform albedo for each pixel set. In the following, they reconstruct the original image using the estimated uniform albedo of all pixel sets. Computing the difference between

the reconstructed image and the original input image results in an estimation of the light reflection, which is then taken as input for estimating the real point light source. They evaluate their work on synthetic and real-world data. The result of a rendering can be seen in the middle part of Figure 2.6.

In 2014 Knorr, et al. [63] published their work on estimating the real-world light from human faces. Their method uses offline machine learning of various face appearances, which are then matched to distinctive observations points in the actual face in the camera image to estimate the real-world light. The database used in the learning step consists of different faces captured under varying illumination. The pose of the face is detected using vision-based tracking. The estimated light, is applied to the virtual objects, positioned relative to the pose of the face, using a spherical harmonics based rendering pipeline (Figure 2.6, right). Moreover, they claim that their approach is fast enough to run on mobile phones.

2.2.7 Photometric Registration From Specular Highlights

Specular highlights can give a strong cue about light sources. If a specular highlight is detected correctly in the image, and the surface normal vector from the position is known, the direction of the incident light ray can be estimated, by inverting the viewing direction from the camera with respect to the surface normal vector. However, reliably detecting specular hightlights is a challenging task, especially in real-world scenarios, where different surface material properties are present, and lighting and geometry can change dynamically. Therefore, most previous work in this area operates on rather static and artificial settings.

In 2006 Lagger, et al. [65] presented solution using specular highlights to estimated multiple light sources. Their work is mainly focused on a robust detection of specular highlights on small moving objects. Hara et al. [39, 40] estimate the light source position and the reflectance parameters from single images in indoor environments without a distant light source assumption. Similar work has been presented by Mashita et al. [77]. Their approach tries to infer the real-world lighting by detecting specular highlights from planar objects. Although their work does not require special lightprobes, it is restricted to the constraint of a known planar surface, which makes it rather impractical for real-world scenarios in unknown environments. This work has been extended by Plopski et al. [92] using real-time geometry reconstruction to leverage the constraint of registered planar surfaces. They presented a system capable of estimating the real-world environment lighting and reflectance parameters of the surface materials in the scene. Similar to Mashita et al. [77], they rely on detecting specular highlights. They demonstrated their algorithms on synthetic test scenes and also real-world scenes. However, reliable detecting view-dependent lighting effects, such as specular highlights, on real-world surfaces is a limiting factor regarding the robustness of their approach. Moreover, in contrast to the work in this thesis, they do not cover fast dynamically changing geometry of larger workspaces including global illumination effects such as shadows and indirect illumination.

Hyunjung Shim [107] estimates all-frequency lighting using pairs of color images and a depth image. The approach is based on separating diffuse and non-diffuse (specular highlights) illumination effects in the color images . Diffuse lighting is modeled with spherical harmonics, and point lights are estimated from the specular highlights independently.

Hence high frequency lighting is not derived from low-frequency lighting. This improves the overall quality of the light estimation. They achieve high quality results, but since the evaluation of the algorithm has been done on modeled synthetic data only, it does not reflect real-world scenarios with moving cameras and noisy signals.

2.2.8 Photometric Registration in Outdoor Augmented Reality

Prior work in outdoor Augmented Reality often combines several different approaches for photometric registration. Shadow cues are also used for estimating the light situation outside [11, 12]. In outdoor Augmented Reality, the main light source during the day is the sun. Therefore photometric registration can be solved by using the geospatial position and the current date and time. These parameters give a first estimate of the position of the sun. In conjunction with depth information from a stereo camera system, Madsen et al. [72] present work on the refinement of the sun position, enabling Augmented Reality rendering with cast shadows. Yanli et al. [69] model the illumination based on the sun as a time-varying directional light, but also track local lighting variations from a sparse set of feature points in moving camera images.

2.3 Global Illumination in Augmented Reality

A large body of work has been published about extending basic local lighting for Augmented Reality, for example, by supporting discrete directional light sources and shadows from real onto virtual objects and vice versa. These approaches are also categorized as common illumination [21, 46, 64] and can be distinguished from global illumination approaches by the fact that no indirect light interaction between objects is computed. A broad over view about real-time global illumination techniques in Virtual Reality can be found in the survey by Ritschel et al. [99], which covers most algorithms used for global illumination in Augmented Reality. In this section, we rather focus on work in the context of global illumination in Augmented Reality with an emphasis on the specific rendering algorithm. In the following, we distinguish prior work by the type of global illumination effects they support.

2.3.1 Indirect Illumination

Grosch at al. [32] presented a solution in 2007, which computes the near-field and far-field illumination for Augmented Reality scenarios, where the daylight created by the sun represents the distant far-field illumination, and the near-field illumination is the indirect lighting present in a room coming through a window. The real-world lighting is captured by an active light probe with a fish-eye lens. Real-time performance for indirect lighting is achieved through precomputing an irradiance cache, based on spherical harmonics. The idea of the irradiance cache is based on subdividing the rendering space into volumetric regions and computing and storing the irradiance or radiance transfer for each region. The final irradiance cache is a linear combination of the real-world lighting represented in spherical harmonics and the precomputed irradiance in spherical harmonics. Far-field illumination relies on image based lighting, where the image from the light probe is sampled

Figure 2.7: Left: Irradiance caching based indirect illumination by Grosch at al. [32]. Middle: Light factorization in combination with direct shadows and near-field shadows by Nowrouzezahrai et al. [85]. Right: Real-time differential rendering by Knecht et al. [61].

using importance sampling. In the left figure of Figure 2.7, a test setup is shown, with the active light probe on the top of the Cornell box. The irradiance cache is computed for the entire volume of the box. Indirect illumination is visible by the colored shadows of the bunny.

Nowrouzezahrai et al. [85] presented work on a light factorization algorithm, which separates the real-world ambient light from real-world directional light sources, which have been measured using a mirror light probe. In the rendering pipeline, they support illumination from real-world lighting in combination with direct shadows, cast shadows and self- shadowing on rigidly animated virtual characters, by employing a mixture of shadow mapping and approximated precomputed radiance transfer for dynamic geometry [97]. They use spherical harmonics to store the radiance transfer and to compute the shading. Their system assumes simple static plane as real-world geometry (see Figure 2.7, middle).

Knecht et al. [61] present an algorithm computing instant radiosity [57] for real-time global illumination, extended by virtual point lights and improved imperfect shadow maps [100]. Their proposed solution measures the real-world lighting using an extra camera with a fish-eye lens. Furthermore, they present a new solution, which computes differential rendering in one shading pass, and reduce temporal flickering exploiting frame coherence. In the right figure of Figure 2.7, an example is shown. Indirect lighting from the virtual dragon is visible as green reflection effects on the real-world tea pot. Compared to the work of Grosch at al. [32], this work does not need expensive pre-computation of an irradiance cache, which enables rigidly moving objects.

A different approach, has been published by Franke [27]. His approach is based on radiance transfer fields [86] and light propagation volumes [53]. To enable differential rendering, the difference between the light propagation of the real-world and the light propagation of the real-world and the virtual world together is computed. The direction of the real-world light sources are determined using marker based tracking or by extracting the light source from the image of an active light probe. The proposed solution is capable of computing dynamic indirect illumination effects from virtual to real. However, they do not incorporate a real-world reconstruction, and no effects from real to virtual geometry

Figure 2.8: The left image shows the results and visualization of delta propagation volumes by Franke [28]. The right images shows indirect illumination effects on the dragon by Lensing et al. [Lensing and Broll].

have been shown. In the left part of Figure 2.8, the light propagation volume of an example is shown.

Lensing et al. [Lensing and Broll] use the depth image from a RGBD camera to compute indirect illumination effects between virtual objects and the real-world scene. They apply a guided filter to the noisy raw depth image. The result is a smooth depth map, which enables a better surface normal estimation. Compared to the work in this thesis, Lensing et al. do not compute a consistent volumetric reconstruction of the real-world geometry, and can therefore only process information which is in the current FOV. Similar to Knecht et al. [61], they employ virtual point lights in their rendering framework. It is important to mention that the light sources are virtual, and they do not incorporate real-world light estimation. In the right part of Figure 2.8, a scenario with a virtual dragon is shown. The torch light illuminates the red planar surface, from which light bouncesf of and becomes visible as indirect illumination on the virtual figure.

2.3.2 Reflections, Refractions and Caustics

The work of Pessoa et al. [90] relies on image-based lighting. They observe a passive light probe (mirror ball) to obtain an environment map and model anisotropic reflection, and refraction effects. In 2012, Pessoa et al. [89] present an extend modular framework of their previous work [90].

Grosch [31] presented a seminal work based on photon mapping, rendering reflections, refractions and caustics from virtual objects into real images using the concept of differential rendering. Kan et al. [Kan and Kaufmann] increase coherence in Augmented Reality rendering by computing arbitrary specular and refractive effects with a real-time raytracer and photon mapper, based on NVIDIA Optix. The solution computes differential rendering in one pass and supports global light transport between real and virtual geometry. Moreover, a user study has been conducted, proofing the positive impact of high quality visual coherence rendering on the Augmented Reality application. In the right figure of Figure 2.9, an example of this work is shown, where a virtual transparent object is placed

in front of a real-world object. Their system is also capable of rendering caustics from virtual objects onto real-world objects in real time. In 2013, Kan et al. [51] developed a differential irradiance cache to improve the performance for their Augmented Reality rendering system. The system provides indirect illumination, reflection and refraction. However, fully recursive raytracing is generally computationally expensive and was only shown for small scenes. Both works of Kan et al. use an active light probe to estimate the real-world lighting.

In 2014, Franke [28] published an algorithm based on delta voxel cone tracing. This work supports soft shadows, diffuse indirect lighting and glossy reflections of different qualities from mirror like to diffuse. The employed technique is based on precomputation and voxel cone tracing [17] and improves previous work on delta voxel tracing [27] by quality and scalability. An example of the results is shown in the middle figure of Figure 2.9.

Knecht et al. [62] demonstrate a system for Augmented Reality, which is capable of rendering virtual objects into a real-world scene, incorporating advanced global illumination techniques with physically plausible reflection and refraction effects. They extended and altered their rendering pipeline based on differential instant radiosity [61], which only supports diffuse and glossy light bounces, by applying several techniques other than expensive virtual point lights, to achieve real-time. For example, to improve the quality of reflection rendering, they use imposters in stead of simple cube maps. Their algorithm for computing caustics is based on Wyman et al. [118], where small quads and photon geometry buffers [105] are used. In the right figure of Figure 2.9, an example of their approach is shown.

Csongei et al. [18] presented an early prototype of a mobile system, which investigates the impact of displaying global illumination based Augmented Reality rendering on a mobile phone. Their approach is based on a server-client model supporting distributed rendering, where expensive global illumination is computed using a ray-tracer on a PC (server). The mobile phone (client) sends the actual camera position and camera image of the phone to the PC and receives back the final rendering. Furthermore, they rely on an image-based lighting model using a pre-captured panorama of the real-world environment. Hence, their photometric registration is static and does not support dynamic light changes.

2.4 Summary

In the following, we summarize the related work in respect to the work presented in this thesis. In Chapter 4, we discuss our contribution to geometry reconstruction. Compared to the work of Newcombe et al. [80], we propose a lightweight solution, which provides the reconstruction of fast geometric changes of dynamic geometry and high quality reconstruction at the same time. Comparing the work of this thesis to other light reconstruction algorithms based on arbitrary geometry, discussed in Section 2.2.7, we support real-time geometry reconstruction of deformable and moving objects. Moreover, we incorporate near-field shadows into our illumination model for estimating the environment lighting to achieve robustness. Compared to the work of Boom et al. [10], we estimate the entire environment lighting, instead of only one point light source, as discussed in Chapter 5. Additionally we incorporate global illumination effects in the Augmented Reality rendering

Figure 2.9: Left: Real-time raytracing and photon mapping based differential rendering by Kan et al. [Kan and Kaufmann]. Middle: Delta voxel cone tracing results by Tobias A. Franke [28]. Right: Reflection and refraction effects by Knecht et al. [62] based on their enhanced instant radiosity framework.

pipeline, such as consistent shadowing and indirect illumination (see Chapter 7). Compared to rather computational expensive global illumination algorithms for Augmented Reality, such as Kan et al. [51], discussed in Section 2.3, our rendering approach is fast enough to process mid-sized workspaces supporting shadow detection from inside and outside the FOV and indirect illumination. Our algorithms are based on accelerated volume ray-tracing and radiance transfer approximation in image space (see Chapter 6) and do not need any kind of precomputation, enabling interactive lighting interaction between deformable real-world geometry and virtual geometry.

3 System Overview

Contents

In this chapter, we discuss the layers of a system we build for photometric registration and Augmented Reality rendering. Since a major goal of this thesis is building a practical Augmented Reality solution using commodity hardware, the combination and interplay of each layer is crucial. Therefore, building such a system requires several design decisions, ranging from the sensor hardware to the software platform. We will report these decision combined with an overview of the layers and the general pipeline in Section 3.1. In Section 5.3 we report on which hardware setups we used for evaluation.

3.1 Overview

Our system can be subdivided into several different layers. In Figure 3.1 we visualize these layers and their relationship to the photometric registration pipeline. In the following, we discuss each layer in turn.

Computer aided design For creating the virtual content, we used computer aided design (CAD) software such as AutoDesk 3D Studio. Our software system allows loading virtual objects in the OBJ format*. Special care has been taken on the complexity of the geometry of the virtual models to meet real-time requirements. We found a maximum of 20000 polygons for the main virtual asset a good upper boundary. This naturally rules out virtual models with high geometric details and therefore high polygon counts, designed for offline rendering. Our system implements different approaches for Augmented Reality

*http://en.wikipedia.org/wiki/Wavefront_.obj_file

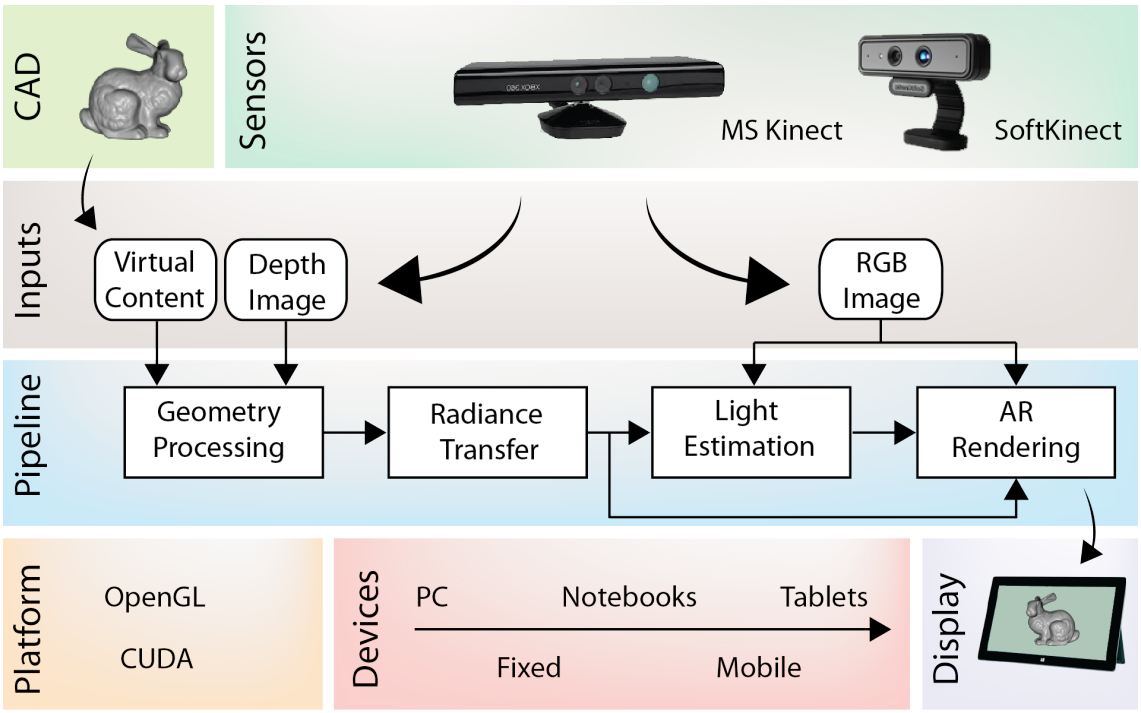


Figure 3.1: This figure visualizes the main layers of our photometric registration and rendering system.

rendering. One approach based on volume ray-tracing requires the representation of the virtual content in both a voxel-based data structure and a mesh-based data structure. We used "binvox"[†], which is a freely available voxelization tool for voxelization of the triangular mesh in a pre-processing step.

Sensors The requirements for the sensors from our system are twofold. Since we build a video see through Augmented Reality system, we need to have a RGB sensor for representing the real-world on a display. Reconstruction of dynamically changing geometry requires a sensor setup which provides the necessary information to create a depth map from the current view as fast as possible. One choice is a stereo setup with two RGB cameras. This solution comes with the expense of computational effort for creating the depth map itself. Monocular camera systems are also capable of reconstructing the geometry of the environment, but have more difficulties capturing dynamically changing geometry [24, 81]. With the Microsoft Kinect, which provides a depth map directly, depth sensors became affordable. Therefore, we designed our software pipeline arround RGBD sensors like the Microsoft Kinect or the SoftKinect from PrimeSense which has a smaller form factor and can be attached to mobile devices. Both devices are low cost consumer hardware. Our primary device is the Microsoft Kinect, which became very popular in the last years. Compared to costly industrial devices the quality of the sensors is limited and provides relatively noisy input data but delivers high frame rates. An important limitation

[†]http://www.cs.princeton.edu/ min/binvox/

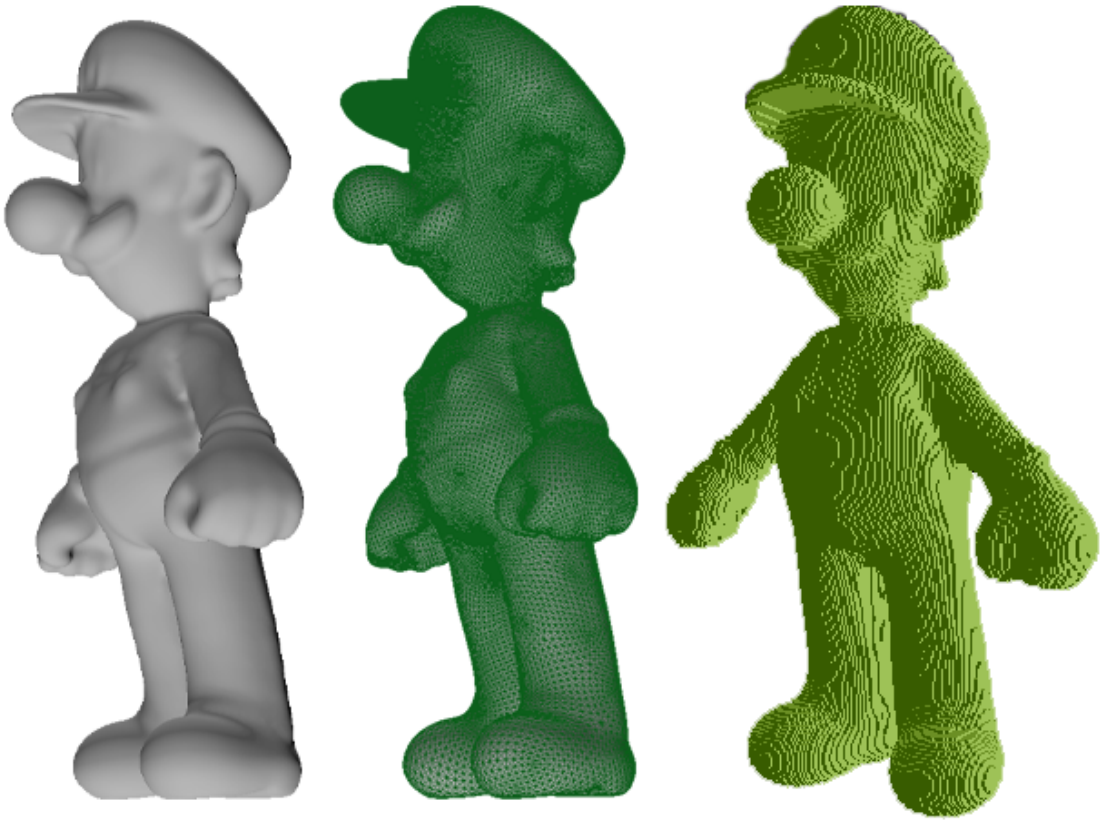


Figure 3.2: Left: shaded 3D model Middle: visualized triangle mesh Right: Voxelized 3D model with a voxel resolution of 256 cubed.

of the Kinect is that it works only indoors. The underlying depth estimation algorithm is based on structured light [121], emitted in the infrared range. Due to the high amount of infrared light of the sun, the Kinect cannot operate outdoors. However, this deficiency is acceptable for indoor scenarios. Other artifacts in the depth map are created from highly reflective surfaces such as mirrors or monitors. An advantage of a depth sensor in general is that it is not dependent on the textureness of a scene. Since the pose estimation algorithm we employ [80] is based on depth data, it can operate in environments with few to no textured areas, a condition where feature based tracking and reconstruction algorithms easily struggle.

Inputs Our system takes inputs from three primary sources, the geometry and texture data for the virtual content, the depth data from the depth sensors and the RGB data from the camera. It is important to note that no additional manual input from the user is required. There are applications such as [54], which require information from the user first, such as the manual approximation of real-world geometry.

Pipeline The photometric registration and Augmented Reality rendering pipeline is the core part of our system. We introduce this pipeline more in detail in Section 3.2.

Platform We implemented our software solution in C++. For graphics processing we used OpenGL 4.0 in combination with CUDA 5.5. All time critical parts of the system were implemented on the GPU exploiting the power of parallel execution.

Devices We tested our system on several different devices. Our primary development and testing device was a personal computer (PC) with an NVIDIA GeForce 780. In combination with a Kinect, we used a handheld depth and RGB sensor and a static display. We also proved that our system is capable to run on a Notebook with a mobile GPU (NVIDIA Quadro K2100M) and a tablet with an NVIDIA GT640M. Especially the latter device represents a fully mobile system, which has no cables attached and therefore does not restrict the user's experience in moving around freely.

Display Our system is based on the concept of video see-through Augmented reality, where the real-world is represented through a constant video stream and the augmentations are rendered on top. Naturally the best user experience is achieved when the RGB camera sensor device is rigidly and attached directly to the display, facing away from the display plane. This is the case for the mobile tablet system explained in the previous paragraph. The user can naturally look through the display and explore the augmented real-world. However, there are Augmented Reality applications which do not require the RGB sensor direction to be aligned with the display plane. Such systems are for example magic mirrors [41]. This configuration is easily build with a stationary PC system.

3.2 Photometric Registration and Augmented Reality Rendering Pipeline

In order to deliver visually coherent rendering in Augmented Reality based on real-world photometric registration, the following four main steps are processed by the pipeline.

Geometry Processing: The first step is concerned with computing the surface geometry of the environment using the depth stream from the depth sensor. Additionally, the 6DOF camera pose is estimated for tracking. The result of the Kinect Fusion [80] reconstruction and tracking algorithm is a volumetric representation of the real-world geometry stored in a 3D voxel volume, where the actual surface of the geometry is stored in an implicit way. To compute the surface, rays are cast from the camera into the volume to determine the surface boundaries with a truncated signed distance function (TSDF). This delivers a more smooth and consistent depth map compared to the raw depth map from the camera. Moreover the volume contains the geometry in 3D, which allows more accurate testing for visibility to compute shadows. In this thesis, we investigated and developed geometry processing algorithms targeting dynamically changing geometry. We describe the evolution of these algorithms more in detail in Chapter 4.

Radiance Transfer: The second step in the pipeline is computing the radiance transfer. Radiance transfer describes how the light travels through the reconstructed geometry and is necessary for solving the photometric registration and for Augmented Reality rendering. This makes the radiance transfer computation a key element in the entire system. The radiance transfer in general is computed by solving the rendering equation for every pixel. In this thesis, the rendering equation solves for the Lambertian illumination model, including first bounce shadows. The representation of the radiance transfer is crucial. For photometric registration, the radiance transfer has to be stored as intermediate value similar to deferred shading approaches based on precomputation [108], but different to traditional rendering, where the result of the radiance transfer can be immediately used to compute the shading of a pixel and can be discarded afterwards. To support dynamic environments including changing geometry and lighting, the radiance transfer computation in this pipeline is computed instantly on a per frame basis. For efficiency, we compress the radiance transfer with spherical harmonics (SH) and store the coefficients per pixel in a camera image aligned buffer. In Chapter 5, we discuss the radiance transfer computation and processing more accurately. Since radiance transfer can become very costly in terms of processing, we investigated several acceleration techniques presented in Chapter sectionsec:acceleration to achieve real time performance.

Light Estimation: The third major step in our pipeline is the light estimation, which estimates the distant real-world environment lighting. The light estimation takes the intensity image of the RGB camera and the pixel aligned radiance transfer in SH as input. Then a linear equation system is solved to estimate the environment lighting. The actual result of the light estimation step is also represented in SH form. We discuss the light estimation more in detail in Chapter 5 and also introduce techniques for more robust light estimation for noisy input data.

Augmented Reality Rendering: The fourth and last step in the pipeline computes the final Augmented Reality rendering. This is the step where virtual content is added. To achieve a visually coherent embedding of the virtual geometry, into the real geometry we employ differential rendering [19]. The concept of differential rendering is that two different shadings are computed. One shading is the shading of the real-world geometry. The other shading is the shading of the real geometry and the virtual geometry together. Shading in our pipeline is the geometry lit by the real-world lighting estimation. The difference of both shadings represents the lighting effects between virtual geometry and real-world geometry and is applied to the real-world camera image in a final composting step. Lighting the geometry in our pipeline is based on deferred SH lighting and requires the radiance transfer buffers in SH and the real-world light in SH. However, in order to compute the shading for the virtual geometry and real-world geometry together, we have to compute another camera image aligned radiance transfer buffer. In total, we compute the radiance transfer twice. We describe our differential rendering pipeline in Chapter 7. Furthermore, we report on the integration of global illumination effects in our Augmented Reality pipeline, supporting consistent shadowing in dynamic environments for the entire workspace and indirect illumination based on the real-world light estimation.

3.3 Measurement Tools and Evaluation Environments

Measuring the quality of the light estimation is an important, but also complex task, where ground truth or reference data is hard to obtain. We solve this problem in different ways and discuss in the following the measurement tools which we employed to obtain reference data (see Section 3.3.1). Moreover, we discuss the creation of special assets to assist controlled experiments in specially designed evaluation environments (see Section 3.3.2).

3.3.1 Light Probes

Light probes for estimating the environment light for AR [19] have been considered as state of the art for many years. For generating reference data, we used two kinds of light probes: passive and active. The passive light probe can be a diffuse sphere or reflective mirror sphere with known reflection properties. The reflected light on the sphere is captured by the camera. For constantly updating photometric registration, the sphere in the camera image has to be detected and tracked over time. This can turn into a complicated computer vision task, especially if the quality of the camera image is low (e.g., camera blur or noise). Moreover it becomes more difficult to track the sphere if it has a purely reflective surface. For example, in [2] a ping pong ball (the light probe) is tracked against a constant black background to facilitate tracking the boundary of the sphere. Therefor most work in Augmented Reality registers light probes to a tracking target (e.g., ARToolKit marker [56]). The advantage is a relatively easy and cost friendly setup. A disadvantage is that the light probe always has to be in the field of view of the camera. This limits the user's radius of action. Second, the registration of the light probe with the camera image can suffer from tracking errors. For acquiring reference data, we also register a spherical light probe to the world coordinate system with an ARtoolKit marker. The sphere has been painted gray using wall paint, which is known for its diffuse reflection characteristic. The reason for choosing a diffuse gray painted sphere over a mirror ball was because it represents a diffuse gray world assumption for the environment. In Figure 3.3 on the left, we show the passive light probe we build.

To acquire more accurate measurements, we also used an active light probe. Active light probes consist of a second camera system, which is inserted into the scene or close to the working space and directly captures the environment lighting. The advantage is that active light probes do not have to be necessarily in the FOV of the camera, which makes them slightly more practical. Additionally light sources are measured directly, and the result of an active light probe can be used as an environment map for image based lighting [19]. An example can be seen in 3.4. For our evaluations we employed a Ladybug3 camera from Point Grey[‡] which consists of a rigid assemply of five cameras with wide angle lenses.

3.3.2 Controlled Environments

Besides state of the art light measuring systems, we also designed and created controllable environments for repeated testing. This includes real-world scenarios where we can control

[‡]http://www.ptgrey.com/ladybug3-360-degree-firewire-spherical-camera-systems

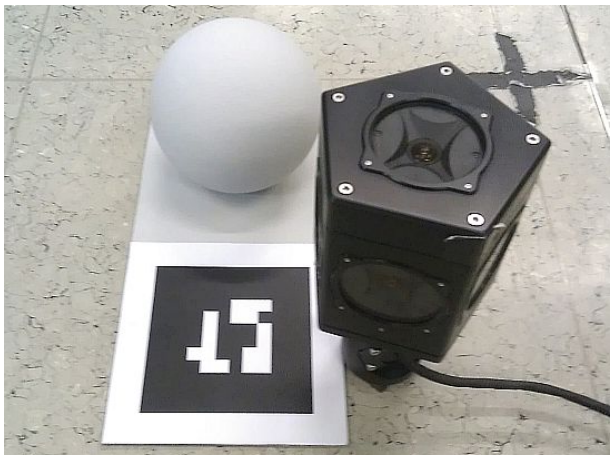


Figure 3.3: The passive light probe, which is a gray sphere, is shown on the left. The camera to light probe registration was solved by using an ARToolkit marker and tracker. On the right side the omnidirectional Ladybug3 camera which was employed as active light probe is shown.

the light situation, the geometry in the scene or even the surface properties of the geometry. We designed a physical AR stage set, which can be easily replicated and has a complete virtual representation for virtual simulation. We also build a life-size kid's room, which provides enough geometric details to demonstrate Augmented Reality scenarios.

Light Setup: In Figure 3.4, we show an example for a controllable light scenario. It consist of four different area lights, which can be switched on and off individually. The purpose of this scenario is that we can control and repeat the exact light situation, while changing, for example, the geometry. To simulate indirect light, we installed light sources (in Figure 3.4 the left most and the right most light source) towards a white reflective surface. Since we were mostly interested in estimating the dominant direction of the light, which we assumed to be white, equal light temperature of the light sources was not a requirement.

City of Sights: The digital and physically available City of Sights [34] dataset has been created to provide better control over environment parameters and investigate effects of immersion factors that are not yet available through current AR hardware. With exactly comparable physical and virtual models, validation experiments can be run, which test some real-world scenarios using a real Augmented Reality system, and some with a simulated Augmented Reality system. Targeting a broader range of Augmented Reality applications, we identified the following set of desirable properties for the model to be useful for a wide range of Augmented Reality research.

1. it should exist physically and virtually,
2. it should exhibit a range of properties complexities in terms of texture and geometry,
3. it should be customizable and extensible,

Figure 3.4: This figure shows a panorama image of the controlled lighting environment. There are two directed light sources in the middle and two indirect light sources on the left and on the right of the panorama. This image has been created by an omnidirectional camera (Ladybug3).

4. it should be physically replicable by anyone,

5. it should be accompanied by ground-truth observations and meta-data.

We used this dataset for our light estimation experiments, exploring the possibility to change the surface texture while keeping the same geometry (see Figure 3.5 bottom row for examples) conducting systematic evaluations in Section 5.3. Moreover, as further described in [34] the dataset can also be used for evaluation of tracking systems, modeling, spatial AR, rendering tests, collaborative AR and user interface design.

Realistic test room: The previously discussed environments and datasets were focused on rather technical evaluations, but do not provide a realistic environment. Therefore we designed and built a room to test and experience our photometric registration and rendering system in real life scenarios. In building this room, we had the following design decisions in mind. The room should

- exist physically,
- exhibit a range of properties/complexities in terms of texture and geometry, and
- should be customizable and extensible.

In Figure 3.6, we show two views of the room with a user operating our system on a tablet computer.

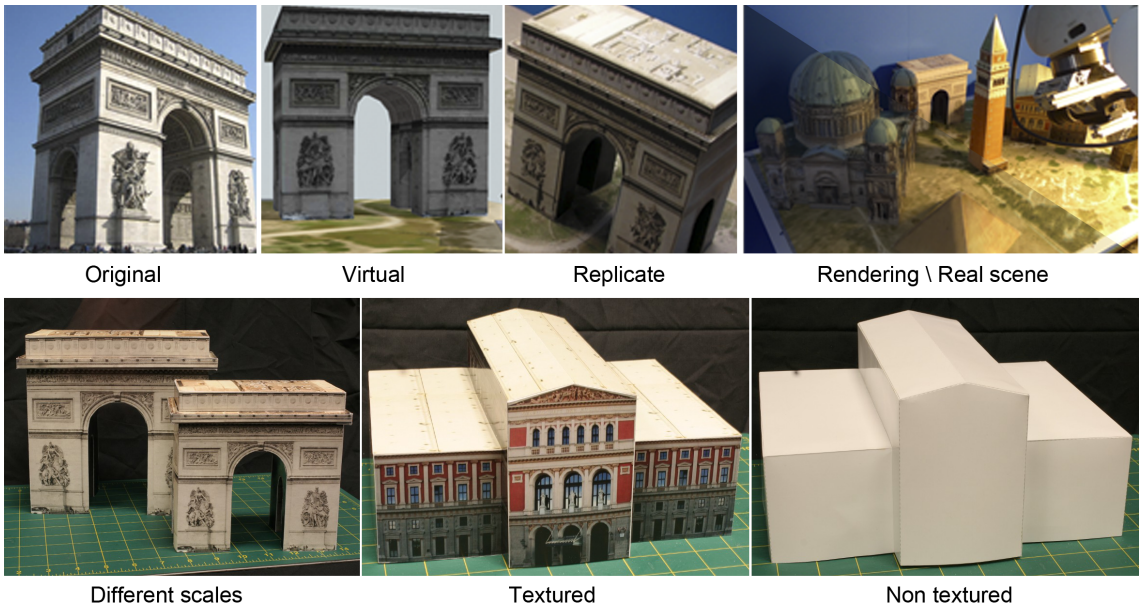


Figure 3.5: The top row shows the original Arc de Triomphe, a virtual model and a miniature paper model replication. On the right of the top row an identical view of the City of Sights, showing a virtual and a real representation of the total assembly beeing overlayed. The bottom row shows different examples how the physical models can be customized.

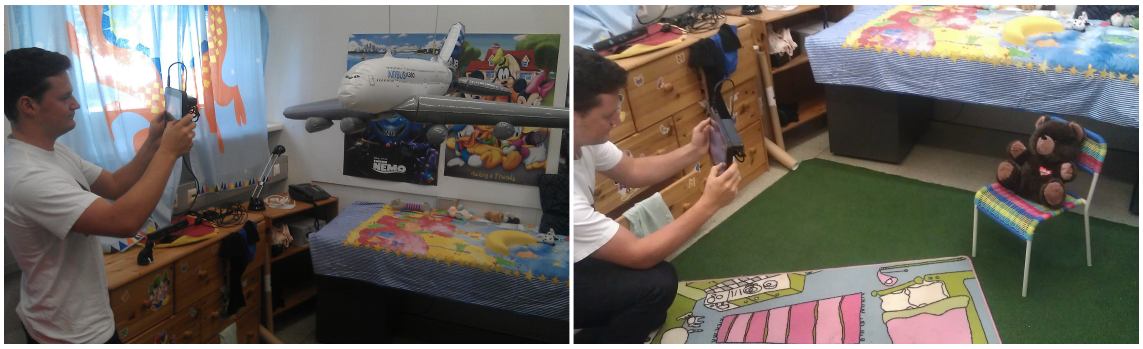


Figure 3.6: Kids' room as example for mid-sized work spaces.

3.4 Conclusion

In this chapter we discussed our system for photometric registration and rendering and its different layers. Moreover we introduced various measuring tools and evaluation environments we designed and which we employed in the following chapters to assess the quality and performance of our algorithms and implementations.

4 Geometry Processing

Contents

Geometry processing involves the reconstruction and preparation of real world geometry for our probe-less photometric registration system introduced in Section 3.1. The knowledge about real world geometry and in particular about its surface normal vectors is essential for the light estimation and also for Augmented Reality rendering. This chapter discusses our two major approaches and their overall impact on our photometric registration system. The first approach (see Section 4.1) is based on reconstructing a virtual model from scene geometry which only covers a small part of the real-world scene. The model is constructed with the help of the user in a pre-processing step and therefore can be classified as a static geometry approach, where the real world geometry is not expected to change after acquisition. The second approach (see Section 4.2) is capable of capturing dynamic geometry in real-time by introducing a geometry reconstruction algorithm which delivers geometry from a combination of depth based volumetric reconstruction of mid-sized environments and fast updating of de-noised depth maps, which allows even capturing fast object movements such as human hands.

4.1 Static Geometry

In this section, we describe how to create a virtual model from static geometry. The overall idea is that the user reconstructs the geometry of an object in the scene with a camera system beforehand. This approach enables the user to choose the geometry for light estimation in the scene and provides more flexibility compared to a solution where the virtual representation of the model - a phantom model - is manually created.

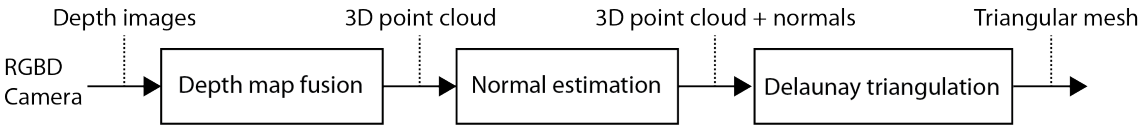


Figure 4.1: This figure highlights the most important stages of the geometry processing pipeline. Depth images from a depth camera (Microsoft Kinect) sampling a real-world object are transformed to a 3D point cloud and finally to a triangular mesh or virtual model. The normal vectors of the model are important for the photometric registration.

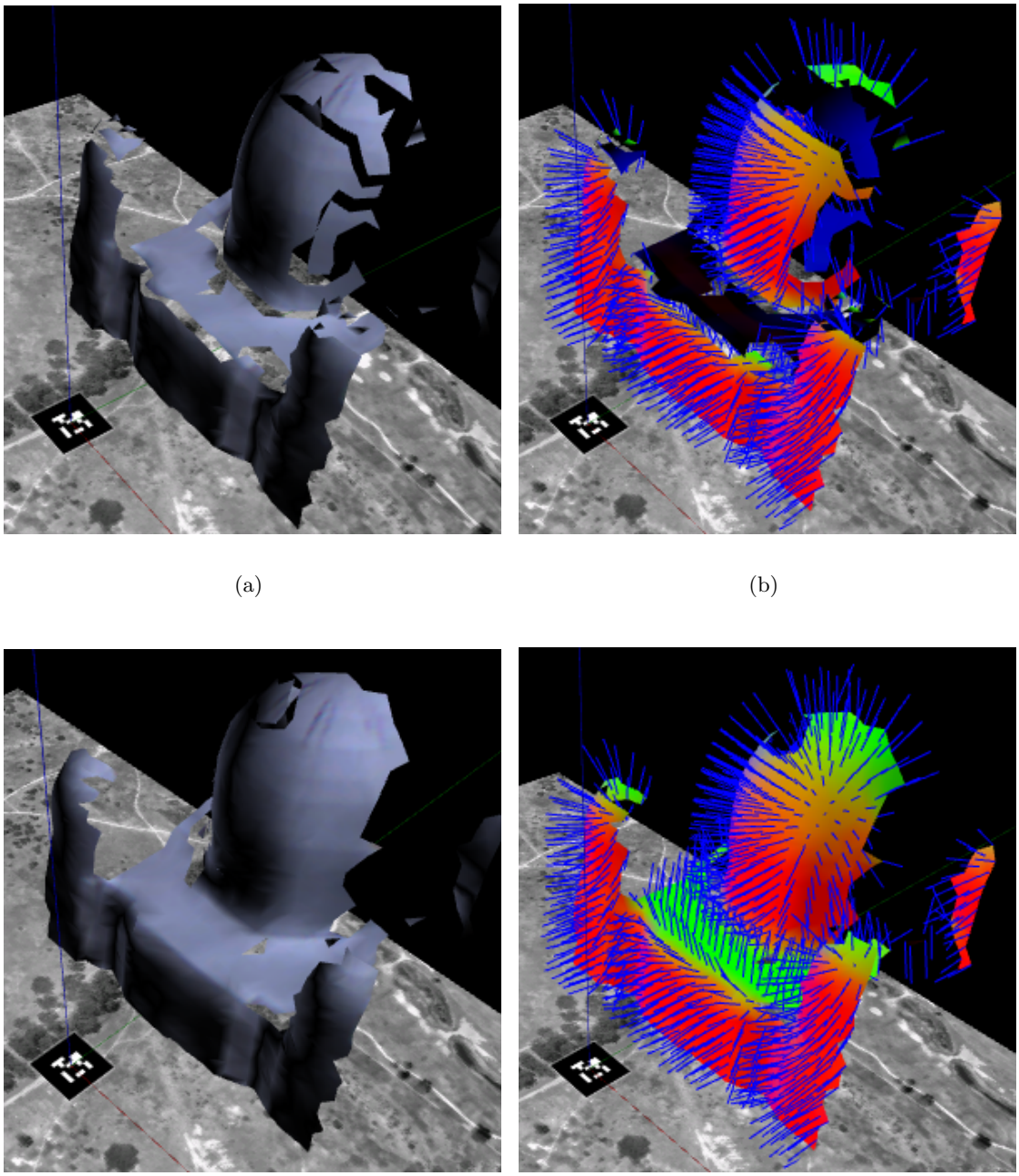
4.1.1 Reconstruction Pipeline

An overview of the detailed pipeline is shown in Figure 4.1. The pipeline can be subdivided into three different stages. In the first stage, depth data is fused in a combined 3D point cloud. In the second stage, we estimate the normal vectors of the point cloud in the third stage, the triangular mesh for the model. In the following, we discuss each step more in detail.

Depth fusion: Our data acquisition step for the model relies on a commercial depth sensor (Kinect), which is able to deliver an RGB image and a depth map for every frame. Multiple depth maps are registered in 3D by tracking the pose of the Kinect device using a vision-based tracker [115] operating on the RGB image. Note that the object of interest must be on a natural feature tracking target for tracking. The resulting 3D point cloud is subjected to per-point normal estimation using a moving least squares method from the Point Cloud Library [102] for extra smoothing.

Normal estimation: The normal estimation algorithm takes the neighboring points in the range of a certain search radius to estimate the normal vector. This algorithm is sensible with respect to the choice of this search radius. If the search radius is too big, the surface model is smoothed too much, and hard edges will not be preserved. If the search radius is too small with respect to the point density, the normal estimation algorithm can fail, and a wrong normal direction is computed. This creates undesirable cracks in the surface. For an example, see Figure 4.2(a,b). To improve the normal estimation, we take advantage of the known viewing direction we have from the pose tracker. It can be assumed that each visible point from the current depth map has a surface normal vector facing towards the viewing vector or, in the worst case, perpendicular to the viewing vector. Normals facing away from the camera are detected by a negative dot product of the viewing vector and the normal vector. The wrong normal vectors are then inverted. Results of this procedure are shown in 4.2(c,d).

Delaunay triangulation: In the last step, we compute a triangular mesh from the point cloud using a fast surface reconstruction method proposed by Marton et al. [76], which is suited for large and noisy input data. The purpose of computing the triangular mesh is to quickly compute visibility for rendering, but also for photometric registration.



(a) (b)

(c) (d)

Figure 4.2: Images (a) and (b) show the results from the surface model estimation without correcting incorrectly estimated normal vectors. Resulting cracks in the surface and incorrect normal vectors negatively influence the lighting estimation. In images (c) and (d), the results of the normal correction are shown.

By visibility we mean the visible surface in the camera view of the triangular mesh has to be determined.

Relying on static geometry for the photometric registration is a fair but severe limitation. Although it allows to compute the real-world lighting per frame it restricts the overall Augmented Reality experience. Therefor we focused in our work on developing a reconstruction pipeline which can handle also changing real-world geometry supporting work areas exceeding desktop spaces. In the following Section 4.2 we discuss the evolution of our dynamic geometry reconstruction approach, which has been become our primary choice of reconstruction method for our photometric registration pipeline.

4.2 Dynamic Geometry

4.2.1 Depth Based Volumetric Reconstruction

The geometry reconstruction and pose estimation is based on the Kinect Fusion algorithm [98]. A 3D voxel volume X is continuously updated with the current depth map from the depth sensor. The result is a surface reconstruction represented as a truncated signed distance function (TSDF) in the voxel volume, together with a 6DOF pose estimation of the camera. To obtain a surface point x and a surface normal vector $\mathbf{n}$ for the current view, we have to perform ray-casting on the reconstructed volume for every frame. Hence, the real-world geometry is represented as an implicit surface, and no polygonal model needs to be computed at any time. We implemented the algorithm on the GPU using CUDA. In Figure 4.3, we show input and output of the Kinect Fusion algorithm.

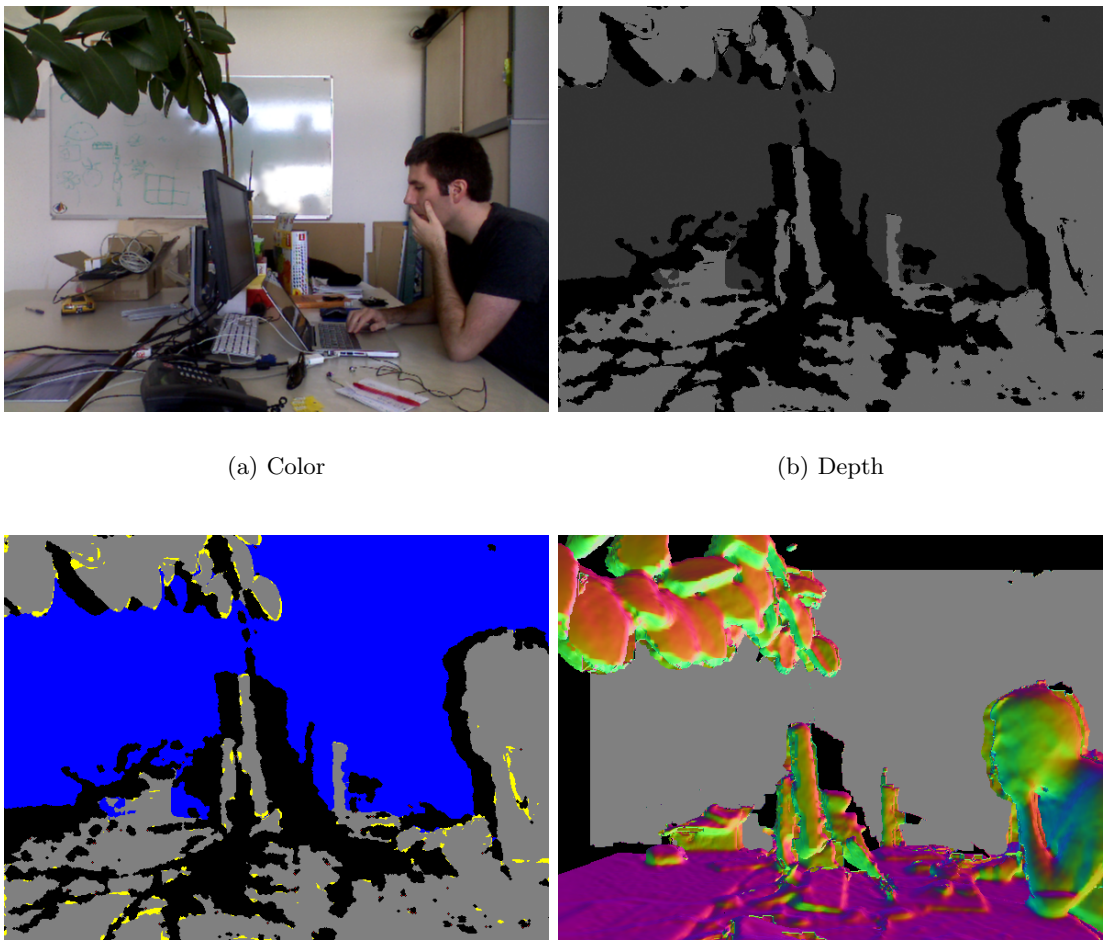
We first employed the Kinect Fusion [98] algorithm in our pipeline at [35, 36]. To improve the algorithm regarding fast geometry updates, we developed a hybrid reconstruction method presented in the following Section 4.2.2.

4.2.2 Hybrid Reconstruction

To react to fast moving objects, we extended the Kinect Fusion [98] volumetric reconstruction algorithm, where the real-world data in the volume X holds static or slowly updated global geometry. Filtering in object-space allows building a global scene model with high quality, and also conveniently provides camera tracking and re-localization abilities. However, fast object motion together with smooth integration resulting in qualitatively better surfaces is not provided in [98].

In our hybrid reconstruction pipeline - an overview of our algorithm is shown in Figure 4.4 - we rather directly extract surfaces from X, using it only as a prior for image-space filtering of the raw depth image D_C. Dynamic information is taken from D_C, while static information is taken from X, which represents the real-world as a volume. This approach can be seen as a variant of spatio-temporal depth filtering, where the temporal prior is taken from the volume rather than from the previous depth image as in [98].

The volumetric prior has multiple advantages over storing only a single depth image. First, the camera motion is already determined during the volumetric integration, so costly optical flow computation for motion compensation is not necessary. Second, in those regions where the image-space filtering relies on information from the volume, higher



(a) Color (b) Depth

(c) Tracking (d) Surface with normal vectors

Figure 4.3: (a) Color image of the Kinect. (b) Depth image of the Kinect. (c) Tracking data visualzation. (d) Surface extracted from the volume and shaded with the surface normal vectors for debug purpose.

quality normals can be extracted. Third, the weights in the volume provide an additional trust measure for the subsequent image-space filtering. The result of the filtering is stored as a geometry buffer G.

In the geometry reconstruction step, the raw data from the RGB-D sensor is processed into a global volume X and a filtered geometry buffer G, which represents the real scene in the current camera FOV. The geometry buffer contains vertex positions, surface normals and color. G and X are used as input to the subsequent light estimation and rendering steps.

The raw depth image D_C contains artifacts, such as holes from missing depth measurements, noise, and poor alignment between color and depth channels. We address these problems by a filter chain designed to separate static and dynamic geometry, while

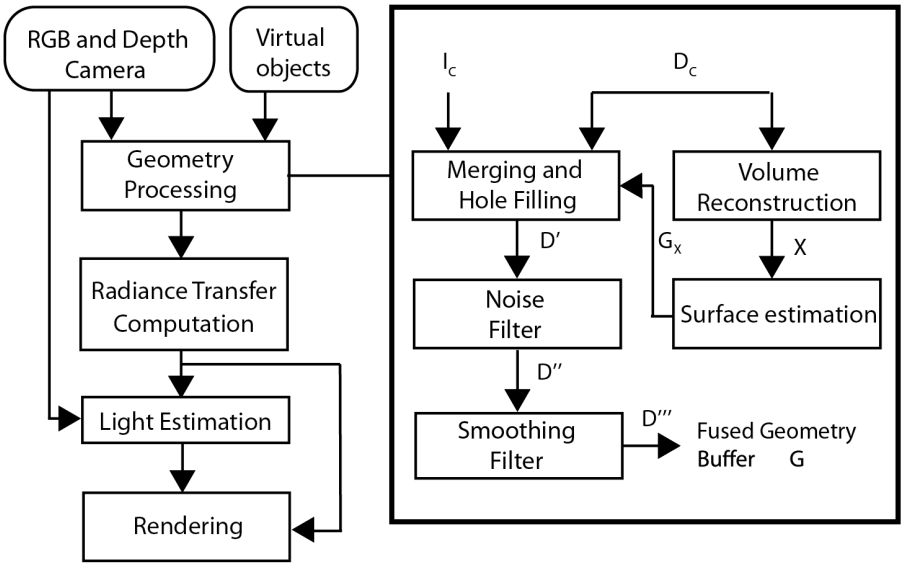


Figure 4.4: Detailed data flow of the hybrid geometry reconstruction pipeline. The environment is captured in volume X, from which a geometry buffer (depth + normals) G_X is extracted and fused with the raw depth map D_C to capture fast geometry changes. The output of the final stage is the fused and filtered geometry buffer G.

suppressing undesired artifacts. The static part of the geometry, X, is obtained by conventional volumetric filtering [44]. This type of filtering is very robust, but discards observations of fast moving objects as outliers.

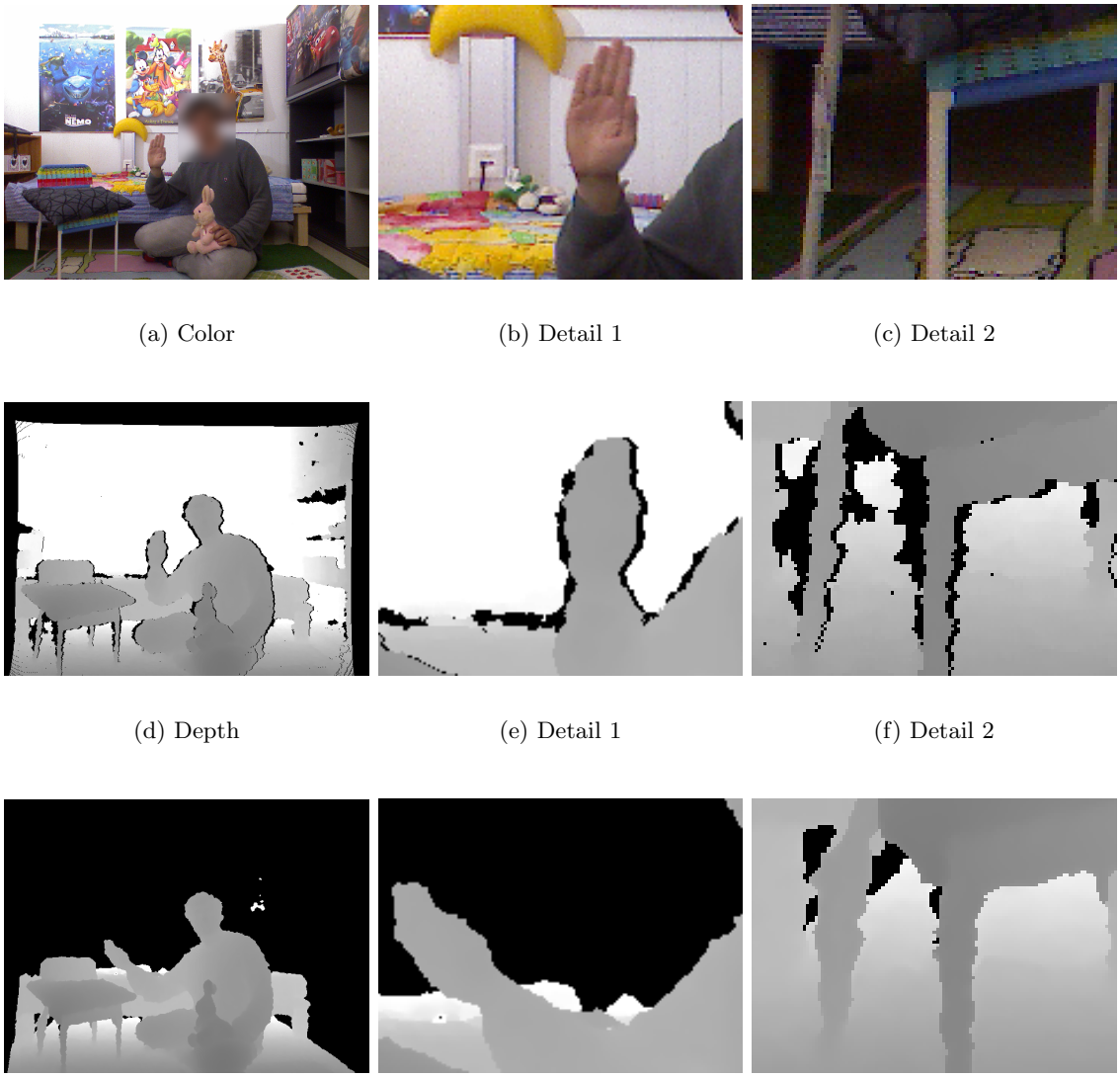
While updating X with D_C, we can extract a geometry buffer $G_X = (D_X, N_X)$ corresponding to the projection of X into the current view. The differences between D_C and D_X will be used to identify where the scene has changed with respect to X. In these regions, the depth information has to be taken from the raw image, even if it has imperfections.

To this aim, three filter passes in image-space are applied. A merging pass fuses D_C and D_X, a smoothing pass aligns depth and color edges, and a denoising pass cleans up erroneous measurements.

Merging and hole filling: The merging pass D' selects among D_C and D_X according to a depth difference threshold λ_D. Non-missing pixels in D_C are directly compared to D_X; if the difference is within the threshold, then the more trustworthy depth D_X replaces D_C. To fill in missing depth pixels, we look at the valid pixels $\Omega_k(p)$ in a $k \times k$ region around a pixel p. We first determine a subset $\Omega_I(p) \subseteq \Omega_k(p)$ of pixels, for which the intensity difference to p lies within a threshold λ_I.

$$\Omega_I(p) = \{q_I \in \Omega_k(p), |I_C(p) - I_C(q_I)| < \lambda_I\} \tag{4.1}$$

This ensures that our support region does not cross an object boundary. We then find the subset $\Omega_D(p) \subseteq \Omega_I(p)$ of pixels for which the depth difference to the volume lies within a



(a) Color (b) Detail 1 (c) Detail 2

(d) Depth (e) Detail 1 (f) Detail 2

(g) Volume (h) Detail 1 (i) Detail 2

Figure 4.5: The two first rows show the input data (Color and Depth) for our filtering algorithm. The second and third column show a detail of the scene. The bottom row shows results of the specific case where the volume reconstruction lags behind the actual depth image from the sensor in the middle row. This can be noticed by comparing the position of the hand in (e) and (h).

threshold λ_D.

$$\Omega_D(p) = \{q_D \in \Omega_I(p), |D_C(q_D) - D_X(q_D)| < \lambda_D\} \tag{4.2}$$

If the majority of inspected pixels in the support region are close in depth to the volume, then the depth value from the volume replaces the depth map input. Otherwise, the median depth of the support region is used.

(a) Mergerd (b) Detail 1 (c) Detail 2

(d) Smoothed (e) Detail 1 (f) Detail 2

Figure 4.6: The middle and bottom collumns show close-up views of the first column. Note the significant improvements in depth edges along the borders of objects, such as the chair legs, after applying our filter pipeline.

$$D'(p) = \begin{cases} D_C(p), \text{ if } \exists D_C(p) \text{ and } |D_C(p) - D_X(p)| > \lambda_D \\ D_X(p), \text{ if } \exists D_C(p) \text{ and } |D_C(p) - D_X(p)| < \lambda_D \\ D_X(p), \text{ if } \nexists D_C(p) \text{ and } |\Omega_D(p)| \geq |\Omega_I(p)|/2 \\ \text{median}(\{D_C(q) : q \in \Omega_I(p)\}), \text{otherwise} \end{cases} \tag{4.3}$$

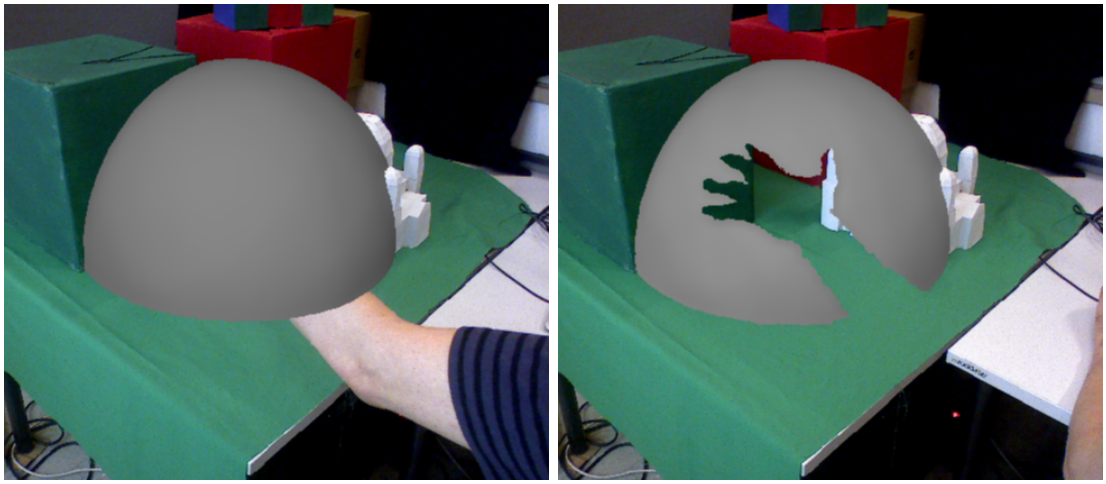
Noise filter: The denoising pass D'' is a joint bilaterial median filter with a smaller window, which operates on all pixels. This pass corrects registration errors between the depth image and intensity image.

$$D''(p) = \text{median}(\{D'(q) : q \in \Omega_I(p)\}) \tag{4.4}$$

Smoothing filter: Because computing a median over a large window is generally costly, we subsample from $k \times k$ to 5×5 before computing the median. This subsampling introduces some noise-related errors, which are cleaned up in the final pass D''' using a small filter window without subsampling. The intensity difference threshold is not needed in this last pass, because we assume that the depth edges are now correct and match the

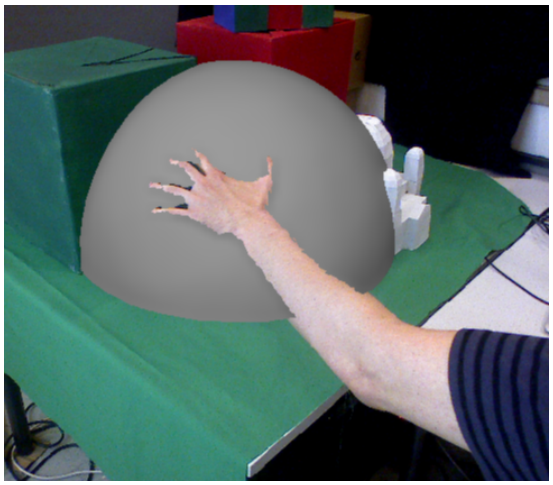
color edges.

$$D'''(p) = \text{median}(\{D''(q) : q \in \Omega_k(p)\}) \tag{4.5}$$

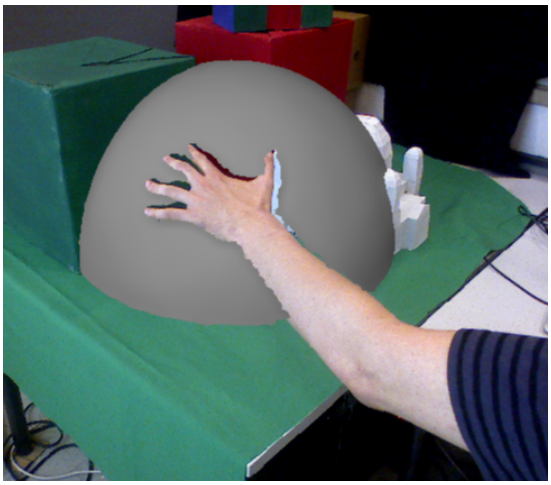


(a)

(b)



(c)



(d)

Figure 4.7: Top row figures show erroneous occlusion handling, where the volumetric reconstruction lagged behind. (a) The inserted arm is not reconstructed in time. (b) After the arm has been removed, the lagging reconstruction still creates wrong occlusions. (c) Our depth merging and filtering algorithm creates correct occlusions. The result of a converged volumetric reconstruction without filtering is shown for comparison in (d).

These filter passes are illustrated in Figure 4.5. A person is waving a hand in front of the camera. Because volumetric integration takes time, the hand is in the wrong place in

the volume. In this case, the merging pass chooses samples from D_C for the hand. Joint color and depth filtering significantly improves depth edges from both the volume and the depth map, as can be observed on the thin chair legs, which are neither well represented in D_C nor in D_X.

Figure 4.7 demonstrates our algorithm for occlusion handling. The two top rows explain the possible errors created by a slower updating geometry reconstruction. In Figure 4.7 (c), the result of applying our merge and filtering algorithm is shown. Figure 4.7 (d) demonstrates the case where only the volumetric reconstruction is used.

4.3 Summary

We showed two different approaches to capture real-world scene geometry for photometric registration. While the first approach was only able to reconstruct static geometry the second hybrid approach based on volumetric reconstruction and depth filtering allows capturing dynamically changing geometry. This is necessary to enable a better Augmented Reality experience. Photometric registration can be performed from different viewing positions and from objects which can freely deform. The different reconstruction algorithm have also a great impact on the subsequent algorithms. In the following chapters, we will introduce the photometric registration pipeline from arbitrary geometry based on the reconstruction algorithms presented in Section 4.2.1 and Section 4.2.2.

5 Photometric Registration

Contents

Applying real-world lighting to the virtual objects is one key requirement of Augmented Reality. Photometric registration in real-time. In this part of the thesis, we explicitly discuss this problem and propose our photometric registration pipeline, which is capable of estimating the environment lighting from arbitrary geometry, making ordinary light probes unnecessary.

This chapter summarizes the work presented in detail by Gruber et al. [36]. Based on the geometry reconstruction algorithm presented in Section 4.2.1, we developed a photometric registration pipeline which solves the inverse rendering problem on a per frame basis. We first discuss the pipeline and data flow in Section 5.1 before we explain the theoretical details of the photometric registration in Section 5.1.2. Furthermore in Section 5.2 we present techniques to improve the robustness of the light estimation. An evaluation based on the prototype developed in [36] proves the applicability of our approach (see Section 5.3). Note that we do not cover the specific Augmented Reality rendering algorithms in detail in this chapter, which is done more thoroughly in Chapter 7.

5.1 Probe-Less Photometric Registration from Arbitrary Geometry

5.1.1 Estimation Pipeline

The light estimation pipeline described in the following order consists of four major stages and is illustrated in Figure 5.1.

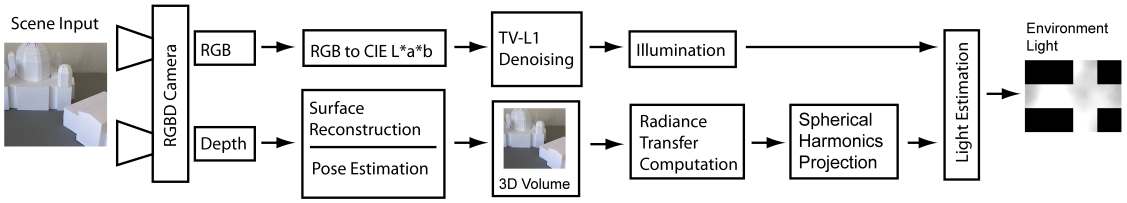


Figure 5.1: Starting from the left, the photometric registration pipeline receives color and depth data from a RGBD camera. Then camera pose and geometry surface are reconstructed to serve as input for the radiance transfer computation, which computes the light transfer for the scene and stores the information in a camera image aligned per-pixel SH buffer. Radiance transfer compressed in SH and the intensity image from the RGB stream (illumination) is required for estimating the actual real-world lighting, which is the final output of the pipeline.

Color and depth image processing: We capture a color and a 16bit depth image from the RGBD camera. Since we are only interested in estimating white colored light sources, we convert the color image from RGB into CIE L*a*b color space to obtain the reflected radiance from the scene. To remove camera noise and high frequency texture parts, we run a TV-L1 de-noising algorithm [120] on the Luminance channel. The color conversion and the TV-L1 algorithm are both implemented on the GPU with CUDA. To improve the registration of the camera image and the depth image, we compensate the radial distortion on both sources, applying a third degree polynomial lens rectification model.

Geometry reconstruction and pose estimation: The geometry reconstruction and pose estimation is discussed in Section 4.2.1. It is important to mention that any geometry reconstruction algorithm which delivers accurate geometry from the real scene with interactive or real-time speed can be employed.

Radiance transfer computation and SH projection: To compute the actual light estimation, we have to model the real-world radiance transfer as realistic as possible. Dynamically moving and deforming real-world geometry precludes pre-computed radiance transfer methods and demands a radiance transfer computation per frame. Depending on the illumination model this can become a very computationally expensive task. The solution of the radiance transfer is then projected into spherical harmonics (SH) basis function. Storing the radiance transfer in world-space is not trivial in our case for two reasons. The geometry could change and this would require the re-computation of the radiance transfer. Another reason is that storing the radiance transfer in SH coefficients (e.g., 16 floats) results in high memory consumption if we store it per voxel. However, this could be levaraged by using a more efficient data structure based on voxel hashing as proposed by [15]. In our approach we use a camera image aligned per-pixel radiance transfer buffer. This buffer stores the coefficients of the radiance transfer in SH per pixel. All radiance transfer computation in our work [35, 36] has been computed on the GPU

using CUDA.

Light estimation: Estimating the light means solving a linear equation system, where the captured scene radiance represented as an intensity image is the observation and the per-pixel radiance transfer represented as SH is the operator. We describe how we setup and solve the linear equation system in more detail in Section 5.1.3. Real-time performance is achieved by partially solving the linear equation system on the GPU [35].

5.1.2 Radiance Transfer

We are interested in a distant light field F, which represents all incident light rays on a hemisphere. Assuming a distant light source implies that we can treat the light source and the lit scene independently from each other. This is an important assumption, which allows us to use spherical harmonics for representing radiance transfer. The principal idea is to recover F from many different observations in one camera image of the scene geometry. The basic relation between the incident light, the observations and the geometry is stated in the reflection equation (see Equation 5.1), as given by Ramamoorthi and Hanrahan [95]. The reflected light field $B(x, \mathbf{w_0})$ is an integrand of the three different terms - incoming lighting $L(x, \mathbf{w_0})$, bidirectional reflectance distribution function (BRDF) $\rho(x, \mathbf{w_j}, \mathbf{w_0})$ and texture $T(x)$.

$$B(x, \mathbf{w_0}) = \int_{\Omega_j} T(x)\rho(x, \mathbf{w_j}, \mathbf{w_0})L_j(\mathbf{w_j})(\mathbf{w_j} \cdot \mathbf{n}_x)d\mathbf{w_j} \tag{5.1}$$

The surface position is denoted with x, and the outgoing direction is $\mathbf{w_0}$. The surface normal vector at position x is denoted as $\mathbf{n}_x$. T represents a simplified texture model with no explicit specular texture. We do not solve the ambiguity of light color and texture color. Instead, we assume that the light color is white, and color contribution comes only from the surface texture itself. The environment light can be dynamic and is assumed to be distant, which implies a homogeneous lighting over the entire surface and hence no local light sources.

We implemented a radiance transfer function for diffuse shadowing, R_{DS} (see Equation 5.2), where $\mathbf{n}_x$ is the surface normal vector and $V(x, \mathbf{w_j})$ is the visibility term for shadow computation. We assume for the entire scene a constant and diffuse Lambertian reflectance modulated with T. This results in the simplified diffuse albedo A. Note that A is neglected in our light estimation algorithm. We interpret A as noise of the reflection, appearing in the camera intensity image. The impact of A is canceled out by robustly solving for the unknown light field:

$$R_{DS}(x) = A(x) \int_{\Omega_j} V(x, \mathbf{w_j})L_j(\mathbf{w_j})max(\mathbf{w_j} \cdot \mathbf{n}_x, 0)d\mathbf{w_j} \tag{5.2}$$

A further reason for computing R_{DS} is differential rendering: Real and virtual illumination effects are combined as discussed more in depth in Section 7.1. In the following Section 5.1.3, we show how we solve for the unknown environment light.

5.1.3 Light Estimation Using Spherical Harmonics

We are interested in computing the unknown lighting terms L_j (intensity) in Equation 5.1 from all surface positions visible in one single view, to further reconstruct F. In the following Equation (5.3), we express F as a reconstruction $\widetilde{F}$ of spherical harmonics basis functions Y weighted by the SH coefficients c over the sphere S. For simplicity, we choose a single index notion of the SH index terms called l and m, where $i = l(l+1) + m + 1$. Moreover, the index k denotes the number of used bands. Obviously, the quality of the reconstruction of F depends on the number of basis functions.

$$\widetilde{F}(S) = \sum_{i=0}^{k^2} c_i Y_i(S) \tag{5.3}$$

The final result of a rendering for a surface point x can be expressed as the dot product of the light source $\widetilde{F}$ and the result of the radiance transfer function $R_{DS}(x)$ at the local surface point x, both projected into SH and represented as a vector of SH coefficients.

$$I(x) = \sum_{i=0}^{k^2} R_{DS}(x)_i \widetilde{F}_i \tag{5.4}$$

Our aim is to estimate the unknown $\widetilde{F}$ from $z \in \{1..Z\}$ reflection observations $I(x_z)$. For estimating $\widetilde{F}$, we stack the observations to a matrix system (see Equations (5.5, 5.6)) and obtain a linear equation system of the form $Ay = b$ of size $Z \times k^2$ with non-square A, which we have to solve for y. Since the system is overdetermined, we minimize the error $|Ay - b|_2$.

$$A = \begin{pmatrix} R_{DS}(x_1)_1 & R_{DS}(x_1)_2 & \cdots & R_{DS}(x_1)_{k^2} \\ R_{DS}(x_2)_1 & R_{DS}(x_2)_2 & \cdots & R_{DS}(x_2)_{k^2} \\ \vdots & \vdots & \ddots & \vdots \\ R_{DS}(x_Z)_1 & R_{DS}(x_Z)_2 & \cdots & R_{DS}(x_Z)_{k^2} \end{pmatrix} \tag{5.5}$$

$$y = \begin{pmatrix} \widetilde{F}_1 \\ \widetilde{F}_2 \\ \vdots \\ \widetilde{F}_{k^2} \end{pmatrix}, \; b = \begin{pmatrix} I(x_1) \\ I(x_2) \\ \vdots \\ I(x_Z) \end{pmatrix} \tag{5.6}$$

5.2 Robust Photometric Registration

Since the quality of the photometric registration heavily depends on the quality of the geometry reconstruction and camera image, which is noisy under real-world conditions, we developed two methods, which improve the robustness of our estimation. The main concern is the selection and interpretation of samples. A sample from the set of visible surface points Z, $M(x_z)$ (see Figure 5.2), is represented by a position x_z in 3D space with a surface normal vector $\mathbf{n}$, the illumination observation from the camera frame $I(x_z)$ and the radiance transfer $R_{DS}(x_z)$ expressed by the SH coefficients.

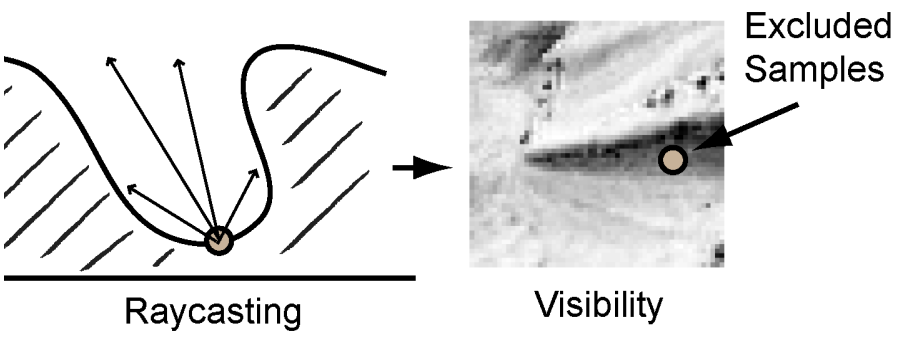


Figure 5.2: We evaluate the visibility of a sample by ray-tracing and exclude samples which potentially can receive only a very little amount of light.

Sample selection by visibility: Surface points can be located in cavities or otherwise heavily occluded areas, where only a small amount of light transfer is potentially possible. From these areas, the measurements will not provide trustworthy data, since we do not model indirect illumination to simulate more complex light transfers. Therefore, we exclude those areas from the measurements. While computing the radiance transfer function R_{DS}, we also determine the visibility term $V(x_z, \mathbf{w_j})$. By doing so, we can measure the occlusion by casting a ray from each sample point into all possible directions $\mathbf{w_j}$ and determine if there is any occlusion from another surface in this direction. If a sample point has too much occlusion (e.g., 95 percent of the rays hit another surface), it is excluded from the lighting estimation (see Figure 5.2).

Weighting by surface normal vector distribution: The ideal surface for estimating the environment illumination would be a sphere, which provides observations from all possible directions. Unfortunately, such a uniform distribution of normal directions is not always naturally given in the real-world. In practice, the scene often consists of a large planar surface (e.g., a table) with a single dominant normal vector. Naturally, samples taken from this surface will have a large impact on the final estimation. Other normal directions will not be sufficiently represented. For example, samples taken from smaller objects in the scene, which have a smaller pixel coverage in the video frame, will not yield a comparable number of samples. To improve our estimation, we therefore weight the samples according to a uniform distribution U of surface normal vectors on a sphere (see Figure 5.3). We bin the samples according to their surface normal vectors into a regular spherical grid. For each bin e in the grid, we compute a weight m_e (see Equation 5.7), where E is the number of bins of the grid and Z is the total number of samples. The weight m_e is the number of samples Q_e in a bin divided by the uniform distribution $U = \frac{Z}{E}$.

$$m_e = \frac{Q_e}{U} \tag{5.7}$$

In the following, we multiply the illumination value $I(x_z)$ and the radiance transfer $R_{DS}(x_z)$ with the corresponding weight m_e. The results are then inserted to the linear

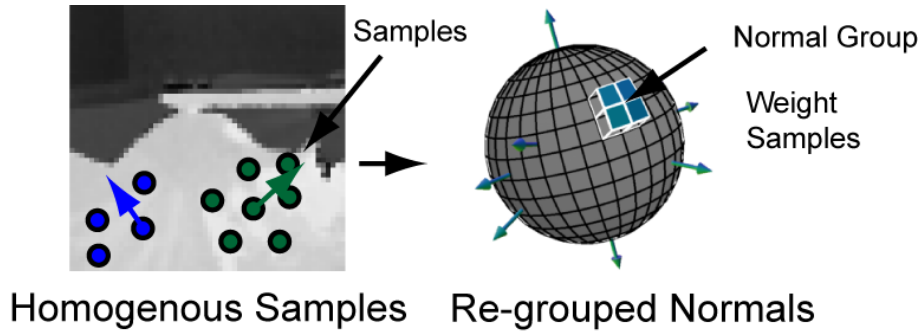


Figure 5.3: We group samples according to their surface normal vectors by a regular grid representing the surface of a sphere.

equation system in Equation 5.6. The weighting will have the following three effects. First, samples from overrepresented areas are normalized by the uniform distribution. Second, samples, which are represented uniformly over a sphere, are not changed. Third, the influence of samples, which are not strongly represented, will be diminished. The benefit of this method can be seen in the real-world experiments in Section 5.3.2.

5.3 Evaluation

In the following, we present the evaluation of our algorithm. We will mainly focus on plausible and perceptively convincing results, rather than on physically correct results. In our evaluation, we used props from the AR stage set "City of Sights" from Gruber et al. [34]. For SH computation, we used four bands, resulting into 16 coefficients.

5.3.1 Synthetic Light Estimation Evaluation

As a first test, we load a known light source, represented as an HDR environment map, and use it to shade the geometry of the reconstructed scene. The output is an image of the rendering which simulates an image coming from the camera. We use this then as input to estimate the synthetic light source. The estimated result should be comparable to the known synthetic input light. The result of the light estimation is given in SH and can be projected into a cube map for one to one comparison to the synthetic input.

The results of this test are shown in Figure 5.4. In this evaluation we took HDR environment maps from Paul Debevec*. Note that in the interpretation of these results, the quality of the geometric reconstruction, for example, the accuracy of surface normal vectors, must be considered. The major conclusion of this test is that the light estimation is working correctly with real-world data from the geometry reconstruction.

5.3.2 Real World Results

In Figure 5.5, we show the results of a systematic test series with three different scenes (A, B and C) and four different lighting situations. All three scenes have been exposed to the

*http://ict.debevec.org/~debevec/Probes/

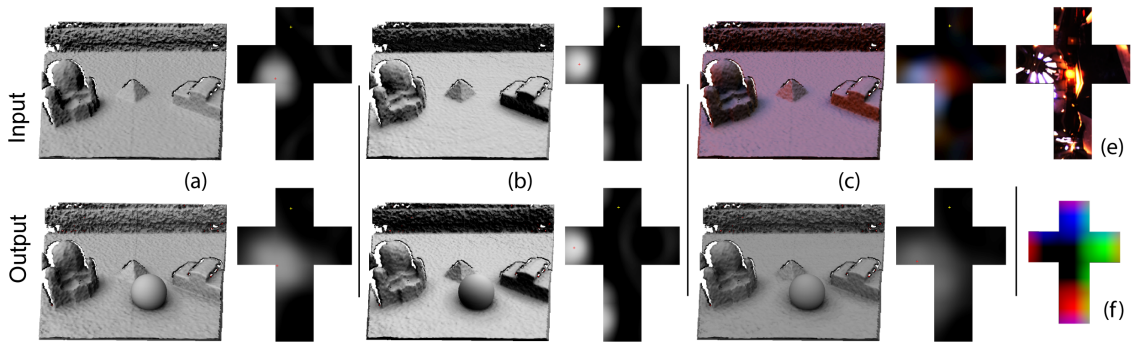


Figure 5.4: This image sequence shows results from the synthetic light reconstruction. The top row (Input) shows the entire real-world scene lit by the synthetic light. The rendered scene is then used as input for the light estimation algorithm. The light source is shown on the right of the rendering, represented as a cube map. The brightness of the cube map has been adjusted for better display in this document. The lower row (Output) shows the real-world scene plus a virtual sphere lit by the reconstructed light source. The solution of the light source reconstruction is shown on the right of the augmented rendering. In examples (a) and (b), we used a light source traveling from left to right. The effect is best observed looking at the pyramid. In example (c), we used the commonly available "'Grace Probe"' (e) HDR map from Paul Debevec. This light source is more complex. Note that we do not estimate the color of the light, therefore the result is also in gray. Image (f) shows the color encoded directions of the environment map.

same four different lighting situations created using a bright directional light source, which changes direction through the series. Note that the light source is not optimally diffuse and creates hard shadows from the real objects. So far, we did not model hard shadows in our system and estimate only low frequency lighting. Therefore the shadows from the virtual objects will appear more soft and blurred. In Figure 5.6, we show the effects of applying our robustness method presented in Section 5.2 on the more complex test scene A. The results show that the robust approach creates a better lighting estimation, which results in a perceptually better rendering. The tests shown in Figure 5.5 demonstrate that our system also works with colored objects, although we do not estimate any material properties. Scene B and C have identical geometrical properties and differ only in the surface color. While the objects in scene C have various colors, the objects in scene B are uniformly gray. Each result is accompanied by the visualization of the lighting estimation as in an unfolded cube map. For comparison, each series has been processed by the naive approach and the robust approach.

5.3.3 Influence of Occlusion

In this section, we discuss the influence of occlusion or self shadowing on the lighting estimation. The occlusion is computed through the visibility test $V(x, \mathbf{w_j})$ in equation 5.2

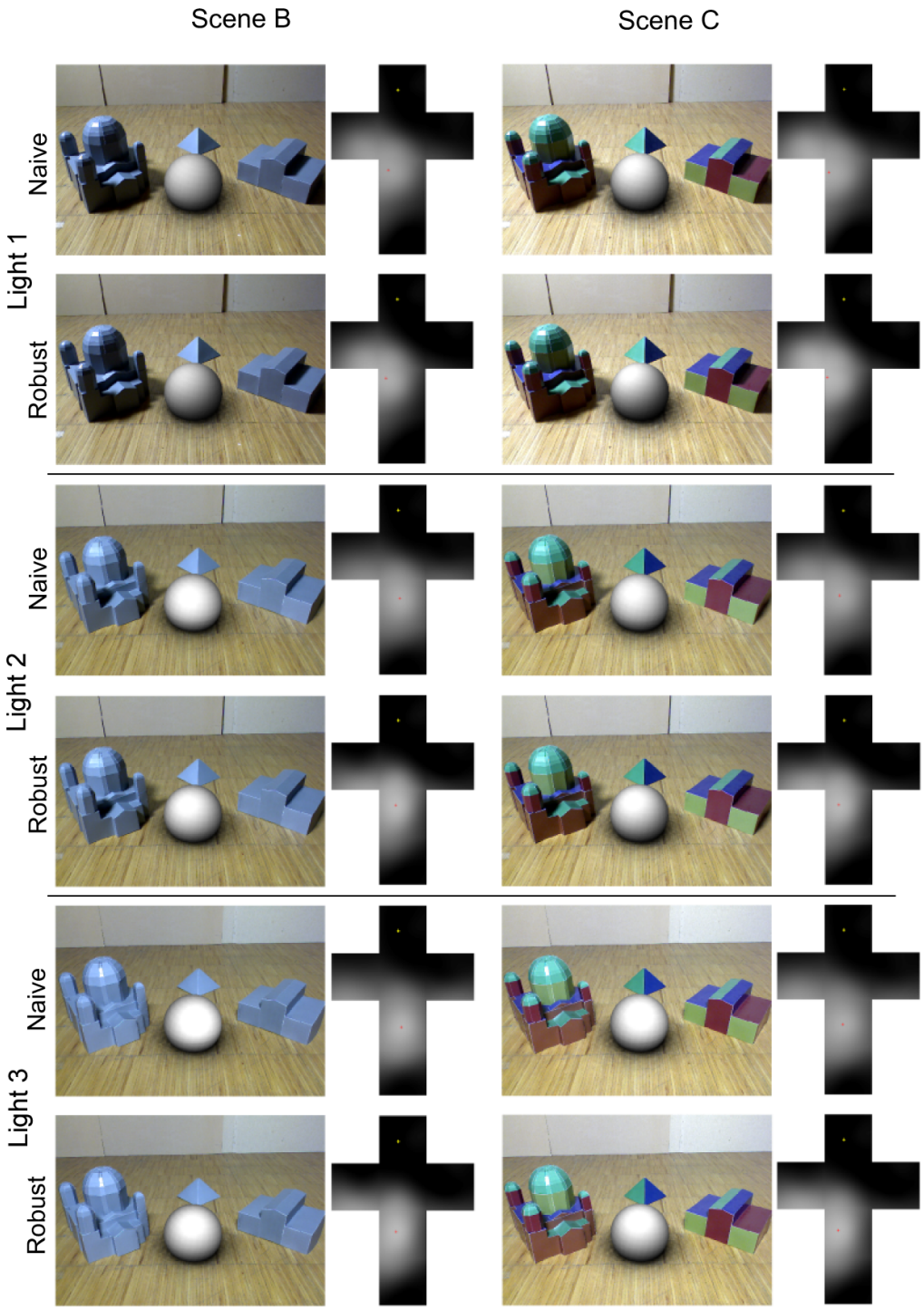


Figure 5.5: This Figure depicts the real-world results of the two scenes B and C, differing in surface colors. Note that there are very little differences in the photometric registration.

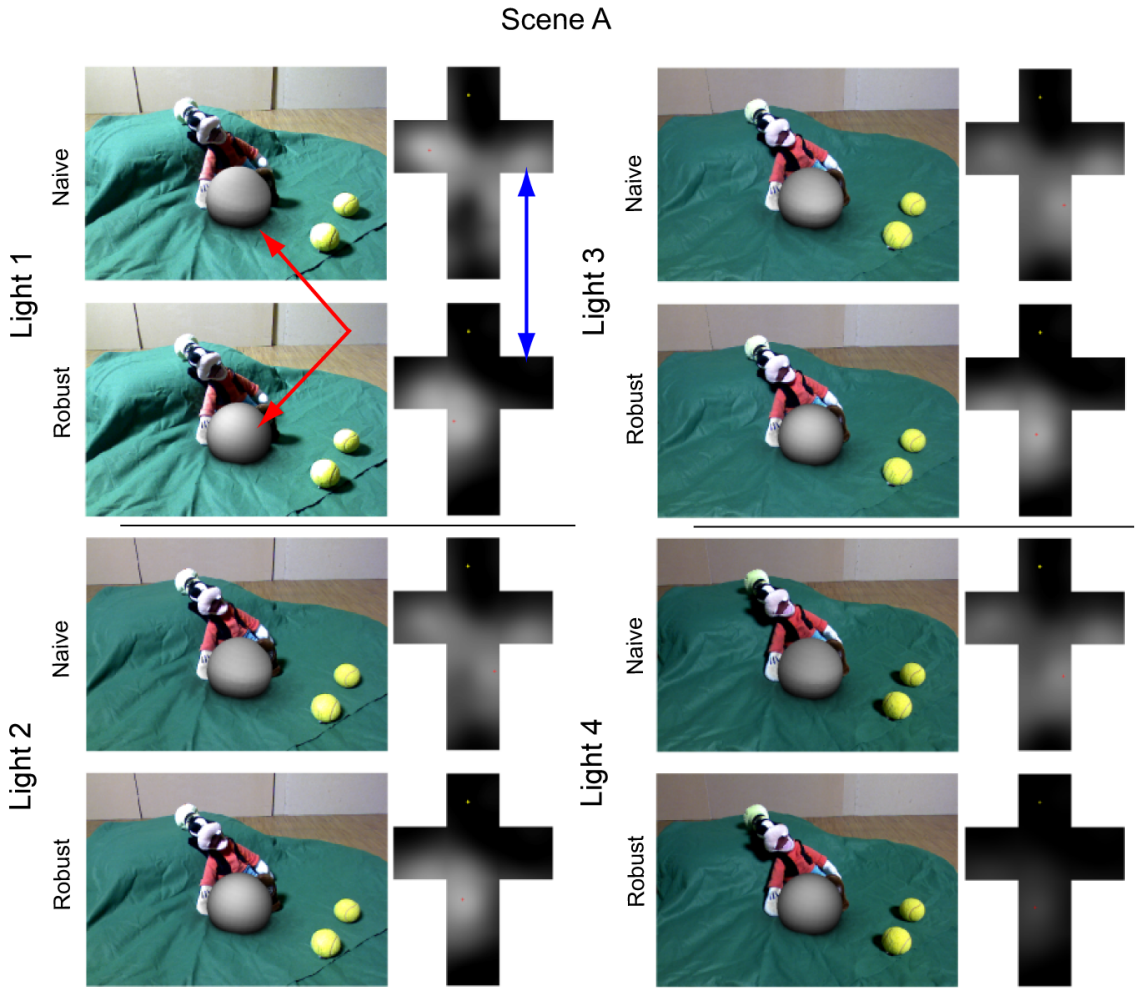


Figure 5.6: The red (scene space) and the blue (illumination space) arrows mark visible difference of the naive and robust approach. We can observe that the robust approach creates more visually pleasant results than the naive approach.

and practically depends on the length of the ray of the ray-tracing step. A longer ray will increase the probability that occlusion from objects that are further away is incorporated into the visibility estimation. For the lighting estimation, we only consider the occlusion of the real-world objects. We increase the ray length from 0 to $0.2m$ to evaluate the influence. As can be seen in Figure 5.7, the solution of the lighting estimation improves with the ray length. The solution is more stable, which can also be noticed in the AR rendering. This is due to the fact that the real-world is modeled more accurately with occlusion than without. The optimal ray length depends on the entire volume of the scene. In our case, the ray length is given in meters, and the entire scene has a volume of $2 \times 2 \times 2$ meters. Note that compared to view dependent methods, the consistent volumetric reconstruction in combination with ray-tracing is more exact. However, in Section 6.2 we will prove that also approximations of the radiance transfer computation in image space create valuable

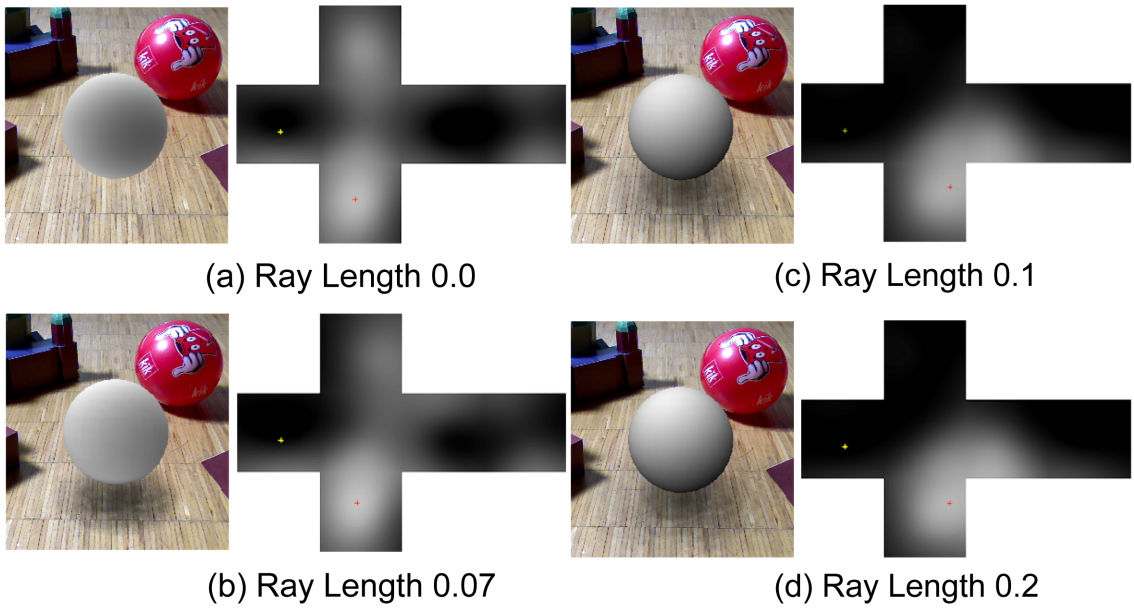


Figure 5.7: We increase the visibility ray length (in meters) from (a) to (d). Comparing the solution of (a) and (d), a noticeable improvement of the estimation can be seen.

results.

5.4 Summary

In this chapter we presented the a general photometric registration pipeline and the results of the implementation in [36]. While the general light estimation approach discussed in Section 5.1 is valid for the entire work of this thesis, the actual implementation shows several limitations. In the following we summarize the main limitations of this pipeline.

Naturally the light estimation also depends on the characteristics of the reconstructed surface. There are surfaces which are problematic. For example, the reconstruction and tracking algorithm does not handle planar surfaces well resulting in drift, and the light estimation algorithm works better on curved surfaces. Moreover, surfaces with strong reflectiveness violate the assumption of diffuse surfaces. The quality of the light estimation also depends on the distribution of the surface material colors. For instance, if a scene is divided into a black and a white area, the algorithm would likely produce wrong results. We support only diffuse shadows and lighting in our estimation and rendering approach. No hard shadows are modeled. Furthermore, as already stated in Section 5.1.2, we do not estimate the color of the real-world surface or the reflection parameters. Therefore we assume a white real-world light source. Moreover a general limitation valid for systems using depth data from a depth sensor based on structured light as the Kinect is the fact that depth maps from specular surfaces, e.g., mirror balls or other shiny surfaces might not provide reliable depth information. This creates holes in the reconstruction and makes the photometric registration from these areas impossible.

In [36] we estimate the lighting at a per frame basis and do not take previous results into account. Although this allows the estimation of fast dynamically changing lights, it can also lead to unstable results. This implementation could suffer from flickering artifacts. In [35] we added a moving average filter over the final 10 light estimation results, which solved the problem.

The reconstruction algorithm [80] employed in [36] has a great impact on the overall result. Although this algorithm improves the quality of the surface reconstruction (surface normal vectors, surface consistency etc.) over time, which is important for the photometric registration, fast changes in the scene do not take effect immediately. Adding and removing objects is possible, but it will take some frames until the entire scene is updated correctly. Moreover, the reconstruction of the scene might be incomplete unless the user makes an effort to scan the scene properly, e.g., by walking around the scene. Additionally the main performance bottleneck in [36] is the computation of the radiance transfer through volume ray-tracing including the visibility test. However, in [36] we achieve interactive frame rates of 5Hz performing the computation on every 4th pixel with a ray length of 10 cm, a total working volume of 2x2x2 m, 16 SH coefficients (4 bands) and 169 rays covering a sphere for visibility testing on commodity hardware (GeForce GTX 580). In the following Chapter 6 we present the evolution of different acceleration methods which improve the performance and demonstrate that the hybrid geometry reconstruction approach discussed in Section 4.2.2 is a valuable alternative to the Kinect Fusion geometry reconstruction algorithm for photometric registration.

6

Acceleration Techniques for Radiance Transfer

Contents

In the previous Chapter 5, we introduced the general photometric registration pipeline and identified radiance transfer computation as the major bottleneck with impact on the overall performance. Depending on the radiance transfer function, radiance transfer can become expensive very quickly, especially if one ore more light bounces are computed. As already discussed, we do not aim for a pre-computed solution, where the radiance transfer is pre-processed and stored in object space on a pre-built static geometry model of the real-world (see Section 4.1). This is a very common technique in real-time graphics and provides high quality results enabling real-time performance [108].

Since we assume that real-world geometry can move and deform rapidly from one moment to the next, we were interested in solutions which can handle dynamic geometry and meet the requirements for photometric registration and Augmented Reality rendering at the same time. In the following sections we propose several solutions to improve performance. We implemented acceleration techniques for volume ray-tracing based radiance transfer computation, which are based on sub-sampling in image and visibility space (see Section 6.1.1), and visibility caching in world space (see Section 6.1.2). Furthermore, we investigated radiance transfer approximation in image space (see Section 6.2). At this point it is important to recall that the computed radiance transfer is fundamental for the photometric registration, but also for the final Augmented Reality rendering and, therefore, the acceleration methods can have impact on both, light estimation quality and rendering quality. Moreover, the proposed acceleration techniques are naturally tightly coupled to the underlying geometry processing methods discussed in Chapter 4.

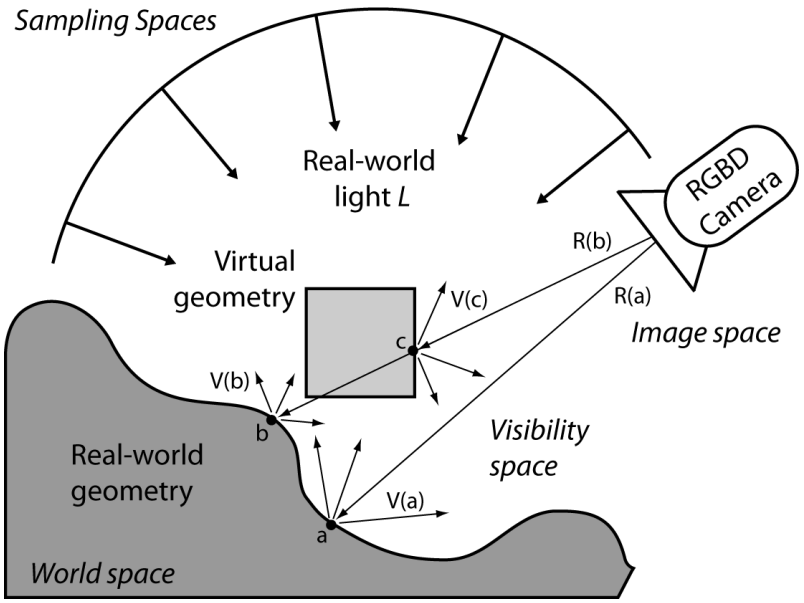


Figure 6.1: This figure explains the different sampling spaces used in our acceleration methods. These are the world space, in which both the real-world and the virtual world exist, the image space, which is the projection of the world space into the camera on the image plane and the visibility space, which describes the space sampled by the visibility rays. The radiance transfer is illustrated by the rays R which are cast from the camera image space into the world space, where they can hit real-world surface points (a,b) and virtual surface points (c). From this point, visibility rays V sampling the visibility space are sent out to test against occluding geometry. Note that for computing solely the photometric registration, only the radiance transfer computed from the real-world geometry is needed. Computing the radiance transfer for the real-world geometry and the virtual world geometry is necessary for Augmented Reality rendering.

6.1 Accelerated Volume Ray-Tracing Based Radiance Transfer

6.1.1 Joint Image and Visibility Space Subsampling

Dense radiance transfer is computed for each pixel or sub-pixel. Depending on the radiance transfer function, radiance transfer can involve several indirect light bounces. This approach is well known as ray-tracing. In our case, we send rays through a volumetric representation of the scene geometry. Since dense radiance transfer sampling is not computationally feasible, we resort to adaptive subsampling and upsampling. Applying sampling mechanism to accelerate ray-tracing is a well known approach in computer graphics, where the overall goal is forward rendering or image generation [109]. In this work, we developed and evaluated sampling methods which are suitable for photometric registration solving the inverse rendering problem and forward rendering problem at the same time.

In Figure 6.1, we visualize the different sampling spaces we address. We define the world space as the space where real-world geometry and virtual geometry co-exist. Furthermore the image space is defined as the projection of the world space into the camera image. The third and last sampling space is the visibility space, which defines the space sampled by the visibility rays, which are part of the radiance transfer computation.

In this section, we introduce an acceleration method which adopts different sampling regimes for image space and visibility space. Since we need to compute the visibility signal V from Equation 5.2 only for currently visible surface points x, we can express it as a 4D function $K(r, s, \phi, \theta)$, where a surface point x is projected onto a pixel (r, s) in image space, and (ϕ, θ) encode the direction of the visibility ray in polar coordinates. The joint image and visibility space subsampling algorithm can be divided into the following four major steps, which are illustrated in Figure 6.1.

Step 1 - Adaptive image space subsampling: Subsampling in image space incorporates two major constraints. The first constraint is that photometric registration requires a regular sampling in image space to evenly measure the reflection of the entire scene as well as possible. We regularly subsample every n^{th} pixel. While this works well for the light estimation, it introduces visible aliasing artifacts in the rendering. As explained in Section 3.1, the computed radiance transfer for the real-world geometry is reused in the rendering step. To improve rendering quality we add dense radiance transfer sampling in areas where interesting light interactions between virtual and real objects are expected. For virtual content, these are non-occluded pixels close to a depth or normal discontinuity, where we expect illumination gradients to occur. We determine these pixels by computing edges in the depth and normal buffer of the virtual geometry. For real-world geometry, pixels affected by virtual geometry in differential rendering must be densely sampled. We determine these pixels by marking surface points hit by visibility rays starting at virtual objects.

Step 2 - Interleaved visibility subsampling: We exploit the limited rate of change in environment lighting by interleaved subsampling in visibility space. Since the visibility space is established relative to world space, and visibility is stored in a bandwidth-limited SH representation, the result is not as susceptible to error in tracking and noise in surface reconstruction, as would be the case with the more common spatially interleaved sampling. For every sample point, we evaluate only a subset of all ray directions per frame. More precisely, for every chosen pixel (r, s) and every k^{th} frame, we compute a different subset of N/k rays out of N possible ray directions. By interleaving the ray directions, each subset evenly samples the entire visibility space. The visibility testing results of the current subset are then cached in an image space aligned 2D cache. After k frames, the previous subset of N/k rays is replaced by new results to refresh the cache. Overall, this results in k caches holding the complete visibility space. The next step explains how we recover the entire visibility space in every frame.

Step 3 - Reprojection and visibility fusion: Since the camera moves, the visibility caches are misaligned. To correctly access each of the k visibility caches for one current

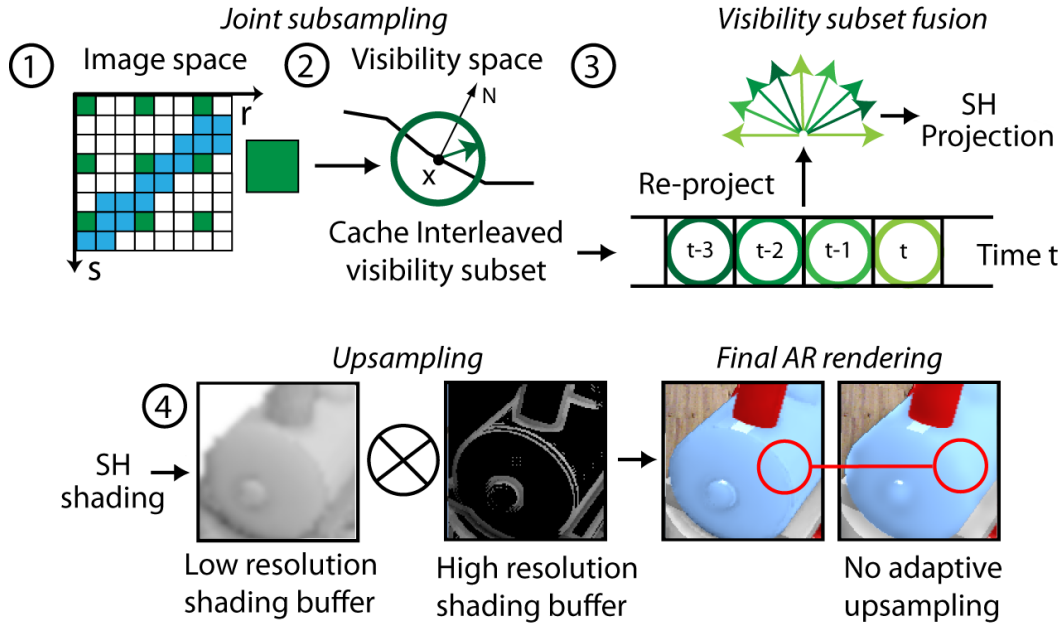


Figure 6.2: Starting from the left, we see the adaptive sampling pattern, consisting of regular subsampled (green) pixels and densely sampled pixels (blue). For all green pixels, we compute an interleaved subset of visibility rays over time. The subsets are stored in a 2D cache and fused by re-projection for every frame. The complete RT signal is projected into SH for light estimation and shading. Due to regular subsampling, the shading map has a lower resolution than the final AR rendering. Therefore, we apply upsampling for the visible augmentations to increase the rendering quality. The visual improvement (highlighted by the red circles) can be seen in the right most part of the figure.

pixel, we reproject the associated 3D surface point into the camera coordinates associated with each 2D cache. Cache entry association after reprojection is based on a nearest neighbor lookup. The subsampling in image space used in step 1 leads to increased robustness of the reprojection, since we project from a higher resolution into a lower resolution 2D cache. As common with reprojection, the depth buffer from the previous frame is used to weed out disocclusions or dynamic scene changes. If a threshold depth difference for the sample point is exceeded, the cache entry for the sample point is discarded and rebuilt.

Step 4 - Upsampling: The final upsampling to full image resolution is performed in a deferred rendering step. Shading for areas with low resolution and high resolution radiance transfer (edges) sampling is computed separately and combined with the color buffer resulting from rendering unshaded virtual content over the video background.

Results of our acceleration method are shown in Section 6.1.3. In the following Section 6.1.2 we introduce a complementary method which caches visibility information in world space.

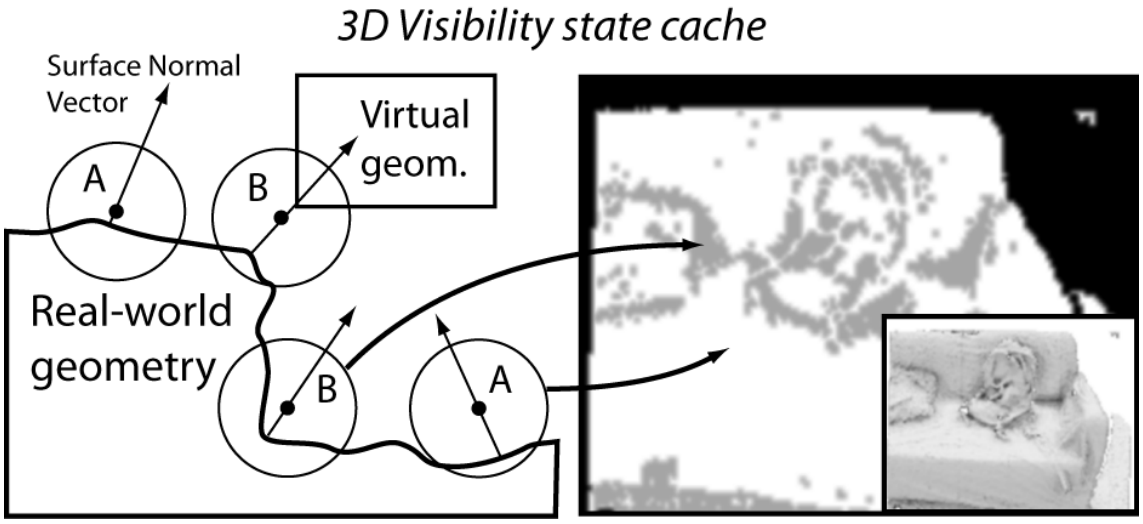


Figure 6.3: Left: Reconstructed surface area with noisy surface normals and virtual geometry. (B: occluded,A: non occluded). Right: Example of the visibility state cache (grey: occluded, white: non occluded). Right lower inset: shaded buffer of the real-world geometry.

6.1.2 Robust Visibility Caching in World Space

The world space is the natural space of the 3D reconstruction, which suggests to adopt it also for radiance caching (per voxel). However, implicit surface reconstruction relies on a probabilistic surface model, i. e., multiple consistent observations of a surface point are required to filter sensor noise, which makes it unreliable for caching radiance transfer directly since the surface constantly updates and changes. To handle dynamic scene changes, we cannot simply turn off the reconstruction updates.

A second characteristic of an implicit surface compared to the polygonal models commonly used in real-time global illumination is that the spatial resolution is much lower than the useful 3D resolution at which the implicit surface can be sampled. This has the consequence that multiple pixels (r, s) in image space may be associated with a single voxel of the 3D volume, especially if the camera is close to the surface. As a consequence, any information stored in world space per voxel is stable with respect to camera movement and tracking errors, but at the expense of low image space resolution. Again, storing radiance transfer per voxel in world space is not suitable, as it would suffer from the insufficient resolution.

We therefore built the radiance transfer cache in image space and combine it with a visibility state cache in world space. Visibility state is a flag that determines if a certain surface point is free from occlusion and therefore need not be considered for visibility testing.

Cache fill: The visibility state is evaluated statistically. If no more than a certain small fraction of the visibility rays is blocked, we assume that the surface point x is unoccluded and store this information. If we evaluate this surface point again in a subsequent frame,

we compute only the Lambertian term without sending visibility rays to test for occlusion. Since the maximum ray length considered for the visibility rays is limited, this optimization is assumed to remain valid until the reconstruction changes in the neighborhood of the surface point. Only one Boolean is required to store the visibility state, so this cache has a negligible memory footprint, which is important since we store the information per voxel.

Cache update: Updating the cache is more difficult, since it requires observing changes in 3D. This problem is handled by approximating the 3D solution with multiple 2D projections. We achieve this by using one or more cache validation cameras (CVC), which obtain depth maps from ray casting into the reconstruction volume. The depth map from a CVC is compared with a stored one to determine invalid cache entries, where the surface has changed. An invalid cache entry is simply removed to trigger resampling at the earliest convenience. CVC placement can be strategically chosen to cover all relevant changes in the volume. We usually place one CVC overhead for observing the whole volume at optimal resolution, and another one as a dynamic view-dependent camera following the user. Note that a dynamic camera must compute two depth maps (from the old and the new point of view) using a ping-pong buffering scheme.

The following evaluation (Section 6.1.3) provides results for all the previously discussed acceleration methods.

6.1.3 Evaluation

6.1.3.1 Evaluation Pipeline Details

We use the system presented in Section 3.1 with a RGBD camera (Microsoft Kinect) at a resolution of 640x480, which is also the final image resolution. The dimensions of the working volume are 2x2x2 meters with a voxel resolution of 268 cubed. For the visibility sampling, we used in total 81 ray directions and a maximum visibility ray length of 10 cm. We empirically determined a sample spacing in images space of 4 pixels and a temporally interleaved sampling interval of 4 frames. Estimation of the environment light: Environment light is estimated in SH form using four bands (16 coefficients). The matrix solver combines CUDA with CPU computation, but could be ported to the GPU entirely. However, the performance impact is negligible. The Augmented Reality rendering pipeline pipeline (discussed more in detail in Chapter 7) is based on deferred shading implemented in CUDA and OpenGL shaders.

We compare different parametrizations of our approach against a reference method with full sampling in image and visibility space ('RM') [36] and regular subsampling in image space with a sample spacing of 4 ('REG4') and 8 ('REG8'). Joint subsampling methods have sample spacings in image space of 1 ('ART1'), 4 ('ART4') and 8 ('ART8'), while sample spacing was set to 4 in visibility space. We use multiple real-world sequences recorded with the RGBD camera as test data sets for performance and rendering quality evaluation. Additionally, the light evaluation requires some form of ground truth. Real-world lighting, such as obtained from a spherical mirror ball, would add additional errors. Therefore, we use synthetic light sources as ground truth. We substitute the input RGB image from the camera by renderings of the real-world geometry lit by the synthetic light

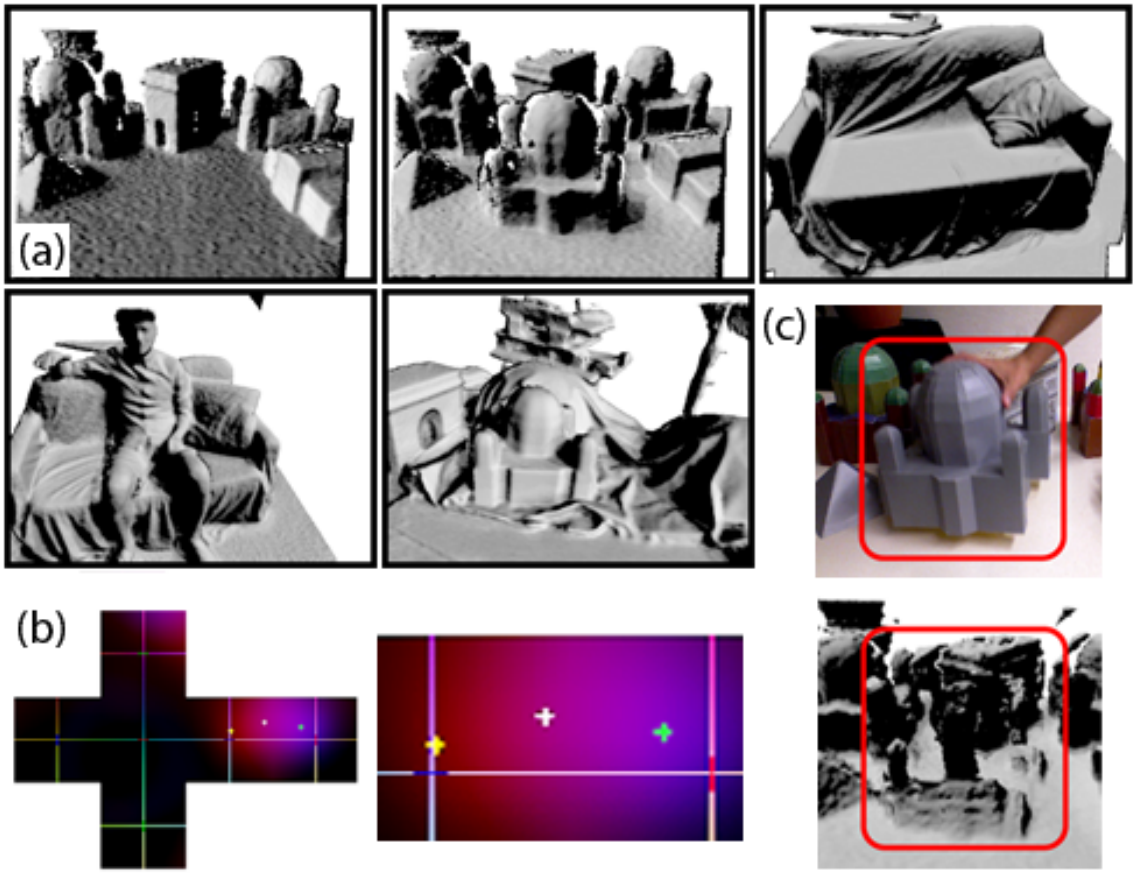


Figure 6.4: (a) Data set frames lit by synthetic light source. (b) Environment light map: estimated (red), synthetic (blue) and difference (violet). Dominant light directions: green (Synth. light ground truth), white (estimate) and current camera view direction (yellow). (c) Test scene with dynamically inserted objects (red rectangle). Top: RGB input video. Bottom: reconstructed volume. The reconstruction lags behind by a couple of frames. This naturally results in a misalignment of the radiance transfer and the reflection observed from the camera image.

sources similar to the evaluation method in Section 5.3. In total, we use five different sequences, each 300 frames long, containing dynamic camera movements and dynamically changing geometry. An overview of the scenes and an example for dynamically changing geometry is shown in Figure 6.4(a) and (c).

6.1.3.2 Performance

We evaluate performance by measuring the computation time of each single frame and report this in frames per second (FPS). The performance mainly depends on the radiance transfer computation, but other parts of the pipeline can have an effect, such as changing geometry. Figure 6.5 reports the FPS median over all test sequences. As expected, RM is

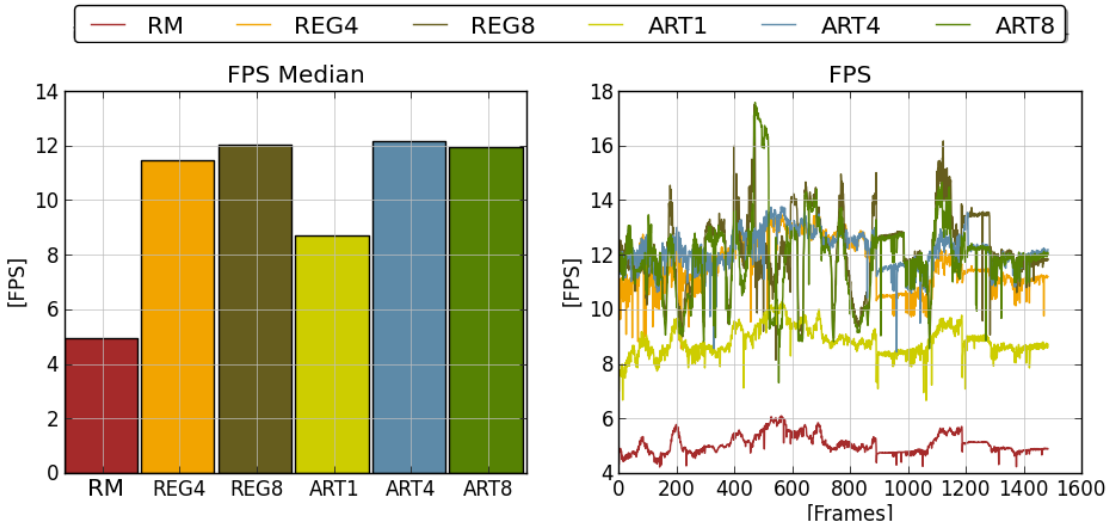


Figure 6.5: Median FPS (Left) and FPS (Right) over all frames. Timings in ms for RT and light estimation: RM 203.5, REG4 88.886, REG8 83.385, ART1 117.06, ART4 85.306, ART8 79.827

the slowest method. ART1 gains performance speedup solely from interleaved sampling in visibility space. The values of the methods REG4 and REG8 demonstrate the upper bounds of performance gain through regular sampling in image space. The results of ART4 and ART8 are slightly faster than their regular counterparts REG4 and REG8, even though the ART methods compute adaptive upsampling to improve the visual quality. The impact of adaptive upsampling on performance can vary, since it is view dependent, leading to stronger variations in frame rate. However, ART4 shows a significant performance boost over RM.

6.1.3.3 Light Estimation Quality

Differences in the light estimation cause brightness and contrast differences in the final rendering. Therefore we examine the quality of the light estimation of each method relative to the reference method and propose two measurements. Note that we use synthetically generated light sources as ground truth. We shade the reconstructed geometry with this synthetic light and use the resulting renderings (Figure 6.4(a)) as simulated input for the photometric registration pipeline. The synthetic light source is created by directly projecting a low frequency function into SH coefficients. This low frequency function simulates an animated circular area light source, which falls off from center to border and has a distinguishable dominant light direction.

Environment map differences: As the environment map is a very common light source representation, we project the estimated light source from SH into a cubic environment map and evaluate the median of the pixel differences between the estimated and the reference environment light over all frames (Figure 6.6).

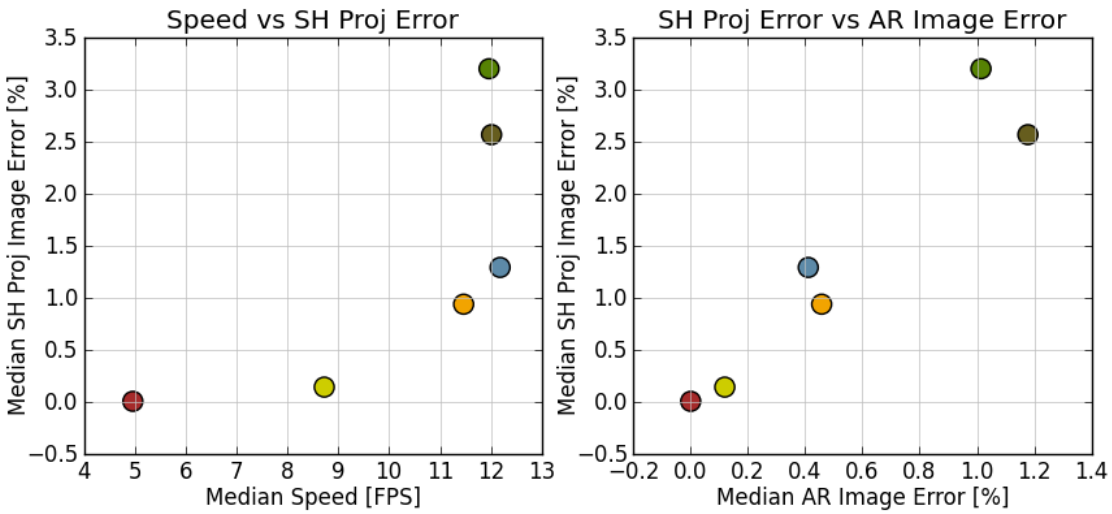


Figure 6.6: Left: the MSAD of the environment light maps. ART4 and ART8 score slightly worse then REG4 and REG8. Right: environment light maps MSAD vs. the AR rendering MSAD. A correlation between image quality and light estimation quality can be observed.

Dominant light source: The dominant light direction can have a strong impact on local shading models and is hence a valuable measure. We estimate the dominant light source direction from the first three SH coefficients [85]. Then we compare the estimated dominant light source with the synthetic one and compute deviations as the dot product of the respective direction vectors (Figure 6.4(b)). The average and standard deviation of the dot product between the reference method and the estimated light over all frames are reported in Table 6.1 for two sequences. We observe a higher error for sequences with moving cameras. However, the absolute error of the reconstructed dominant light estimation is relatively small.

	RM	REG4	REG8	ART1	ART4	ART8
Avg	0.9488	0.9399	0.9393	0.9574	0.9475	0.9521
Std	0.0348	0.0446	0.0449	0.0320	0.0412	0.0376
Avg	0.9616	0.9606	0.9626	0.9621	0.9559	0.9624
Std	0.0196	0.0261	0.0257	0.0186	0.0302	0.0256

Table 6.1: Average and standard deviation of the dot product between the estimated and the synthetic dominant light source directions. First row: static camera path. Second row: dynamically changing camera path.

6.1.3.4 Rendering Quality

Rendering quality assessment is more complex than performance assessment and requires multiple metrics [114] for a complete analysis: Therefore, we compare the final rendered result from each method to RM using both absolute differences and HDR-VDP2 [74].

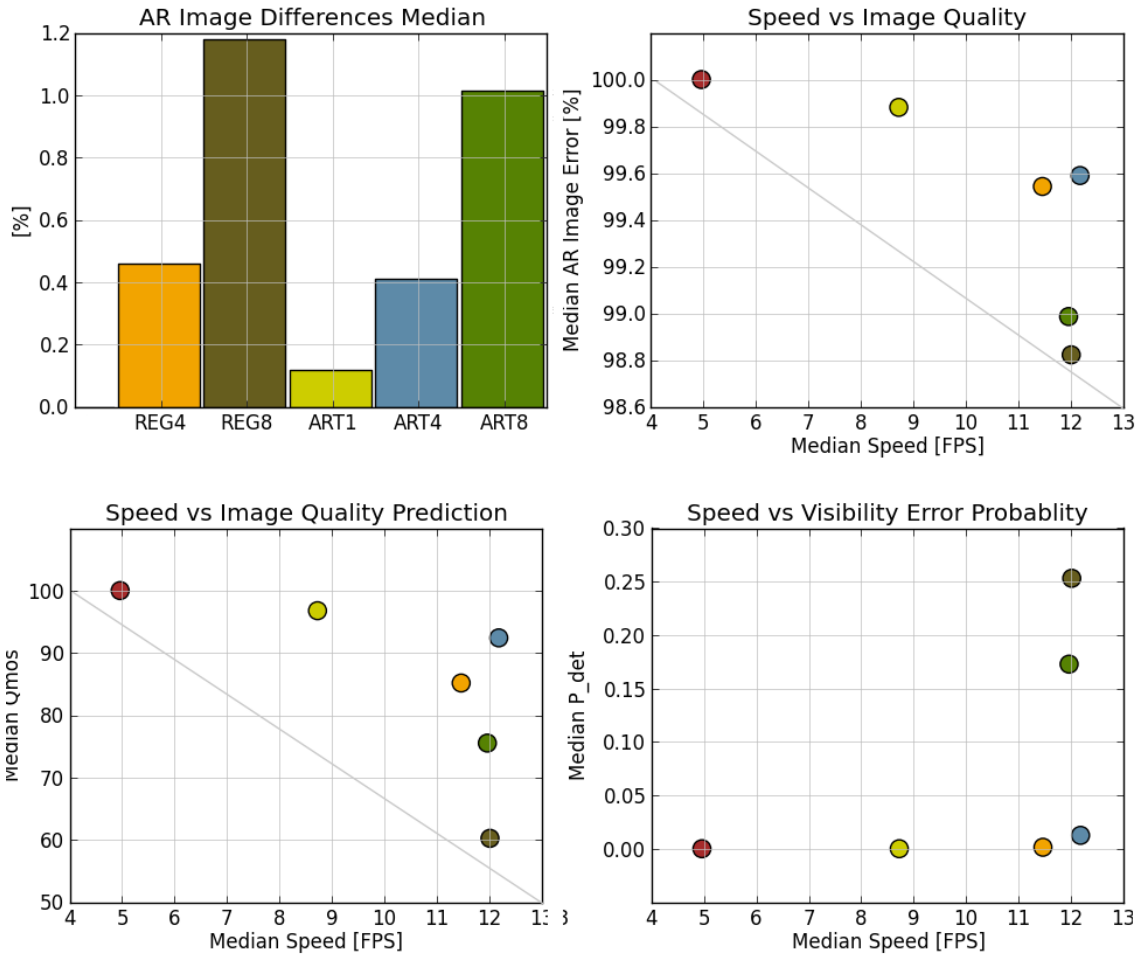


Figure 6.7: First and second column: median of absolute image differences (MAID) over all frames of the data set. The joint subsampling methods (ART1, ART4 and ART8) have smaller pixel differences to RM. In column 2, we express image difference as percentage of the image quality, where 100 percent is the best quality, for better comparison to the values in column 3 which show the results of the image quality prediction. Note that the x-axis shows the median FPS. The best method has to appear in the upper right corner of the plot. Column 4: Speed vs. probability that an image error is visible to the user. The results indicate that ART4 has the best trade-off between speed and image quality.

Absolute differences detect all deviations from the RM image, while perceptually driven measurements such as HDR-VDP2 mainly rely on saliency detection, which focuses on contrast and brightness changes. We inspect the rendering quality of the final AR image sequence frames along the following dimensions:

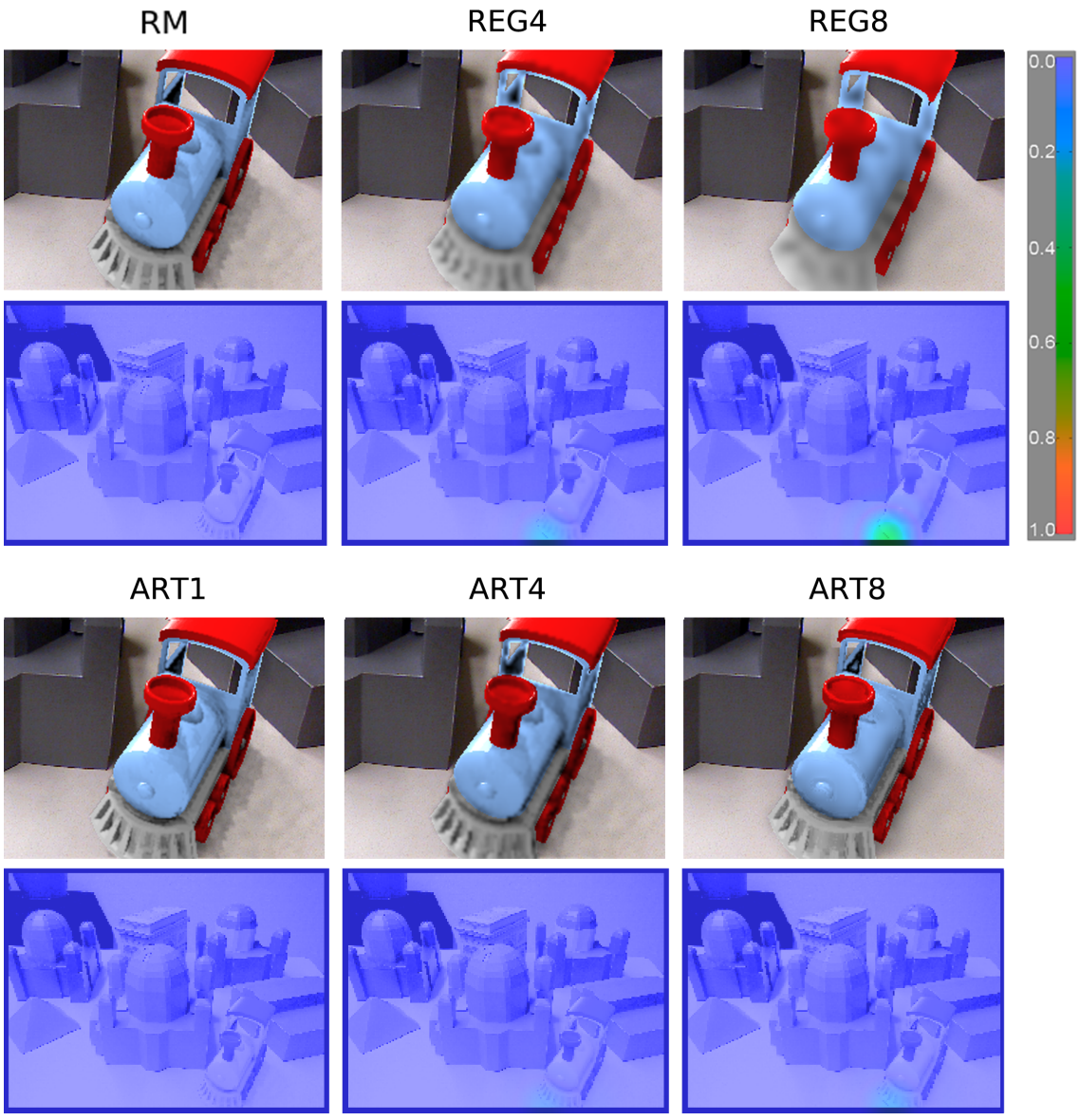


Figure 6.8: First row: close-up views of one AR test sequence. The methods solely sampling in image space (REG4 and REG8) loose details in the high frequency parts of the geometry due to naive upsampling. The joint subsampling methods incorporate adaptive upsampling, which preserves those critical areas. Second row: heat map visualization of the visibility error probability P_{det} (legend on the right of the figure).

Image differences: As a first assessment, we investigate pixel-wise image differences compared to RM. We compute the median of the sum of absolute pixel differences (MSAD) over all frames (cf. Figure 6.7, top left). Figure 6.7(top right) shows the normalized MSAD values as image quality (best quality is 100 percent corresponding to the RM output)

vs. the median performance over all frames. These results correspond to the renderings (Figure 6.8), where the ART methods subjectively show better results than the REG methods. Note that image differences were determined over the entire image, but only areas affected by virtual content or differential rendering can cause errors. The size of these areas varies and can be rather small, which makes the measurements very sensitive.

Image quality prediction: We compute the subjective mean opinion score Q_{mos} from HDR-VDP2, which predicts image quality as perceived by a human observer (cf. Figure 6.7, bottom left).

Visibility error probability: We also measured the maximum probability $P_{det} = max(P_{map})$ from HDR-VDP2 that a human observer would perceive any image difference between the reference image and the other methods (cf. Figure6.7, bottom right). In the even rows of Figure 6.8, P_{map} is visualized as a heat map for one frame over all methods. In summary, the results imply that ART4 has comparable image qualities to the reference method RM.

Although the presented sampling and caching methods enable significant performance improvements (see Section 6.2.3.2), the combination of the employed Kinect Fusion geometry reconstruction (cf. Section 4.2.1) and the visibility test based on ray-tracing does not provide enough performance to deal with fast changing geometry. In the following Section 6.2 we discuss an alternative method to volume ray-tracing based radiance transfer computation for photometric registration, which is based on the hybrid reconstruction algorithm discussed in Section 4.2.2 and operates in image space.

6.2 Radiance Transfer Approximation in Image Space

6.2.1 Screen-Space Directional Occlusion

In this section we discuss how to approximate the radiance transfer based on the diffuse reflection R_{DS} (see Equ. 5.2) in image space [37]. Instead of computing the visibility term V by expensive raytracing, we approximate it with a variant of screen-space directional occlusion (SSDO) [101], where visibility V' is computed by spherically sampling the space near x. We define the approximated diffuse reflection P_{DS} on a surface point x with normal $\mathbf{n}(x)$ and albedo $A(x)$ is an integral over the incoming radiance L from all possible directions w_j, weighted by incident angle and the visibility $V'(x, w_j)$. With discrete sampling in W directions, we write:

$$P_{DS}(x) = A(x) \sum_{j=1}^{W} V'(x, \mathbf{w_j}) L(\mathbf{w_j})(\mathbf{w_j} \cdot \mathbf{n}_x)\ d\mathbf{w_j} \tag{6.1}$$

A sample point y_j in object space is computed by adding a pseudo-random displacement to x along the specific direction w_j. The point y_j is then projected into the image-space aligned geometry buffer to look up the actual position s_j on the surface. If s_j is closer to the camera than x, we assume that light is blocked towards x along direction w_j. Unlike standard SSDO [101], we do not instantly compute the lighting for each ray

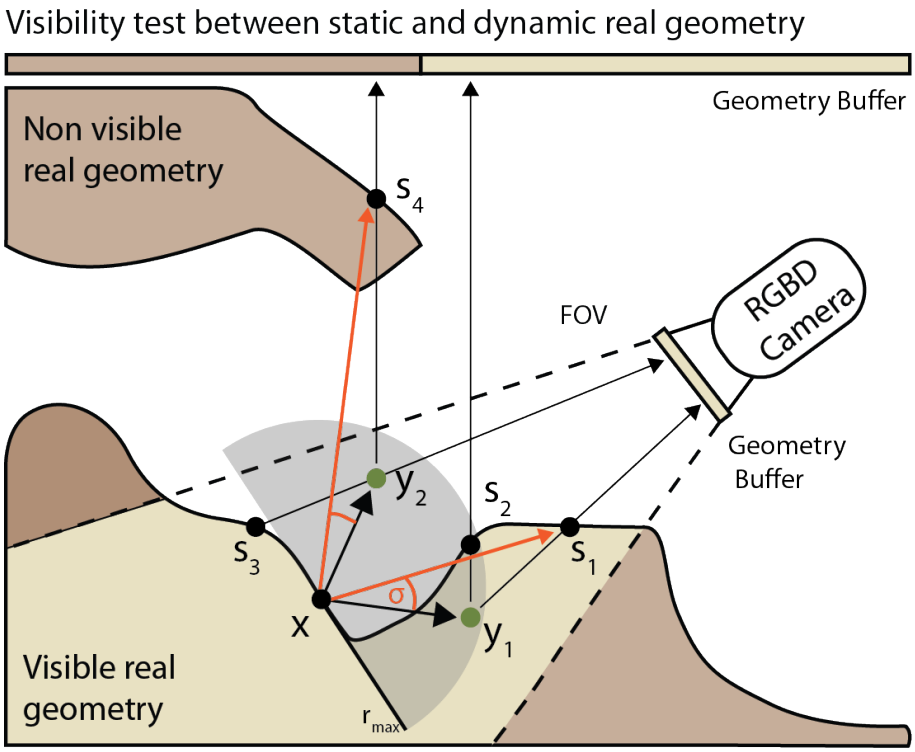
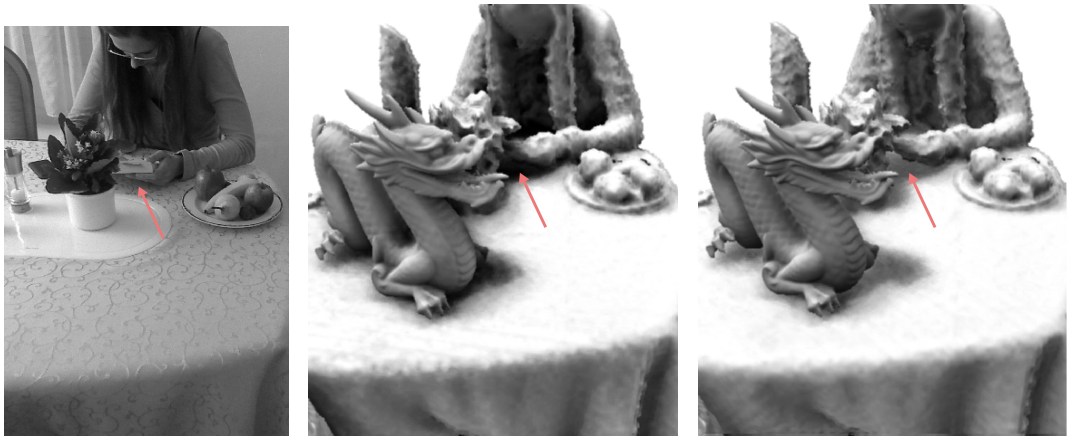


Figure 6.9: Visibility testing for point x. A sample point y_1 is tested by comparing its depth to the projection onto the surface, s_1. s_1 is closer to the camera image plane than x, so x is occluded along this direction. A second sample y_2 passes the depth test in the main camera buffer, but is occluded in the auxiliary geometry buffer covering the top view. The red arrows visualize the actual direction t_j from x to s_j and the angular distance to the visibility direction ω_j, which has to lie inside the threshold χ.

direction based on a known incident light source, but rather project the result as already explained in Section 5.1.2 into per-pixel SH for deferred light estimation and rendering. For projecting radiance transfer into spherical harmonics functions efficiently, we improve the SH weights computation for all W visibility directions, pre-computing random offsets for an area of 2×2 pixels rather than a single one. Such an approach is faithful to the SSDO idea of *coupled* ambient occlusion and directional lighting, and results in more realistic rendering results. Moreover compressing the resulting radiance transfer into SH reduces storage and computation requirements for the remaining pipeline, and also conveniently enforces low-pass filtering on the light estimation, which is necessary for robust extraction from unreliable scene reflections.

6.2.2 Self-Occlusion Correction

Approximating the visibility test in image space is error-prone. A naive visibility test delivers a binary result, which is 0 only if a sample does not pass the depth test. Wrong



(a) Input (b) No correction (c) Corrected visibility

Figure 6.10: (a) Intensity input image. (b) Without corrected visibility testing, occluded areas are too dark (red arrow) (c) Correcting for visibility testing results in more realistic shadows and allows light to reach corners.

visibility quantization can lead to over-estimation of occlusion, making shadows too dark (cf. Figure 6.10(b)). We account for visibility quantization errors by modeling V' as a continuous value, which rises from 0 to 1 with the angle between $\mathbf{w}_j$ and the normalized vector $\mathbf{t}_j$ from x to s_j. In Figure 6.10(c), we show the results of the self-occlusion correction.

$$\mathbf{t}_j = (s_j - x)/|s_j - x| \tag{6.2}$$

$$V'(x, \mathbf{w}_j) = min(\chi, 1 - (\mathbf{t}_j \cdot \mathbf{w}_j))/\chi \tag{6.3}$$

The threshold χ in Equation 6.3 is related to the angle between adjacent sampling directions. We account for the jittered random sampling by using the median of all angles σ_{ij} between neighboring samples $\mathbf{w}_i$ and $\mathbf{w}_j$, as determined from a spherical Delauney triangulation Ψ of the sampling. To account for the Nyquist limit, we determine χ from half of this angle:

$$\chi = \cos(\text{median}(\alpha_{ij} \in \Psi)/2) \tag{6.4}$$

In the following Section 6.2.3, we evaluate the light estimation quality of the image based radiance transfer approximation algorithm proposed in this section against the results of the light estimation produced by the algorithm based on volume ray-tracing discussed in Section 5.1 and the accelerated version discussed in Section 6.1.

6.2.3 Evaluation

6.2.3.1 Light estimation

In this section we evaluate the light estimation quality comparing the dominant light directions of each method. The dominant light direction is extracted from the environment light estimation given in SH and has a great impact on the final rendering. We compared two configurations the image-space approach (GR15: full sampling and GR15Fast: quarter sub-sampling in image space) against a standard Augmented Reality light estimation method (LP) based on a diffuse spherical light probe, employed in previous work [2, 19]. Additionally we compared GR15 against the approach of Gruber et al. [36] (GR12) for the light estimation, which is equivalent to the reference method (RM) in the previous evaluation Section 6.1.3.

Since LP, the method based on passive light probes, naturally can suffer from light probe tracking, we additionally captured the entire testing setup with an omnidirectional HDR camera (PointGrey Ladybug3). We placed the omnidirectional camera at the same position of the passive light probe and manually registered it to the common world coordinate system. We obtained environment images of our testing setup directly sensing the light sources (see top row of Figure 6.11). Our test data set consists of six real-world scenes with increasing complexity of geometry and textural elements (Figure 6.11, bottom row). The test data set has been captured in the testing environment described in Section 3.3.2, were we installed four area light sources arranged in a half circle around the center. We recorded RGB-D sequences with a Kinect for each scene and light source with repeating camera movements, where each sequence has 1000 frames. In total, we recorded and evaluated 24000 frames.

In Figure 6.11, we show two types of results. The first results are the median value of all measured dominant light directions of all three evaluated methods and LP (yellow). We visualized the median points on the panoramic images from the omnidirectional cameras. As can be seen, the LP is more accurate, but overall, all methods estimate the dominant light direction quite successfully. Note that the results of GR15 and GR15Fast are very similar and therefor overlap. For the case of Light 4, LP clearly provides better results. This is due to the fact that the geometry of the LP method provides perfect surface normals as input, compared to the methods based on geometry reconstruction. If the camera does not sense enough information from the scene geometry, it is possible that certain surface normals are not sampled, which leads to biased light estimations.

The box plots in Figure 6.11 show the angular distance of GR12, GR15, and GR15Fast to LP. We visualized the mean (red bar) for each scene over all light sources In scene 1 we have a sphere as input geometry (same geometry as the light probe from LP) and directly compare results between perfect input data (LP) and input data from the geometry reconstruction. In scenes where the amount of captured geometry and the geometrical variety is small with respect to the sampled environment, but contains different surface colors, such as Scene 3 in Figure 6.11, the light estimation algorithms deviate more. The best results are obtained in Scenes 5 and 6, where the surface geometry provides sufficient input for the light estimation. Overall, our image based methods subjectively have the same quality of light estimation results as the volume raytracing based method (GR12).

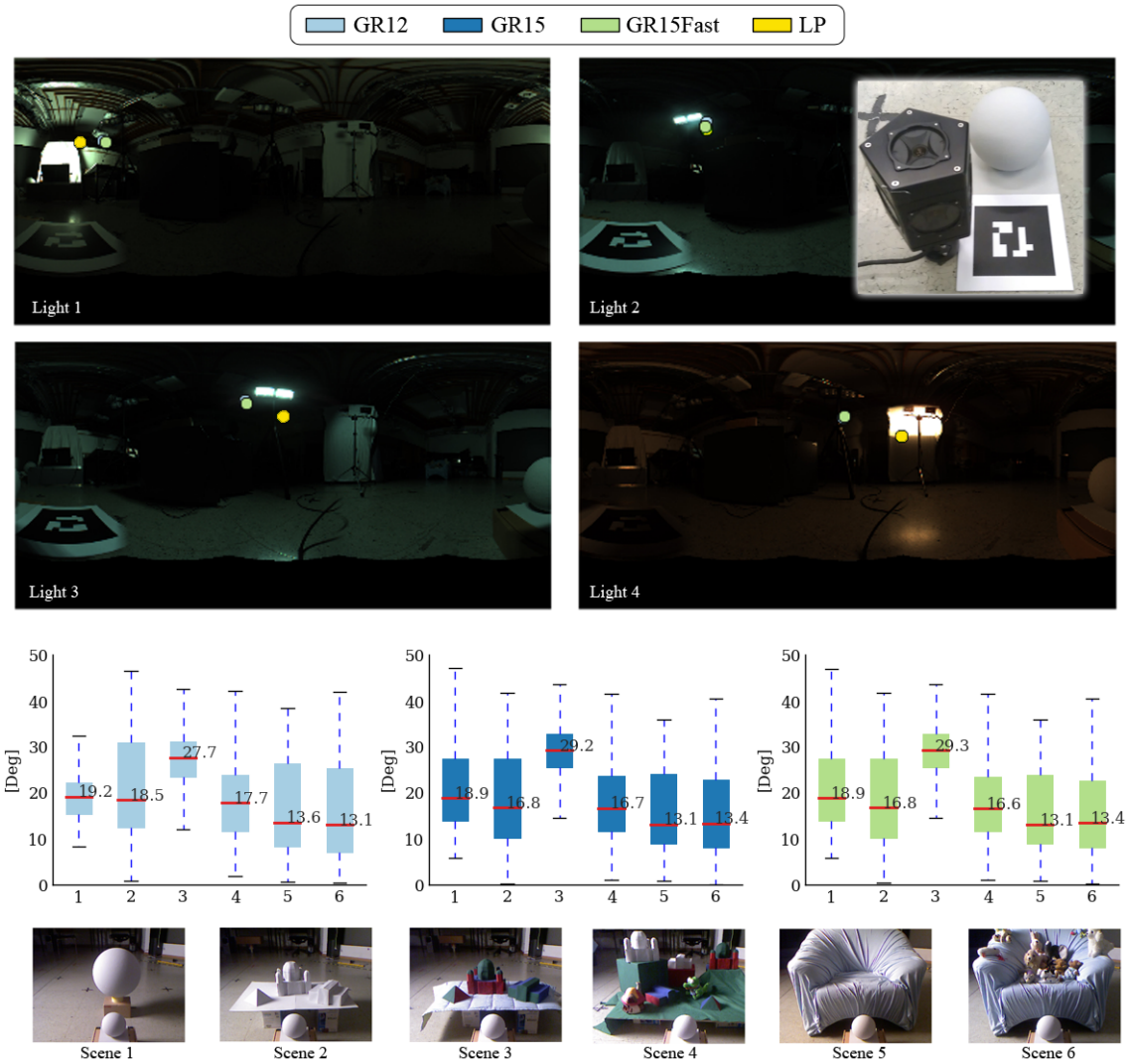


Figure 6.11: The top row images show environment maps of the testing setup with each light source. The environment maps are created with the omnidirectional Ladybug3 camera shown in the inset at the right. Each colored dot visualizes the median of the angular distance of the dominant light direction of each method to the LP method (yellow), drawn on top of the environment maps. For the LP, method we used a diffuse gray painted sphere, which meets best our diffuse gray world assumption. The registration of the light probe was solved by using an ARToolkit marker, which also defines the origin of our test setup. The middle row shows the median of the angular distance of the dominant light directions of each method to LP. Each column in the plots correspond to the scenes in the bottom row.

	GR12	GR15	GR14	GR15Fast
FPS	5.8	13.98	12.29	22.46
Update	172.45	71.5	81.36	44.53
Input	7.3	7.06	7.44	7.02
Reconstruct	11.1	10.9	10.24	10.64
Surface	6.69	12.57	6.69	12.6
DynGeo	0	1.19	0	1.02
RadTransfer	130.48	36.22	50.54	10.5
LightEst	3.86	2.67	5.27	1.8
Rendering	0.92	0.86	0.98	0.87

Table 6.2: Performance evaluation of our light estimation and rendering system against previous work based on volume ray-tracing. The results of GR15 and GR15Fast show strong performance improvements for the radiance transfer computation over GR12 and GR14.

6.2.3.2 Performance

We measured the performance of time-critical passes from the entire system. The major passes are labeled and described as follows: *Input*: The input data processing consists of capturing RGB an depth data and applying a noise filter to the RGB image. *Reconstruct*: With Kinect Fusion [80], we compute the geometry reconstruction and estimate the 6DOF camera pose. *Surface*: The implicit surface from the volume modeled as a truncated signed distance function is extracted by ray-casting the volume, which also includes the surface extraction for the occlusion buffers. *DynGeo*: The hybrid geometry processing algorithm explained in Section 4.2. *RadTransfer*: Radiance transfer computation (RT) differs on the principal method. These are based on volume ray-tracing for (GR12 and GR14) or image space approximation (GR15 and GR15Fast). *LightEst*: Light estimation depends on the number of samples consisting of the per pixel radiance transfer in SH and the intensity image from the camera. *Rendering*: The rendering pass includes rasterization of the virtual objects, differential rendering and Augmented Reality compositing.

In a first test (Table 6.2), we compared GR12, which is based on volume ray-tracing, against GR15, which is based on image-based occlusion techniques, on a desktop computer (PC) with an NVIDIA GeForce 780. Both methods regularly sample the entire image space. Besides GR12, we compared GR14 [35], which is based on a 4×4 regular sub-sampling in image space and adaptive edge refinement, against the fast variant of GR15, which also uses a 4×4 regular sub-sampling in image space (GR15Fast). The size of the working volume is $2m^3$ with a 256^3 voxels.

In a second test (Table 6.3), we compared the performance of the PC to a notebook (NB) with an NVIDIA Quadro K2100M and a tablet (M) with an NVIDIA GT640M LE. To meet the hardware limitations of NB and M, we reduced the reconstruction to 64^3 voxels and omitted computing D'''(see Section 4.2.2). The methods based on volume ray-tracing (GR12 and GR14) did not achieve frame rates above 1Hz and have been left out in this evaluation. All timings, except from frames per seconds (FPS) are in milliseconds.

	GR15			GR15Fast		
	PC	NB	M	PC	NB	M
FPS	18.24	3.23	2.19	33.60	7.34	5.0
Update	54.8	309.86	456.78	29.76	136.24	200.48
Input	0.8	0.86	1.29	0.79	0.81	0.9
Reconstruct	9.42	37.68	66.0	9.42	36.86	66.16
Surface	5.28	30.73	45.68	5.28	30.02	44.99
DynGeo	1.27	18.25	25.77	1.12	17.89	25.75
RadTransfer	34.57	214.27	314.68	9.96	38.69	59.17
Light Est.	2.56	4.56	1.98	1.93	8.52	2.24
Rendering	0.89	3.52	1.36	1.24	3.43	1.26

Table 6.3: In this table, we show the performance measurements on a mobile device (M) compared to the values measured from a Notebook (NB) and a PC. We achieve interactive frame rates with the performance optimized variant (GR15Fast) of our algorithm. A major bottleneck for the mobile device is volume processing including reconstruction and surface evaluation. However, this could be, for example, substituted by a state of the art mobile visual SLAM system [24, 103] with the loss of fast dynamically changing geometry. The lower resolution of the volume does not affect camera tracking or light estimation, but can create visual artifacts at shadows.

6.3 Combination of Acceleration Techniques

In Section 6.1, we discussed three different acceleration techniques: adaptive image space subsampling, interleaved visibility subsampling, and robust visibility caching in world space. These acceleration techniques are not necessarily bound to ray-tracing only and have potential to be applied in combination with our approximated radiance transfer technique discussed in Section 6.2. Therefore, we implemented a combination of regular subsampling in image space with the approximated variant of radiance transfer computation (GR15Fast). We computed the radiance transfer only for every 4^{th} pixel in both dimensions. For the final shading pass, we up-sampled the result to the resolution of the final Augmented Reality output image using bicubic interpolation. This naturally introduced artifacts, which could be improved by additionally applying the adaptive refinement pass, where radiance transfer is computed every pixel on critical visible parts of the geometry (e.g., edges). This would come with the expense of additional radiance transfer computation on critical surface areas of the geometry as described in Section 6.1.1. However, the performance impact would mainly depend on the complexity of the virtual model. A further alternative approach would be the implementation of a more intelligent up-sampling algorithm compared to bicubic up-sampling, which operates on the intensity image and the geometry as well, such as enhanced morphological antialiasing [48].

Interleaved sampling in visibility space is the second acceleration technique to consider. In Section 6.1.3.3, we distribute the set of different visibility directions over a couple of frames. This involves caching the visibility information in 2D buffers and reconstructing the entire visibility signal for each frame by re-projecting over the geometry. This technique works reliably well for slow changing geometry, as evaluated in Section 6.1.3.4. Due to the performance gain of the approximated radiance transfer in combination with the hybrid

surface geometry reconstruction algorithm discussed in Section 4.2.2, we are able to process faster changing geometry. This makes re-projection over geometry a more challenging task and would require computing the entire flow of each individual pixels to provide robust re-projection. To avoid the costs of pixelflow computation, we did not consider any acceleration technique which involved re-projection.

Visibility caching in object space, as described in Section 6.1.2, is complementary to the other techniques. In this technique, we store information about the amount of occlusion of a certain surface point. Depending on this information, we decide to compute the visibility test or not. This improves the performance for rather static scenes, which have little amount of self-occlusion. However, for more dynamic scenes, caching would not create a strong performance gain, since we would have to recompute the entire radiance transfer including visibility testing. Moreover this techniques comes with the extra expense of reading and updating the cache entries.

In conclusion, we showed by implementing GR15Fast the maximum performance improvement the system can achieve without enabling radiance transfer caching. However, for optimal radiance transfer caching in combination with the hybrid geometry processing algorithm discussed in Section 4.2.2, a more intelligent data structure, different from the regular 3D voxel volume, would have to be considered.

6.4 Summary

In this chapter we presented different acceleration methods to reduce the computational costs for radiance transfer computation, leveraging the major performance bottleneck we identified in Chapter 2.2. At first we discussed acceleration methods which were suitable for volumetric ray-tracing based radiance transfer computation (see Section 6.1). The evaluation of these acceleration methods in Section 6.1.3, based on subsampling and visibility caching, showed that the methods improve the performance keeping the light estimation and visual quality at acceptable levels. However, computing the visibility test using ray-tracing and hence marching into the 3D voxel volume of the reconstruction is a very expensive operation. Moving away from ray-tracing, we further improved the performance of the radiance transfer computation by approximating the visibility test in image space (see Section 6.2). This has also the advantage that it is agnostic from the geometry reconstruction algorithm.

However, we evaluated the image space approximation method in combination with the hybrid dynamic geometry reconstruction algorithm presented in Section 4.2.2 against the volume ray-tracing based methods and state of the art reference methods based on spherical light probes. We achieved remarkable performance gains enabling us to run the photometric registration and rendering pipeline on mobile hardware.

In the following Chapter 7, we will discuss the Augmented Reality rendering techniques, which benefit from the performance gain, enabling global illumination effects in dynamic environments.

7 Global Illumination in Dynamic Environments for AR

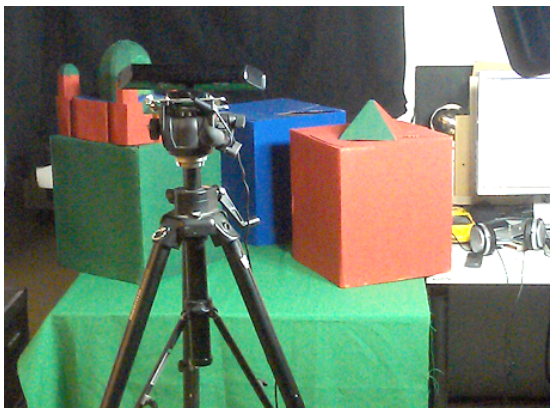
Contents

In the previous chapters about our photometric registration approach and several acceleration methods we proposed a real-time capable photometric registration pipeline from unknown and unprepared scenes, which is a key technology for visually coherent rendering in Augmented Reality. Having the real-world lighting estimated we are able to compute global illumination rendering effects between both real and virtual such as shadows and indirect illumination.

In this chapter we, discuss the evolution and development of the Augmented Reality rendering stage briefly introduced in Section 3.1. In Section 7.1, we explain how we use the estimated lighting for differential rendering. Then we describe the first steps towards interactive global illumination in dynamic environments for Augmented Reality in Section 7.2. This approach builds on volume ray-tracing and can already provide convincing results in combination with the acceleration methods described in Section 6.1. However, the ability to render lighting effects and shadows from mid-sized environments with fast changing geometry is not feasible with volume ray-tracing because of performance impacts. We provide a solution for global illumination supporting fast changing geometry in the successive Section 7.3, where we present an image based global illumination approach, which considers both the geometry inside and outside the FOV, so that radiance can be transferred across the FOV boundary. In Section 7.4, we confront these two principal rendering approaches.

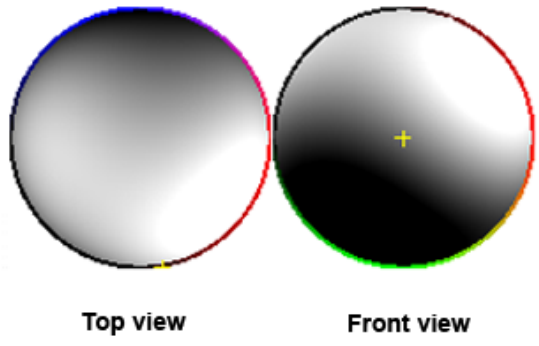
7.1 Differential Rendering

The goal of the rendering stage is to apply the estimated lighting on the scene. Since we are interested in rendering virtual objects into the real scene, we have to compute the shading on the virtual objects, but also in the areas where light interaction between virtual and real-world geometry happens, such as shadows from a virtual object onto a real object. The remaining areas in the image should not be affected by the rendering and directly show the real-world through the camera image. To make the light interaction between real-world models and virtual models visible, we use differential rendering as



(a) Workspace

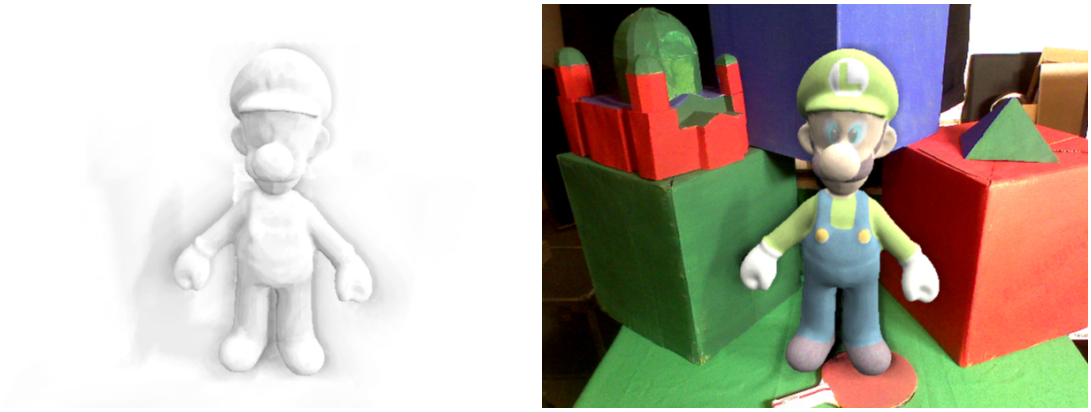
(b) Occlusion



(c) Light estimate

(d) Shading buffer R'_b

Figure 7.1: Different stages of the light estimation. (a) Overview of the setup with work space and camera. (b) Visualization of the computed occlusion for the geometry. (c) Top and front view of the light estimation projected onto a sphere. (d) Diffuse shading of real and virtual geometry.



(a) Differential buffer (b) Final rendering

Figure 7.2: (a) Differential rendering buffer I_{DS} and (b) final rendering I_{DC}.

proposed by Debevec et al. [19]. The principal idea of differential rendering is to compute the lighting of the real-world and the lighting of the real and virtual world separately. The difference is then applied to the camera image. Differential rendering requires computing the diffuse shadowed radiance transfer two times, once for the real-world, R, and once for the combination of the real-world and the virtual world R', both represented in SH and stored in camera image aligned buffers. The radiance transfer computation is based on the Lambert illumination model supporting shadows, which is described in Equation 5.2.

We exploit the assumption that our estimated real-world light is white, computing two monochromatic differential rendering buffers, R_b for real and R'_b for real+virtual. In Section 5.1.2, we discussed how to estimate $\widetilde{F}$ which is the real-world environment lighting represented in SH. Since the radiance transfer is already represented in SH form, we can directly evaluate the shading R_b and R'_b by computing the dot product of $\widetilde{F}$ and R or R'. In Figure 7.1, we illustrate the outputs of the different steps, including an overview of the workspace and a visualization of the visibility test as also the shading result of R'_b. Note that we projected $\widetilde{F}$ onto a sphere for better visualization in Figure 7.1(c). The directional differential rendering result I_{DS} is then computed as follows:

$$I_{DS} = (1 + H(R'_b) - H(R_b)) \tag{7.1}$$

$$I_{DC} = I_{RGB} I_{DS} \tag{7.2}$$

In Equation 7.2, we add the difference between the two differential rendering buffers to the color camera input image I_{RGB}. To support all frequencies, we have to multiply the buffers R_b and R'_b with the color information of I_{RGB} first. Note that we apply a tone mapper H from [96] to R_b and R'_b based on the average intensity of each buffer. This is necessary because our differential rendering buffers are computed in high dynamic range, while the input data I_{RGB} originates from a low dynamic range camera. See Figure 7.2(a)

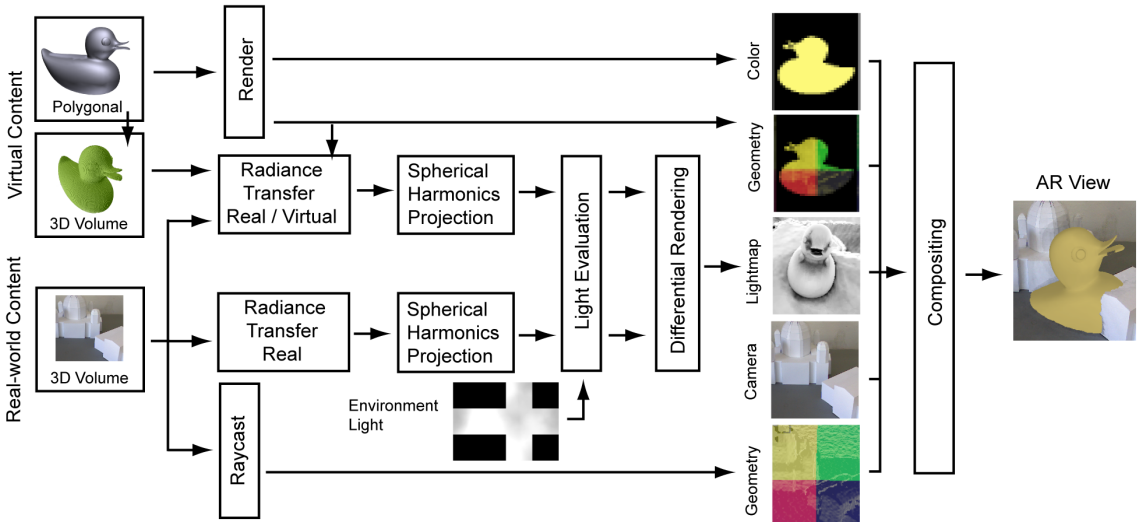


Figure 7.3: The photometric registration and rendering pipeline is a combination of differential rendering with a global illumination representation based on SH.

for the result of an differential rendering buffer I_{DS} and the final blending in Figure 7.2(b).

In this section we discussed the principle method to apply the estimated lighting in the rendering pipeline, using the two radiance transfer buffers R_b and R'_b. In the following sections we will discuss two principal methods for computing the radiance buffers and the differential rendering buffer. The focus of the discussion lies in the visual impact of each method on the final rendering. Moreover these methods are naturally tightly connected to the geometry processing (see Chapter 4), the photometric registration stage (see Chapter 5), and the acceleration methods presented in Chapter 6.

7.2 Interactive Volume Ray-Tracing Based AR Rendering

7.2.1 Pipeline

Similar to the real-world geometry, we also create an implicit surface representation of the virtual geometry stored in a voxel volume. This is done with an off-line 3D mesh voxelizer as described in Section 3.1. Choosing a unified data structure for the real-world geometry and the virtual geometry enables a straight forward evaluation of the visibility term of the radiance transfer function by multi volume ray-tracing. Alternatively, the geometry of the virtual volume could be also processed as triangle mesh on the GPU using standard GPU ray-tracing methods. In that case, the algorithm for computing the radiance transfer between the real and the virtual geometry would have to deal with different memory access patterns (regular volume grid versus hierarchical triangle mesh structure) during one visibility test, lowering cache performance. In Figure 7.3, a more detailed systematic overview of the entire photometric registration and rendering pipeline presented in Gruber et al. [36] is shown. In particular, it shows the data flow from the real-world geometry and the virtual geometry to the light estimation and final rendering.

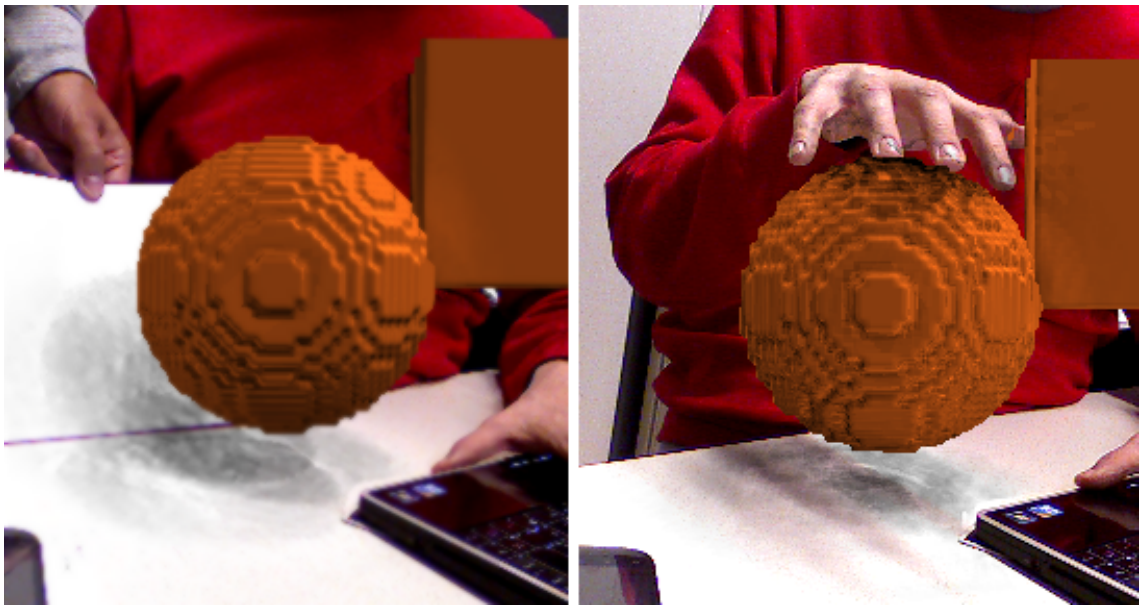


Figure 7.4: The left image shows shadows casting from the virtual object on the table and the paper. The right image shows a shadow from the real-world geometry (hand) onto the virtual geometry.

In this section, we especially focus on the three main passes necessary for computing the differential rendering discussed in Section 7.1.

First pass: In the first pass, an off-screen geometry buffer rendering of the virtual scene is created using standard OpenGL, using the camera pose that was determined through tracking with KinectFusion. This render pass produces a color buffer of the unlit virtual scene together with a geometry buffer G_v (i.e., x/y/z coordinates of every fragment in camera coordinates) and a normal buffer. The color buffer is later used in the compositing step to compute the final shaded surface colors, while G_v is used to initialize the volume ray-tracing employed in the second rendering pass.

Second pass: In the second pass, the radiance transfer is computed. The first solution for the real-world only is R, which is used for the light estimation and the rendering. The second solution is R', which is the radiance transfer for real and virtual objects together. We project both R and R' to SH coefficients. Finally. the lighting is evaluated by computing the dot product of the SH coefficients of the environment lighting with the SH projected radiance transfer, and inserting the result into the differential rendering equation (cf. Equ. 7.2), which yields I_{DS}.

Third pass: The compositing pass is necessary to create the final Augmented Reality image. The main purpose is to handle occlusions of virtual and real objects correctly and to compute the final rendering. For handling occlusions, the depths values in the geometry buffer of the real-world surface and the geometry buffer of the virtual surface

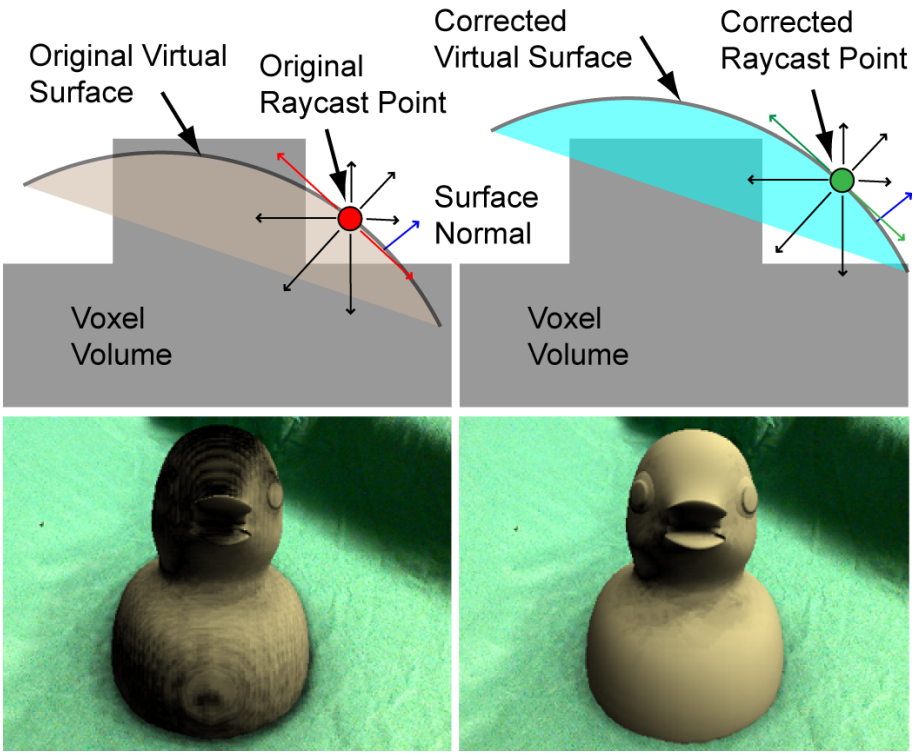


Figure 7.5: In the upper left image the problem of self occlusion artifacts is depicted. The lower left image shows the virtual object (duck) without occlusion offset and the lower right image shows the duck with occlusion offset. As can be seen, the surface of the duck in the lower right image has less stains.

are compared. Depending on the depth test, the color of the camera frame or the color of the virtual object is taken and multiplied by I_{DC}. Note that the real-world surface might contain discontinuities or holes, where KinectFusion was unable to provide a proper reconstruction. This can for example happen when the depth sensor is confused by highly specular surfaces. In these cases, a proper workaround is to assume that the virtual surface is closer than any real surface, and consequently use the shading information from the virtual shading. To work in linear color space, we removed the gamma correction in the video color frame at the beginning of the pipeline. As a post-processing step, we finally apply the gamma correction to the output color.

See Figure 7.4 for an example of the effects achieved with differential rendering. Shadows are casted from the virtual to the real and vice versa. For better understanding we visualized the voxel volume of the virtual object.

7.2.2 Visual Artifacts

In volume ray-tracing, we shoot rays through the volume to search for the implicit surface boundaries and compute the visibility test. This requires a regular sampling of the 3D volume and can naturally create aliasing artifacts, noticeable as wrong self-occlusions.

Self-occlusion is typically caused when the resolution of the voxel grid does not match the resolution of the virtual geometry or when the reconstructed real surface is noisy (see Figure 7.5(a)). As one improvement, we take advantage of the geometry buffer G_v produced in the first pass and start the ray-tracing pass from this surface instead from the implicit surface estimated by the TSDF. We also use the normal buffer from G_v, as it has higher accuracy than normals that could be estimated from the TSDF. Moreover we introduce an offset for the starting point for each ray along the surface normal by the length of one voxel cell.

7.2.3 Reasoning

In Section 6.1, we discussed the limitations of volume ray-tracing based radiance transfer computation. A major limitation is the expensive visibility test which requires sending secondary rays through the volume. This has severe impacts on the performance, depending on the length of the ray and the number of rays. Therefore, we developed acceleration methods based on visibility caching and sub-sampling in image space. We discussed these methods in the context of overall performance, light estimation quality, and also visual quality. The visual quality was mainly affected through artifacts caused by image space subsampling and upsampling as illustrated in Figure 7.6. We provided extensive evaluation on this in Section 6.1.3.

In this section we continue the discussion on limitations of volume based radiance transfer in the context of real-time global illumination. To obtain interactive or real-time performance, the ray length and the number of rays has been limited. This becomes apparent when the occluding geometry is further away from the surface then the ray length. Therefore the approachs from Gruber et al. [35, 36] handle only near field illumination and shadowing. An alternative solution is to add conventional shadow mapping. In this solution we combine near field global illumination based on volume ray-tracing with shadow mapping supporting one dominant light direction. The dominant light direction is computed from the SH coefficients from the real-world light estimation $\widetilde{F}$. The shadow maps are computed by evaluating the implicit surfaces from the real and the real and virtual voxel volume. Although this creates shadows from more distant occluders it requires additional surface extraction and naturally impacts the performance. In Figure 7.7 we show two different results where near field or contact shadows around corners are combined with the distant shadows from the shadow map.

In Section 6.2, we demonstrated that by approximating the radiance transfer in image space we can achieve comparable results to volume ray-tracing for the photometric registration and achieve significant performance improvements at the same time. In the next Section 7.3, we will discuss the benefits of the image space approximation from the rendering point of view and show consistent shadowing for the entire workspace and indirect illumination effects.

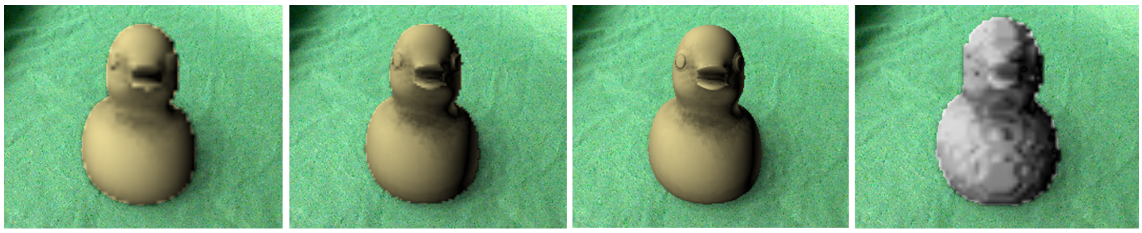


Figure 7.6: In this figures we show the typical aliasing artifacts created by sub-sampling in image spaceFrom left to right: Sample every 4th pixel, every 2nd pixel, every pixel, voxel representation.

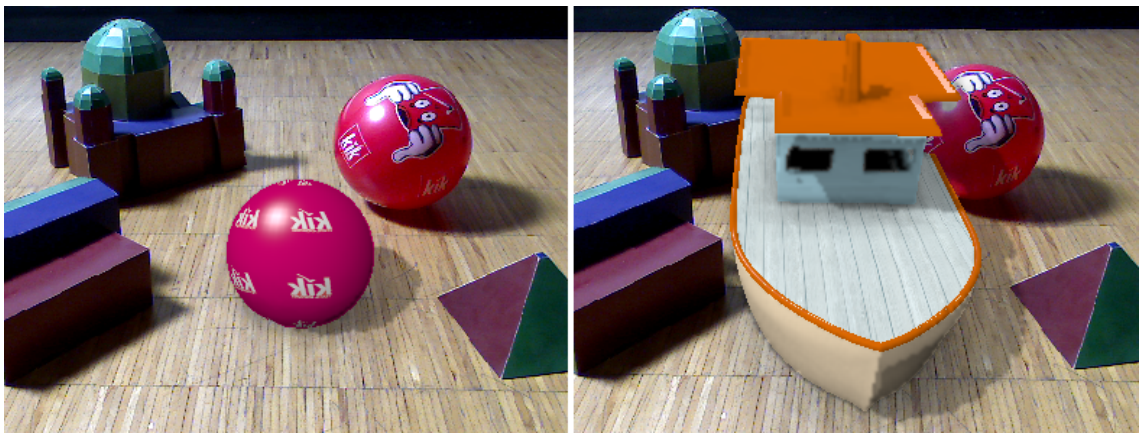
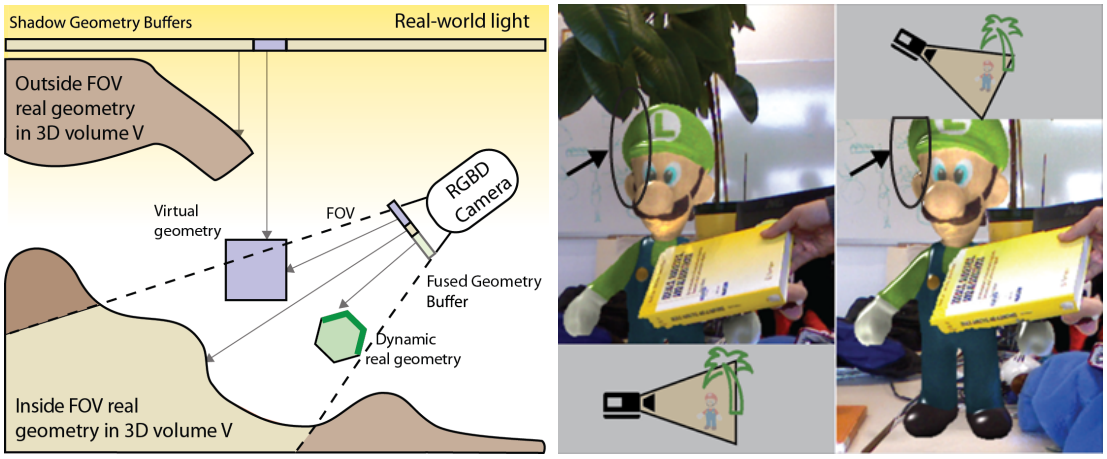


Figure 7.7: SH shadowing mixed with shadow mapping.

7.3 Real-Time Image Based Global Illumination for AR

7.3.1 Pipeline

In Section 6.2 we introduced photometric registration based on image-space occlusion detection which is fast and independent of global scene complexity. As expected the image based method brought significant performance gain which allows us to add more global illumination effects. More importantly, the radiance transfer approximation provides also good results for the photometric registration. The approximation error introduced by computing occlusion in image-space rather than in object-space is negligible compared to geometric reconstruction errors. However, the rendering pipeline discussed here is different in certain aspects to the rendering pipeline for volume ray-tracing based rendering (see Section 7.2.1). The main difference is that virtual geometry does not have to be pre-processed into a voxel based representation anymore. The rendering pipeline of the virtual geometry is a deferred shading pipeline, where the geometry information is stored in intermediate geometry buffers containing geometry information and the albedo. To meet mobile hardware requirements, we implemented a variation of our approach (GR15Fast) accelerating the radiance transfer computation by regularly sub-sampling in image space. This creates a quarter resolution shading buffer R_b and R'_b which is then up-sampled



(a) Workspace (b) Consistent Shadows

Figure 7.8: Left: Work space with geometry inside (light brown) and outside (dark brown) the camera FOV. Static real geometry (brown), dynamic real geometry (green) and virtual geometry (purple) are all registered in the fused geometry buffer G. Additional shadow geometry buffers are placed around the main camera frustum to compute occlusion or visibility. Right: Example for consistent shadowing. The shadows casted by the plant on the virtual figure remains after moving the plant out of the FOV.

using a bicubic interpolation with standard Gauss filtering with a kernel radius of eight. In Section 7.4, we juxtapose full sampling (GR15) and the fast approach with 4×4 subsampling. In the following Section 7.3.2, we discuss how we achieve consistent shadowing for global illumination in mid-sized work spaces.

7.3.2 Consistent Shadowing

We understand consistent shadowing as computing the shadows from all visible geometry in the FOV, but also from static geometry outside the FOV (see Figure 7.8(a)). For Augmented Reality, this means that shadows are potentially cast from objects which are not in the FOV or moved out of the FOV for a couple of frames. An example is illustrated in Figure 7.8(b), where the camera moves down in the second frame, and the plant which casts a shadow on the virtual object moves out of the screen. Considering the reconstructed static geometry of the plant we are able to compute the shadow also if the plant is not visible. Consistent shadowing becomes necessary, if desktop sized scenarios, which normally fit into the FOV, are exceeded. In the following we describe our algorithm.

To compute the radiance transfer *from both* geometry inside and outside the FOV, but *only to* geometry inside the FOV we create additional shadow geometry buffers extracted from the real-world volume geometry in V with orthographic projection. We approximate a partial cubic map with the main camera aligned geometry buffer G and three additional

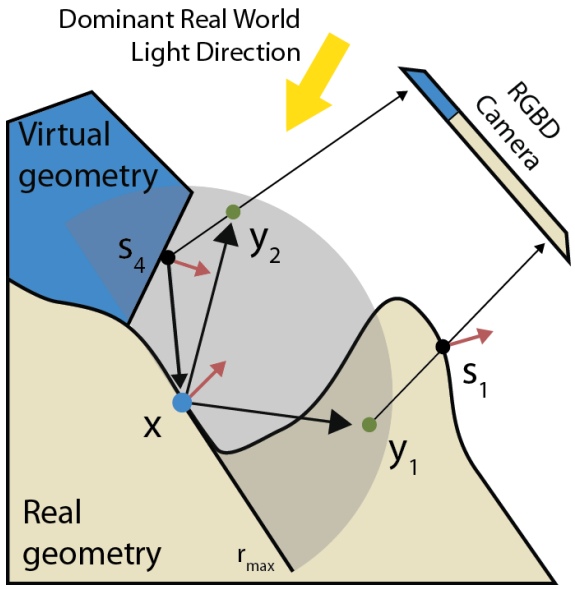


Figure 7.9: Indirect lighting is computed, if occlusion is detected. For example, s_4 reflects light towards x, but s_1 faces away from x and hence does not reflect light.

shadow geometry buffers for the left, right and top parts of the scene. Additionally, we add a shadow geometry buffer projected from the dominant light direction, which is extracted from the SH coefficients of the real-world light estimation. All auxiliary shadow geometry buffers hold vertex positions in the same coordinate system, which makes testing sampling points y_j in each buffer straight forward. The shadow buffers are then directly used for additional visibility testing in the image based radiance transfer computation discussed in Section 6.2. This variant of visibility approximation may miss thin objects in the scene and is only truly correct for the rays which are facing towards the camera. However, our evaluation in Section 6.2.3 shows that this approximation suffices for light estimation. In Section 7.4, we will show that this holds also for the visual quality of the rendering.

7.3.3 Indirect Illumination

Our indirect illumination algorithm is based on the SSDO [101] algorithm introduced in Section 6.2. Indirect diffuse illumination R_{ID} from one light bounce can easily be incorporated into SSDO as illustrated in Figure 7.9. When a sample y_j near x is found invisible, the light transport from the corresponding surface point s_j to x is approximated. R_{ID} is computed as a diffuse reflection from a single dominant light L_D with the direction $\mathbf{w}_D$ without further visibility testing.

$$R_{ID}(x) = \sum_{j}^{W}(1 - V'(x, \mathbf{w}_j))L_D(\mathbf{w}_D)max(\mathbf{n}(s_j) \cdot \mathbf{w}_D, 0)\Gamma(x, j)O(x, j) \tag{7.3}$$

The indirect illumination is weighted by a "form factor" term $\Gamma(x, j)$, which computes

Lambertian weights for the angles between the normalized transmittance direction $-\mathbf{t}_j$ and the normals $\mathbf{n}(x)$ and $\mathbf{n}(s_j)$, respectively, and corrects for the distance from x to s_j. The term $O(x, j)$ suppresses light transfer from points facing away from x:

$$\Gamma(x, j) = \frac{(\mathbf{n}(x) \cdot t_j)(\mathbf{n}(s_j) \cdot (-\mathbf{t}_j))}{(x - s_j)^2 \pi} \tag{7.4}$$

$$O(x, j) = \max(sgn(\mathbf{n}(x) \cdot (-\mathbf{t}_j)), 0) \tag{7.5}$$

For adding indirect illumination to differential rendering, we compute analogue to the previous pass (see Section 7) the indirect illumination buffers R_{ID} and R'_{ID}. Extracting the dominant light direction L_D, as discussed in Section 5, is solved by using the second, third and fourth SH coefficient of the environment light. While L_D is monochromatic, indirect illumination considers colored light transport based on the albedo $A(s_j)$ of the sample point. However, for differential rendering, we only want to add those pixels to the final result which come from the light interaction between real and virtual, R'_{ID}, and exclude pixels which are affected from the real-world only, since they are present in the camera image already. Therefore, in Equation 7.6, we select the right pixels from R'_{ID}.

$$I_{ID} = \begin{cases} R'_{ID}, & \text{if } R'_{ID} \neq R_{ID} \\ 0, & \text{otherwise} \end{cases} \tag{7.6}$$

$$I_{AR} = I_{DC} + I_{ID} \tag{7.7}$$

The final Augmented Reality image I_{AR} in Equation 7.7 is computed by adding the differential rendering result from the combination of spherical harmonics lighting I_{AR} from Equation 7.2 and indirect lighting I_{ID}. Results of our indirect illumination algorithm are presented in Figure 7.10).

7.4 Evaluation

Figure 7.11 visually compares the rendering results from the previously discussed algorithms for the same scene, where GR12 [36] is based on volume ray-tracing without any acceleration techniques, GR14 [35] shows the results of the rendering with the acceleration techniques based on adaptive subsampling and caching (see Section 6.1). The results based on radiance transfer approximation in image space and consistent shadowing are denoted with GR15 and GR15Fast for the accelerated version, both presented in [37]. The scene shows a virtual figure (dragon) under an office table. While the near field shadows from GR12 and GR14 are more accurate compared to GR15 the visibility rays do not catch the geometry of the table which clearly casts a shadow on the floor. In the result of GR15 and GR15Fast the shadows are more consistent to the real-world shadows since they catch the occlusion from the table. While the difference in visual quality between GR12 and GR14 is almost not visible, we can clearly see some visual degradation from GR15 and GR15Fast where the shadows are darker and more blurry because of the sampling.

7.5 Summary

In this chapter we discussed the Augmented Reality rendering stage enabling real-time Augmented Reality global illumination for mid-sized workspaces more in detail and compared the volume ray-tracing approach and the rendering pipeline based on radiance transfer approximation. We identified consistent shadowing over the entire work space as one key feature for global illumination in Augmented Reality. However, visual artifacts can arise if the entire scene has not been reconstructed. Occlusions cannot be computed consistently then. For example, when only the front of a table is reconstructed, this would create missing shadows under the table. However, this is a fundamental limitation of any single camera approach. An alternative solution would be to use a multiple camera approach such as presented in [49] where multiple MS Kinects are sensing the room from different viewing angles. A further key feature of our global illumination pipeline is the incorporation of indirect illumination. As already mentioned we compute the indirect illumination separately based on the main light direction from the real-world lighting, which provides plausible results from a perceptive point of view but does not claim to be physically correct. A sever limitation to all algorithms which take only the dominant light direction into account is that it limits the algorithm to one primary light direction. This assumption does not hold for situations with multiple strong light sources. For example two equally light sources placed left and right in the room. A solution to that is to compute all dominant light directions from the estimated real-world lighting as proposed in [20]. An alternative would be to expand the radiance transfer function and support more indirect passes, projecting the information into SH. However, while consuming significantly more compute power for indirect rays, this approach would require deeper knowledge about the reflection and surface material properties of the real-world. Since we do not estimate the surface reflection properties we have chosen speed over correctness. This is also the reason why the indirect illumination term has not been used in the actual photometric registration pipeline.

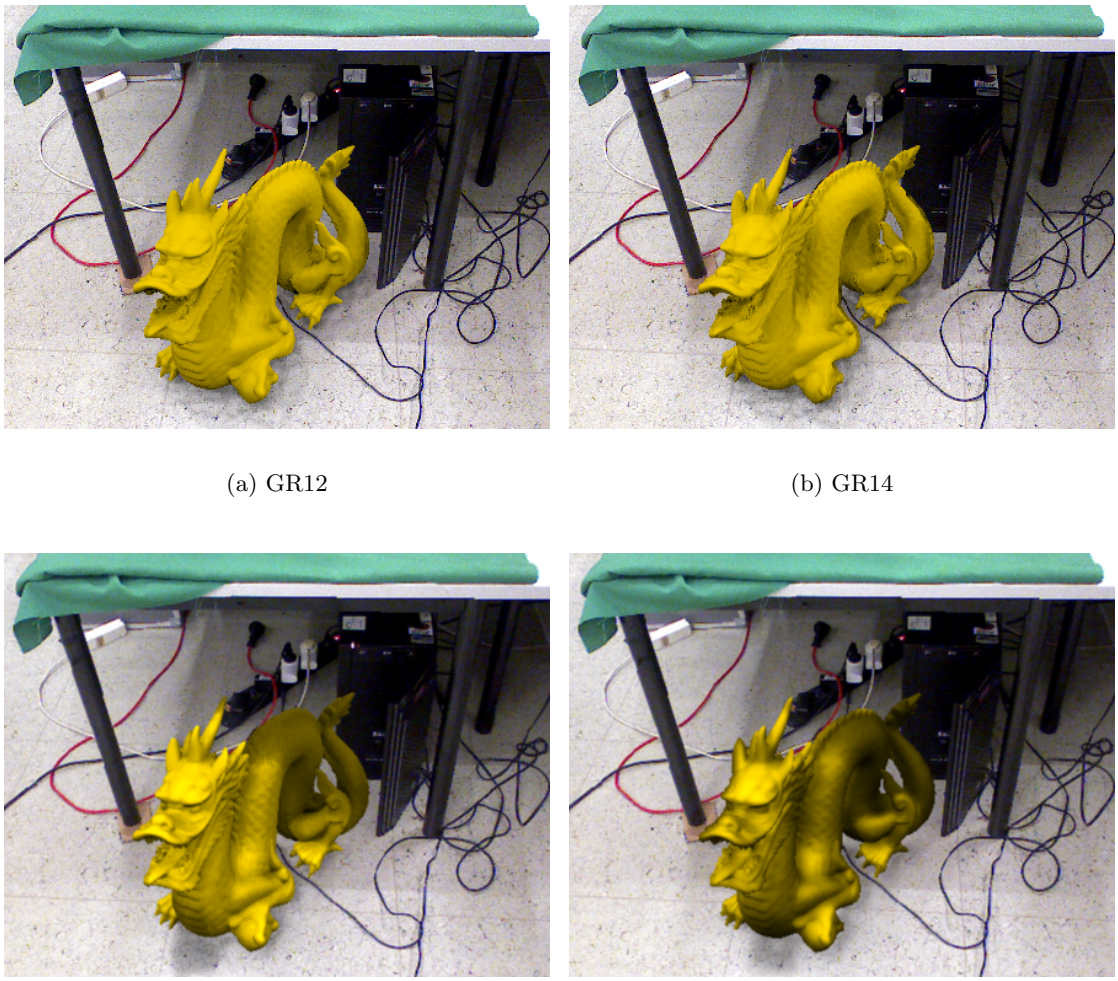
(a) SH lighting

(b) Shading of R' and indirect illumination

(c) With indirect illumination

(d) Difference between (a) and (c)

Figure 7.10: (a) Scene without indirect illumination, (b) Indirect illumination added to the shaded R' buffer. (c) Result with indirect illumination. Note the subtle effects of indirect illumination on real-world diffuse surfaces. (d) Difference image between scene without and with indirect illumination.



(a) GR12 (b) GR14

(c) GR15 (d) GR15Fast

Figure 7.11: Comparison: (a) volume ray-tracing, (b) accelerated volume ray-tracing (c) image based radiance transfer, (d) fast variant of the image based radiance transfer approach in combination with sub-sampling in image space. Due to the limitations of expensive raytracing in (a) and (b), rays have a limited length and do not reach all surfaces. This is in contrast to (d) and (c), which capture lighting and shadowing effects from more distant occluders as well.

8 Conclusion

Contents

8.1 Summary of the Results

The goal of this thesis was to build a photometric registration and Augmented Reality rendering system for dynamically changing environments. We aimed to explore and improve the possibilities of current state-of-the art geometry reconstruction algorithms to provide, in combination with photometric registration, an alternative solution to light probes. More specifically, we built a system which does not require additional light probes for estimating the real-world environment light. Moreover, we showed in our thesis how to integrate the estimated light into a global illumination rendering solution. To demonstrate our algorithms, we built several prototypes and test environments. In the evolution of our work, we identified radiance transfer computation as a key element impacting performance and quality of the final rendering. Therefore, a great part of our work focuses on finding and evaluating different approaches to compute the radiance transfer, meeting the requirements for robust photometric registration and rendering in real-time. Furthermore, we presented a global illumination rendering pipeline for Augmented Reality, which takes dynamically changing geometry and lighting into account. In the following we summarize the major achievements of our work.

Geometry reconstruction: Capturing dynamically changing geometry is a key element for our photometric registration pipeline. In Section 4.2.2, we describe a hybrid geometry reconstruction algorithm, which combines the output of volumetric depth integration over time (see Section 4.2.1) with the immediate output of the depth sensor which is responsive to geometric changes. The volumetric integration provides a higher quality

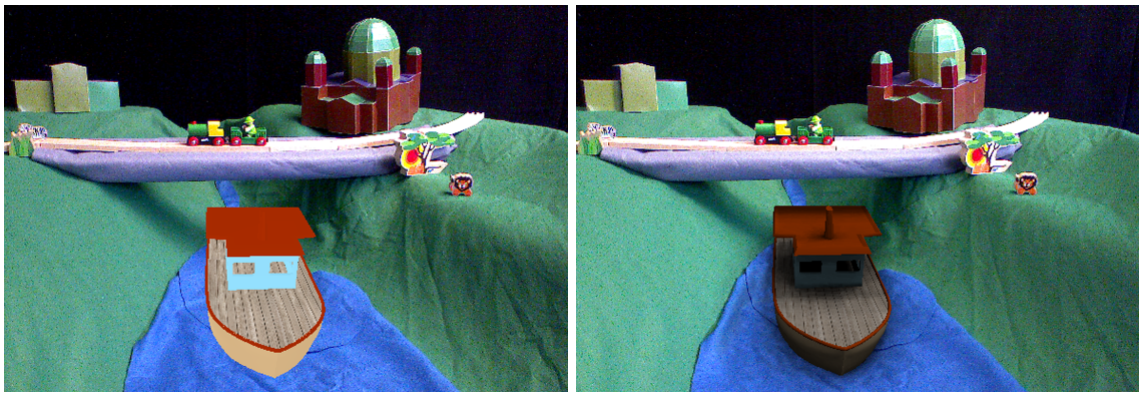


Figure 8.1: The left figure above shows a virtual ship without lighting. The right figure shows the same object lit by our real-world lighting estimation. Note that in this example we support only near field shadowing.

surface reconstruction, which works well for rather static geometry. Our hybrid approach merges this output with the output from the depth sensor to produce a single, filtered geometry buffer consisting of high quality surface data for static geometry and smoothed and filtered geometry data for dynamic geometry. This approach enables more realistic light interaction between real-world geometry and virtual geometry.

Photometric registration: In Chapter 2.2, we discussed our photometric registration approach, which computes the radiance transfer based on the real-world geometry to estimate the environment lighting. We especially demonstrated the benefit of incorporating visibility information into the radiance transfer computation to improve the robustness of the light estimation. Overall, we showed in our evaluations in Section 5.3 that our approach can replace traditional light probes. In Figure 8.1, we show examples of the impact of our photometric registration algorithm for Augmented Reality rendering. Where we show the same scene with and without the estimated real-world lighting applied on the virtual object and the ability of estimating lighting from unprepared scenes (see Figure 8.2).

Performance: We identified that computing the radiance transfer with visibility information, requiring secondary rays, on dynamically changing geometry imposed, the biggest performance bottleneck. Therefore, we developed several acceleration techniques (see Chapter 6) to achieve better frame rates. We especially highlight the fact that each acceleration technique has been developed under the objective of preserving light estimation quality and visual quality at the same time. From evaluating and comparing our different approaches in Section 6.2.3, we can conclude that the acceleration technique described in Section 6.2 based on approximation of the radiance transfer in image space provides superior performance over the acceleration techniques based on volume ray-tracing in terms of runtime (see Table 6.2) and creates comparable results regarding light estimation and rendering quality. We also favor this approach, since the speed gain enables a higher grade of interactivity, which is crucial for Augmented Reality and gives the possibility to run the system on mobile hardware. Moreover, as already discussed in Section 6.2.3, a

Figure 8.2: Left image: real-world bridge casts a shadow onto the virtual ship and the virtual ship casts shadows onto the real-world environment. Right image: the direction of the virtual shadows can be compared with the real-world shadows, as cast by the little lion, for example. Middle insert: we added a virtual hat onto the head of a real person to demonstrate the ability of the system to work in unprepared environments, too.

combination of the acceleration techniques is possible, as we demonstrated by additionally performing regular sampling in image space.

Augmented Reality rendering: In this thesis, we cover several different techniques related to the field of visually coherent rendering for Augmented Reality. Seamless blending between real and virtual is our main motivation. By incorporating real-world lighting into the differential rendering pipeline discussed in Section 7.1, we are able to model dynamic lighting effects between real and virtual geometry. Furthermore, we show that, with the right combination of geometry reconstruction and rendering methods, it is possible to compute global illumination effects in real time for mid-sized workspaces. In Figure 8.3, we show shadowing results created in our test environments, while, in Figure 8.4, we present a series of images taken from a video sequence to demonstrate indirect illumination between a real book and a virtual figure.

8.2 Limitations Summary

In this section, we point out the limitations we encountered while building and evaluating our photometric registration and rendering system.

- **Real world material surface properties estimation:** For the photometric registration, we assume a Lambertian gray world and do not estimate the material color or the reflectance properties from the real-world geometry. By using dense sampling of the scene for the photometric registration, we treat the absence of material color information as noise. High-frequency textures, where the surface color or intensities change quickly in the spatial domain are suppressed by filtering the camera image.

Figure 8.3: Consistent shadowing in mid-sized environments. Left: The dragon under the bed catches the shadow from the bed while the dragon in front of the girl casts a shadow onto the carpet. Right: The red paddle (real) has the same shadow line as the hat of Luigi.

However, extreme cases, where the surface geometry is very dark to almost black and covers huge part of the working space, can lead to biased light estimations. Moreover, we do not estimate the bidirectional reflectance distribution function (BRDF) of the surface. Knowledge about the BRDF would contribute to the photometric registration and to the global illumination rendering.

- **Surface normal vector distribution:** The light estimation heavily depends on the surface normal vectors of the scene. The more different surface normal vectors are present the better, for the scene reconstruction. Therefore, scenes with only a few different surface vector normals, such as a plain table or a floor, provide restricted information for light estimation, and thus lead to a biased result. We show that we can overcome this problem to a certain extent by weighting the input data.

- **Local light sources:** We do not support local light sources in the FOV, such as candles or torch lights. These are cases which have to be considered separately. Our solution covers a good range of lighting scenarios for a broad category of Augmented Reality applications.

- **Reconstruction:** Our photometric reconstruction depends on the quality of the surface geometry reconstruction. Naturally, this quality can have an impact on the overall application. We showed in our work that the reconstruction for Augmented Reality has to react to dynamic scene changes while preserving the quality of the reconstruction. However, we want to point out that our approach does not require a specific reconstruction solution. The geometry reconstruction algorithm can be easily substituted by future alternative solutions, which might provide better results.

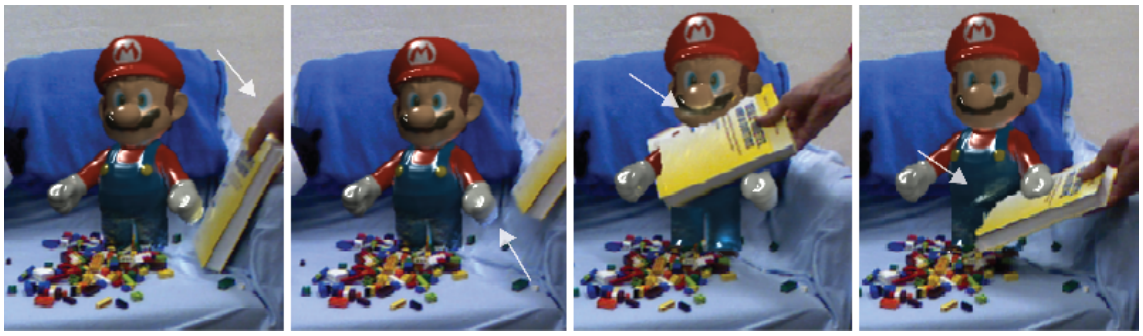


Figure 8.4: This figure shows a sequence of images from a video. The main light of the real-world lighting comes from the left. The virtual character has a shiny and reflective material, as indicated by the specular highlights and rather glossy interreflections. From left to right: The yellow book sends reflections to the right hand of the character. The book is removed. The right hand receives indirect illumination from the couch. The book is now waved through the character to demonstrate the dynamic geometry reconstruction capabilities of our approach, providing correct occlusions between real and virtual and supporting indirect illumination effects.

8.3 Future Directions

Following the path of photometric registration research for Augmented Reality we see various different future directions.

To improve photometric registration, more investigations into exploiting the input data from the RGB camera would be valuable. In particular, this could be the incorporation of the camera exposure to introduce high dynamic range into the light estimation pipeline, to improve the sensitivity of the light estimation.

Investigating and building a real-time system combining light estimation and surface material estimation from dynamically changing environments would be a further logical path to go. Knowledge about surface material properties would improve the photometric registration and support more physically correct indirect illumination effects. Moreover, this system could be used to improve real-time intrinsic image decomposition as well. For example, the estimated real-world lighting, including the radiance transfer with occlusion information, could be used for initializing intrinsic image decomposition algorithms.

Furthermore, investigating the impact and applicability of compressing radiance transfer using wavelets instead of spherical harmonics would be an interesting problem. Wavelets [82] can represent view-dependent lighting effects such as specular hightlights, which would result into a more descriptive and physically plausible radiance transfer representation.

Our assumption of distant light sources does not hold in all real-world scenarios. Local lighting effects, e.g., a torch light, would be neglected by our system. To add local lighting effects, we have to estimate the lighting for local regions and consider these solutions during rendering.

We believe that adding real-world lighting to Augmented Reality rendering is part of

a natural development, which has been demonstrated for many years by the film industry, creating stunning visual effects by producing a compelling merging of virtual and real. This thesis brought us a step closer to photorealistic Augmented Reality and interactive movies.

Bibliography

[1] Agarwal, S., Snavely, N., Simon, I., Seitz, S. M., and Szeliski, R. (2009). Building rome in a day. In *Twelfth IEEE International Conference on Computer Vision (ICCV 2009)*, Kyoto, Japan. IEEE. (page 14)

[2] Aittala, M. (2010). Inverse lighting and photorealistic rendering for augmented reality. *The Visual Computer*, 26(6-8):669–678. (page 17, 18, 34, 75)

[3] Arief, I., McCallum, S., and Hardeberg, J. Y. (2012). Realtime estimation of illumination direction for augmented reality on mobile devices. In *Color and Imaging Conference*, pages 111–116, Los Angeles, CA, USA. IS&T and SID. (page 20)

[4] Azuma, R. (1995). A Survey of Augmented Reality. 6:355–385. (page 2)

[5] Bajura, M., Fuchs, H., and Ohbuchi, R. (1992). Merging virtual objects with the real world: Seeing ultrasound imagery within the patient. computer graphics. vol 26, no 2. july. (page 4)

[6] Barron, J. T. and Malik, J. (2013). Intrinsic scene properties from a single rgb-d image. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '13, pages 17–24, Washington, DC, USA. IEEE Computer Society. (page 17)

[7] Bell, S., Bala, K., and Snavely, N. (2014). Intrinsic images in the wild. *ACM Trans. on Graphics (SIGGRAPH)*, 33(4). (page 17)

[8] Berger, M., Tagliasacchi, A., Seversky, L. M., Alliez, P., Levine, J. A., Sharf, A., and Silva, C. (2014). State of the art in surface reconstruction from point clouds. *Eurographics STAR (Proc. of EG'14)*. (page 14)

[9] Blake, A. (1985). Boundary conditions for lightness computation in mondrian world. *Computer Vision, Graphics, and Image Processing*, 32(3):314–327. (page 16)

[10] Boom, B., Orts-Escolano, S., Ning, X., McDonagh, S., Sandilands, P., and Fisher, R. (2013). *Point light source estimation based on scenes recorded by a RGB-D camera.* (page 21, 26)

[11] Cao, X. and Foroosh, H. (2007). Camera calibration and light source orientation from solar shadows. *Comput. Vis. Image Underst.*, 105(1):60–72. (page 23)

[12] Cao, X. and Shah, M. (2005). M.: Camera calibration and light source estimation from images with shadows. In *In Proceedings: IEEE Conference on Computer Vision and Pattern Recognition*, pages 928–923. Academic Press. (page 23)

[13] Caudell, T. and Mizell, D. (1992). Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, pages 659–669 vol.2. IEEE. (page 2)

[14] Chang, J., Cabezas, R., and Fisher, JohnW., I. (2014). Bayesian nonparametric intrinsic image decomposition. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision - ECCV 2014*, volume 8692 of *Lecture Notes in Computer Science*, pages 704–719. Springer International Publishing. (page 16)

[15] Chen, J., Bautembach, D., and Izadi, S. (2013). Scalable real-time volumetric surface reconstruction. *ACM Trans. Graph.*, 32(4):113:1–113:16. (page 15, 50)

[16] Chen, Q. and Koltun, V. (2013). A simple model for intrinsic image decomposition with depth cues. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 241–248. (page 17)

[17] Crassin, C., Neyret, F., Sainz, M., Green, S., and Eisemann, E. (2011). Interactive indirect illumination using voxel cone tracing. In *Computer Graphics Forum*, volume 30, pages 1921–1930. Wiley Online Library. (page 26)

[18] Csongei, M., Hoang, L., Sandor, C., and Lee, Y. B. (2014). Global illumination for augmented reality on mobile phones. In *Virtual Reality (VR), 2014 iEEE*, pages 69–70. (page 26)

[19] Debevec, P. (1998). Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 189–198. ACM. ACM ID: 280864. (page 9, 17, 18, 33, 34, 75, 83)

[20] Debevec, P. (2006). A median cut algorithm for light probe sampling. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA. ACM. (page 92)

[21] Debevec, P., Wenger, A., Tchou, C., Gardner, A., Waese, J., and Hawkins, T. (2002). A lighting reproduction approach to live-action compositing. Technical report. (page 23)

[22] Debevec, P. E. and Malik, J. (1997). Recovering high dynamic range radiance maps from photographs. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 369–378. ACM Press/Addison-Wesley Publishing Co. ACM ID: 258884. (page 18)

[23] DiVerdi, S., Wither, J., and Hollerei, T. (2008). Envisor: Online environment map construction for mixed reality. In *Virtual Reality Conference, 2008. VR '08. IEEE*, pages 19–26. (page 19)

[24] Engel, J., Schöps, T., and Cremers, D. (2014). LSD-SLAM: large-scale direct monocular SLAM. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II*, pages 834–849. (page 14, 30, 78)

[25] Foix, S., Alenya, G., and Torras, C. (2011). Lock-in time-of-flight (tof) cameras: A survey. *Sensors Journal, IEEE*, 11(9):1917–1926. (page 14)

[26] Fournier, A., Gunawan, A. S., and Romanzin, C. (1993). Common illumination between real and computer generated scenes. In *Graphics Interface '93*, pages 254–262. (page 4)

[27] Franke, T. A. (2013). Delta light propagation volumes for mixed reality. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 125–132. (page 24, 26)

[28] Franke, T. A. (2014). Delta voxel cone tracing. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 39–44. (page 25, 26, 27)

[29] Freeman, W. and Pasztor, E. (1999). Learning low-level vision. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1182–1189 vol.2. (page 16)

[30] Funt, B. V., Drew, M. S., and Brockington, M. (1991). Recovering shading from color images. In *ECCV-92: Second European Conference on Computer Vision*, pages 124–132. Springer-Verlag. (page 16)

[31] Grosch, T. (2005). Differential photon mapping: Consistent augmentation of photographs with correction of all light paths. In *Eurographics 2005 Short Papers, Trinity College, Dublin, Ireland*. (page 25)

[32] Grosch, T., Eble, T., and Mueller, S. (2007). Consistent interactive augmentation of live camera images with correct near-field illumination. In *Proceedings of the 2007 ACM Symposium on Virtual Reality Software and Technology*, VRST '07, pages 125–132, New York, NY, USA. ACM. (page 23, 24)

[33] Grosch, T., Mueller, S., and Kresse, W. (2003). Goniometric light reconstruction for augmented reality image synthesis. In *Proc. GI Jahrestagung*, Frankfurt, Germany. (page 17, 18)

[34] Gruber, L., Gauglitz, S., Ventura, J., Zollmann, S., Huber, M., Schlegel, M., Klinker, G., Schmalstieg, D., and Hollerer, T. (2010). The city of sights: Design, construction, and measurement of an augmented reality stage set. In *Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on*, pages 157–163. (page 10, 35, 36, 54)

[35] Gruber, L., Langlotz, T., Sen, P., Hoherer, T., and Schmalstieg, D. (2014a). Efficient and robust radiance transfer for probeless photorealistic augmented reality. In *2014 IEEE Virtual Reality, VR 2014, Minneapolis, MN, USA, March 29 - April 2, 2014*, pages 15–20. (page 10, 42, 50, 51, 59, 77, 87, 91)

[36] Gruber, L., Richter-Trummer, T., and Schmalstieg, D. (2012). Real-time photometric registration from arbitrary geometry. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 119 –128. (page 8, 42, 49, 50, 58, 59, 66, 75, 84, 87, 91)

[37] Gruber, L., Ventura, J., and Schmalstieg, D. (2014b). Efficient and robust radiance transfer for probeless photorealistic augmented reality. In *2014 IEEE Virtual Reality, VR 2014, Minneapolis, MN, USA, March 29 - April 2, 2014*, pages 15–20. (page 7, 10, 72, 91)

[38] Haller, M., Drab, S., and Hartmann, W. (2003). A real-time shadow approach for an augmented reality application using shadow volumes. In *VRST*, pages 56–65. (page 20)

[39] Hara, K., Nishino, K., and Ikeuchi, K. (2003). Determining reflectance and light position from a single image without distant illumination assumption. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 560–567. IEEE. (page 22)

[40] Hara, K., Nishino, K., and lkeuchi, K. (2005). Light source position and reflectance estimation from a single view without the distant illumination assumption. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(4):493–505. (page 22)

[41] Hauswiesner, S., Straka, M., and Reitmayr, G. (2013). Virtual try-on through image-based rendering. *IEEE Trans. Vis. Comput. Graph.*, 19(9):1552–1565. (page 32)

[42] Ikeda, T., Oyamada, Y., Sugimoto, M., and Saito, H. (2012). Illumination estimation from shadow and incomplete object shape captured by an rgb-d camera. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 165–169. (page 21)

[43] Irschara, A., Zach, C., Frahm, J.-M., and Bischof, H. (2009). From structure-from-motion point clouds to fast location recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2599–2606. (page 14)

[44] Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., and Fitzgibbon, A. (2011). KinectFusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *UIST'11*. (page 15, 44)

[45] Jachnik, J., Newcombe, R. A., and Davison, A. J. (2012). Real-time surface light-field capture for augmentation of planar specular surfaces. In *ISMAR*, pages 91–97. (page 4, 19)

[46] Jacobs, K. and Loscos, C. (2006). Classification of illumination methods for mixed reality. *Computer Graphics Forum*, 25(1):29–51. (page 16, 23)

[47] Jain, S. (2003). A survey of laser range finding. (page 14)

[48] Jimenez, J., Echevarria, J. I., Sousa, T., and Gutierrez, D. (2012). Smaa: Enhanced morphological antialiasing. *Computer Graphics Forum (Proc. EUROGRAPHICS 2012)*, 31(2). (page 78)

[49] Kainz, B., Hauswiesner, S., Reitmayr, G., Steinberger, M., Grasset, R., Gruber, L., Veas, E., Kalkofen, D., Seichter, H., and Schmalstieg, D. (2012). OmniKinect: real-time dense volumetric data acquisition and applications. In *VRST '12: Proceedings of the 2012 ACM symposium on Virtual reality software and technology*. (page 8, 92)

[Kan and Kaufmann] Kan, P. and Kaufmann, H. High-quality reflections, refractions, and caustics in augmented reality and their contribution to visual coherence. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 99–108. (page 18, 25, 27)

[51] Kan, P. and Kaufmann, H. (2013). Differential irradiance caching for fast high-quality light transport between virtual and real worlds. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 133–141. (page 4, 18, 26, 27)

[52] Kanbara, M. and Yokoya, N. (2004). Real-time estimation of light source environment for photorealistic augmented reality. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2 - Volume 02*, pages 911–914. IEEE Computer Society. (page 17, 18)

[53] Kaplanyan, A. and Dachsbacher, C. (2010). Cascaded light propagation volumes for real-time indirect illumination. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, pages 99–107. ACM. (page 24)

[54] Karsch, K., Hedau, V., Forsyth, D., and Hoiem, D. (2011). Rendering synthetic objects into legacy photographs. In *Proceedings of the 2011 SIGGRAPH Asia Conference*, SA '11, pages 157:1–157:12, New York, NY, USA. ACM. (page 17, 31)

[55] Karsch, K., Sunkavalli, K., Hadap, S., Carr, N., Jin, H., Fonte, R., Sittig, M., and Forsyth, D. (2014). Automatic scene inference for 3d object compositing. *ACM Trans. Graph.*, 33(3). (page 17)

[56] Kato, H., Billinghurst, M., Poupyrev, I., Imamoto, K., and Tachibana, K. (2000). Virtual object manipulation on a table-top ar environment. In *Augmented Reality, 2000. (ISAR 2000). Proceedings. IEEE and ACM International Symposium on*, pages 111–119. (page 34)

[57] Keller, A. (1997). Instant radiosity. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 49–56, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co. (page 24)

[58] Keller, M., Lefloch, D., Lambers, M., Izadi, S., Weyrich, T., and Kolb, A. (2013). Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision (3DV)*. (page 15)

[59] Kerl, C., Sturm, J., and Cremers, D. (2013). Dense visual SLAM for RGB-d cameras. In *Proc. of the Int. Conf. on Intelligent Robot Systems (IROS)*. (page 15)

[60] Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small ar workspaces. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, ISMAR '07, pages 1–10, Washington, DC, USA. IEEE Computer Society. (page 14)

[61] Knecht, M., Traxler, C., Mattausch, O., Purgathofer, W., and Wimmer, M. (2010). Differential instant radiosity for mixed reality. In *2010 9th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 99–107. IEEE. (page 4, 18, 24, 25, 26)

[62] Knecht, M., Traxler, C., Winklhofer, C., and Wimmer, M. (2013). Reflective and refractive objects for mixed reality. *Visualization and Computer Graphics, IEEE Transactions on*, 19(4):576–582. (page 26, 27)

[63] Knorr, S. and Kurz, D. (2014). Real-time illumination estimation from faces for coherent rendering. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 113–122. (page 21, 22)

[64] Kolivand, H., Hasan Zakaria, A., and Sunar, M. S. (2014). Shadow generation in mixed reality: A comprehensive survey. *IETE Technical Review*, (ahead-of-print):1–13. (page 23)

[65] Lagger, P. and Fua, P. (2006). Using specularities to recover multiple light sources in the presence of texture. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 1, pages 587–590. (page 22)

[66] Land, E. H., John, and Mccann, J. (1971). Lightness and retinex theory. *Journal of the Optical Society of America*, pages 1–11. (page 16)

[67] Lee, K., Zhao, Q., Tong, X., Gong, M., Izadi, S., Lee, S., Tan, P., and Lin, S. (2012). Estimation of intrinsic image sequences from image+depth video. In Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., and Schmid, C., editors, *Computer Vision - ECCV 2012*, volume 7577 of *Lecture Notes in Computer Science*, pages 327–340. Springer Berlin Heidelberg. (page 17)

[Lensing and Broll] Lensing, P. and Broll, W. Instant indirect illumination for dynamic mixed reality scenes. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 109–118. (page 4, 15, 25)

[69] Liu, Y. and Granier, X. (2012). Online tracking of outdoor lighting variations for augmented reality with moving cameras. *Visualization and Computer Graphics, IEEE Transactions on*, 18(4):573–580. (page 23)

[70] Lopez-Moreno, J., Garces, E., Hadap, S., Reinhard, E., and Gutierrez, D. (2013). Multiple light source estimation in a single image. *Computer Graphics Forum*, 32(8):170–182. (page 17)

[71] Lopez-Moreno, J., Hadap, S., Reinhard, E., and Gutierrez, D. (2010). Computer graphics in spain: A selection of papers from ceig 2009: Compositing images through light source detection. *Comput. Graph.*, 34(6):698–707. (page 17)

[72] Madsen, C. B. and Nielsen, M. (2008). Towards probe-less augmented reality - a position paper. In *GRAPP*, pages 255–261. (page 23)

[73] Maimone, A. and Fuchs, H. (2011). Encumbrance-free telepresence system with real-time 3d capture and display using commodity depth cameras. In *Proc. ISMAR '11*, pages 137–146. (page 15)

[74] Mantiuk, R., Kim, K. J., Rempel, A. G., and Heidrich, W. (2011). HDR-VDP-2: a calibrated visual metric for visibility and quality predictions in all luminance conditions. *ACM Trans. Graph.*, 30(4):40:1–40:14. (page 69)

[75] Marschner, S. R. and D, P. (1998). Inverse rendering for computer graphics. (page 16)

[76] Marton, Z. C., Rusu, R. B., and Beetz, M. (2009). On Fast Surface Reconstruction Methods for Large and Noisy Datasets. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan. (page 40)

[77] Mashita, T., Yasuhara, H., Plopski, A., Kiyokawa, K., and Takemura, H. (2013). In-situ lighting and reflectance estimations for indoor ar systems. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 275–276. (page 22)

[78] Meilland, M., Barat, C., and Comport, A. (2013). 3d high dynamic range dense visual SLAM and its application to real-time object re-lighting. In *Proceedings IEEE ISMAR 2013*. (page 4, 19)

[79] Milgram, P. and Kishino, F. (1994). A Taxonomy of Mixed Reality Visual Displays. *IEICE Transations on Information and Systems*, E77-D(12):1321–1329. (page 2)

[80] Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011a). KinectFusion: Real-time dense surface mapping and tracking. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, page 127i¿½136, Washington, DC, USA. IEEE Computer Society. (page 15, 26, 31, 32, 59, 77)

[81] Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011b). Dtam: Dense tracking and mapping in real-time. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2320–2327, Washington, DC, USA. IEEE Computer Society. (page 14, 30)

[82] Ng, R., Ramamoorthi, R., and Hanrahan, P. (2004). Triple product wavelet integrals for all-frequency relighting. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 477–487, New York, NY, USA. ACM. (page 99)

[83] Nillius, P. and Eklundh, J.-O. (2001). Automatic estimation of the projected light source direction. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–1076–I–1083 vol.1. (page 17)

[84] Nishina, Y., Okumura, B., Kanbara, M., and Yokoya, N. (2008). Photometric registration by adaptive high dynamic range image generation for augmented reality. In *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*, pages 53–56. (page 18)

[85] Nowrouzezahrai, D., Geiger, S., Mitchell, K., Sumner, R., Jarosz, W., and Gross, M. (2011). Light factorization for mixed-frequency shadows in augmented reality. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 173–179, Washington, DC, USA. IEEE Computer Society. (page 4, 24, 69)

[86] Pan, M., Wang Xinguo Liu, R., Peng, Q., and Bao, H. (2007). Precomputed radiance transfer field for rendering interreflections in dynamic scenes. In *Computer Graphics Forum*, volume 26, pages 485–493. Wiley Online Library. (page 24)

[87] Patow, G. and Pueyo, X. (2003). A survey of inverse rendering problems. *Comput. Graph. Forum*, 22(4):663–688. (page 16)

[88] Pentland, A. P. (1982). Finding the illuminant direction. *J. Opt. Soc. Am.*, 72(4):448–455. (page 17)

[89] Pessoa, S. A., Moura, G. d. S., Lima, J. P. S. d. M., Teichrieb, V., and Kelner, J. (2012). Real-time photorealistic rendering of synthetic objects into real scenes. *Comput. Graph.*, 36(2):50–69. (page 25)

[90] Pessoa, S. A., Moura, G. d. S., Lima, J. P. S. M., Teichrieb, V., and Kelner, J. (2009). A global illumination and BRDF solution applied to photorealistic augmented reality. *Virtual Reality Conference, IEEE*, 0:243–244. (page 25)

[91] Pilet, J., Geiger, A., Lagger, P., Lepetit, V., and Fua, P. (2006). An all-in-one solution to geometric and photometric calibration. In *IEEE/ACM International Symposium on Mixed and Augmented Reality, 2006. ISMAR 2006*, pages 69–78. IEEE. (page 19)

[92] Plopski, A., Mashita, T., Kiyokawa, K., and Takemura, H. (2014). Reflectance and light source estimation for indoor ar applications. In *Virtual Reality (VR), 2014 iEEE*, pages 103–104. (page 22)

[93] Powell, M., Sarkar, S., and Goldgof, D. (2001). A simple strategy for calibrating the geometry of light sources. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(9):1022–1027. (page 18)

[94] Ramamoorthi, R. and Hanrahan, P. (2001a). An efficient representation for irradiance environment maps. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 497–500. ACM. ACM ID: 383317. (page 16)

[95] Ramamoorthi, R. and Hanrahan, P. (2001b). A signal-processing framework for inverse rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 117–128. ACM. ACM ID: 383271. (page 16, 51)

[96] Reinhard, E., Stark, M., Shirley, P., and Ferwerda, J. (2002). Photographic tone reproduction for digital images. *ACM Trans. Graph.*, 21(3):267–276. (page 83)

[97] Ren, Z., Wang, R., Snyder, J., Zhou, K., Liu, X., Sun, B., Sloan, P.-P., Bao, H., Peng, Q., and Guo, B. (2006). Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. *ACM Trans. Graph.*, 25(3):977–986. (page 24)

[98] Richardt, C., Stoll, C., Dodgson, N. A., Seidel, H.-P., and Theobalt, C. (2012). Coherent spatiotemporal filtering, upsampling and rendering of RGBZ videos. In *Eurographics*. (page 14, 15, 42)

[99] Ritschel, T., Dachsbacher, C., Grosch, T., and Kautz, J. (2012). The state of the art in interactive global illumination. *Comput. Graph. Forum*, 31(1):160–188. (page 23)

[100] Ritschel, T., Grosch, T., Kim, M. H., Seidel, H.-P., Dachsbacher, C., and Kautz, J. (2008). Imperfect shadow maps for efficient computation of indirect illumination. *ACM Trans. Graph.*, 27(5):129:1–129:8. (page 24)

[101] Ritschel, T., Grosch, T., and Seidel, H.-P. (2009). Approximating Dynamic Global Illumination in Screen Space. In *Proceedings ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. (page 72, 90)

[102] Rusu, R. B. and Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China. (page 40)

[103] Schöps, T., Engel, J., and Cremers, D. (2014). Semi-dense visual odometry for AR on a smartphone. (page 78)

[104] Seitz, S., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 519–528. (page 14)

[105] Shah, M., Konttinen, J., and Pattanaik, S. (2007). Caustics mapping: An image-space technique for real-time caustics. *Visualization and Computer Graphics, IEEE Transactions on*, 13(2):272–280. (page 26)

[106] Shen, L., Tan, P., and Lin, S. (2008). Intrinsic image decomposition with non-local texture cues. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7. (page 16)

[107] Shim, H. (2012). Estimating all frequency lighting using a color/depth image. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 565–568. (page 22)

[108] Sloan, P.-P., Kautz, J., and Snyder, J. (2002). Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph.*, 21(3):527ï¿½536. ACM ID: 566612. (page 33, 61)

[109] Sobierajski, L. M. and Avila, R. S. (1995). A hardware acceleration method for volumetric ray tracing. In *Proceedings of the 6th Conference on Visualization '95*, VIS '95, pages 27–, Washington, DC, USA. IEEE Computer Society. (page 62)

[110] Stauder, J. (1999). Augmented reality with automatic illumination control incorporating ellipsoidal models. *IEEE TRANS. MULTIMEDIA*, 1:136–143. (page 20, 21)

[111] Sutherland, I. E. (1968). A head-mounted three dimensional display. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I on - AFIPS '68 (Fall, part I)*, page 757, New York, New York, USA. ACM Press. (page 2)

[112] Tomasi, C. and Kanade, T. (1992). Shape and motion from image streams under orthography: A factorization method. *Int. J. Comput. Vision*, 9(2):137–154. (page 14)

[113] Van Den Hengel, A., Sale, D., and Dick, A. R. (2009). SecondSkin: An interactive method for appearance transfer. *Computer Graphics Forum*, 28(7):1735–1744. (page 17)

[114] Čadík, M., Herzog, R., Mantiuk, R., Myszkowski, K., and Seidel, H.-P. (2012). New measurements reveal weaknesses of image quality metrics in evaluating graphics artifacts. *ACM Trans. Graph.*, 31(6):147:1–147:10. (page 69)

[115] Wagner, D., Reitmayr, G., Mulloni, A., and Schmalstieg, D. (2009). Real time detection and tracking for augmented reality on mobile phones. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):355–368. (page 40)

[116] Wang, Y. and Samaras, D. (2003). Estimation of multiple directional light sources for synthesis of augmented reality images. *Graphical Models*, 65(4):185–205. (page 20, 21)

[117] Weber, M., Blake, A., and Cipolla, R. (2004). Towards a complete dense geometric and photometric reconstruction under varying pose and illumination. *Image and Vision Computing*, 22(10):787–793. (page 21)

[118] Wyman, C. and Davis, S. (2006). Interactive image-space techniques for approximating caustics. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, I3D '06, pages 153–160, New York, NY, USA. ACM. (page 26)

[119] Yu, Y., Debevec, P., Malik, J., and Hawkins, T. (1999). Inverse global illumination: Recovering reflectance models of real scenes from photographs. pages 215–224. (page 18)

[120] Zach, C., Pock, T., and Bischof, H. (2007). A duality based approach for realtime tv-l1 optical flow. In *In Ann. Symp. German Association Patt. Recogn*, pages 214–223. (page 50)

[121] Zhang, L., Curless, B., and Seitz, S. M. (2002). Rapid shape acquisition using color structured light and multi-pass dynamic programming. In *In The 1st IEEE International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 24–36. (page 31)

[122] Zhou, F., Duh, H.-L., and Billinghurst, M. (2008). Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. In *7th IEEE/ACM International Symposium on Mixed and Augmented Reality, 2008. ISMAR 2008*, pages 193–202. (page 13)