Improving 3D reconstruction via RGB-D camera registration and shading-based surface refinement

Deng, Teng

2019

Deng, T. (2019). Improving 3D reconstruction via RGB-D camera registration and shading-based surface refinement. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/88852>

<https://doi.org/10.32657/10220/47633>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**
SINGAPORE

Improving 3D Reconstruction via RGB-D Camera Registration and Shading-based Surface Refinement

DENG TENG

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2019

Improving 3D Reconstruction via RGB-D Camera Registration and Shading-based Surface Refinement

Deng Teng

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

2019

Abstract

The process of generating a three-dimensional (3D) computer model of a real-world object or scene from two-dimensional images is called *3D Reconstruction*. Nowadays, the emergence of consumer-grade RGB-D cameras brings opportunities and challenges in 3D reconstruction. On one hand, geometric information can be captured with ease using these consumer-grade RGB-D camera, resulting in a number of interesting and exciting 3D applications. On the other hand, the quality of the 3D data suffers from distortions and noises, limiting the quality of the 3D reconstruction. Shapes obtained from consumer-grade RGB-D cameras are in general of low quality and usually lack of enough surface details. Meanwhile, great progresses have been made on the surface recovery using shape-from-shading (SfS), in which surface details can be obtained by analyzing the relationship between lighting conditions, object materials and geometries. This thesis enhances the recent 3D reconstruction techniques by improving the calibration and registration of the depth capturing devices and refining surface details of rough and noisy initial shape through SfS.

Firstly, we introduce a multiple RGB-D camera registration method using local rigid transforms. While impressive 3D reconstruction results have been obtained, combining data acquired by multiple RGB-D cameras constitutes a technical challenge. Several methods have been proposed to estimate the internal parameters of each RGB-D camera (such as depth mapping function and focal length). Despite that the textured geometry obtained by each RGB-D camera individually is visually attractive, even state-of-the-art methods have difficulties in correctly combining the textured geometries obtained by several RGB-D cameras via a rigid transformation. Based on this observation, our approach registers the RGB-D cameras by a smooth field of rigid transformations, instead of a single rigid transformation. Experimental results on challenging data demonstrate the validity of the proposed approach.

Secondly, we propose a method for directly registering multiple depth cameras using everyday objects. It is a more practical method for conducting and maintaining registration of multi-depth sensors, via replacing checkerboards with everyday objects found in the scene, such as regular furniture. Particularly, high quality pre-scanned 3D shapes of standard furniture are used as calibration targets. We propose a unified framework that jointly computes the optimal extrinsic calibration and depth correction parameters. Experimental results show that our proposed method significantly outperforms state-of-the-art depth camera registration methods.

Thirdly, we present subdivision-based representations for both lighting and geometry in shape-from-shading. A very recent shading-based method introduced a per-vertex

overall illumination model for surface reconstruction, which has the advantages of conveniently handling complicated lighting condition and avoiding explicit estimation of visibility and varied albedo. However, due to its discrete nature, the per-vertex overall illumination requires a large amount of memory and lacks intrinsic coherence. To overcome these problems, we propose to use classic subdivision to define the basic smooth lighting function and surface and introduce additional independent variables into the subdivision to adaptively model sharp changes of illumination and geometry. Compared to previous works, the new model not only preserves the merits of the per-vertex illumination model, but also greatly reduces the number of variables required in surface recovery and intrinsically regularizes the illumination vectors and the surface. These features make the new model very suitable for surface detail recovery under general, unknown illumination condition. Particularly, a variational surface reconstruction method built upon the subdivision representations for lighting and geometry is developed. The experiments on both synthetic and real-world datasets have demonstrated that the proposed method can achieve memory efficiency and improve surface detail recovery.

Lastly, we improve the subdivision-based lighting model and exploit parallel scheme of the algorithm for GPU implementation to achieve real-time performance. Particularly, we present a spatial-varying illumination model for shading-based depth refinement based on a smooth Spherical Harmonics (SH) lighting field. The proposed lighting model can recover shading under challenging unknown lighting conditions, thus improving the quality of recovered surface detail. To avoid over-parameterization, local lighting coefficients are treated as a vector-valued function which is represented by subdivided surfaces using Catmull-Clark subdivision. We solve our lighting model utilizing a highly parallelized scheme that recovers lighting in a few milliseconds. A real-time shading-based depth recovery system is implemented with the integration of our proposed lighting model. We conduct quantitative and qualitative evaluations on both synthetic and real-world datasets under challenging illumination. The experimental results show our method outperforms the state-of-the-art real-time shading-based depth refinement system.

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

29-Jan-2019

.....

Date

.....

Deng Teng

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

29-Jan-2019

.....
Date

.....
Prof. Cai Jianfei

Authorship Attribution Statement

This thesis contains material from 4 paper(s) published in the following peer-reviewed journal(s) where I was the first and/or corresponding author.

Chapter 3 is published as Deng, T., Bazin, J.-C., Martin, T., Kuster, C., Cai, J., Popa, T., and Gross, M. (2014). Registration of multiple rgbd cameras via local rigid transformations. In Proceedings of International Conference on Multimedia and Expo, pages 1-6. IEEE.

The contributions of the co-authors are as follows:

- Prof. Cai provided the initial project direction and edited the manuscript drafts.
- Dr. Bazin, Dr. Martin, Dr. Kuster and Dr. Popa provided suggestions on the proposed method and assisted in the experiments.
- I prepared the manuscript drafts. The manuscript was revised by Dr. Bazin, Dr. Martin, Dr. Kuster, Dr. Popa and Prof. Gross.

Chapter 4 is published as Deng, T., Cai, J., Cham, T.-J. and Zheng J. (2017) Multiple Consumer-grade Depth Camera Registration Using Everyday Objects. Image and Vision Computing, 62:1-7.

The contributions of the co-authors are as follows:

- Prof. Cai provided the initial project direction and edited the manuscript drafts.
- I wrote the drafts of the manuscript. The manuscript was revised together with Prof. Cham and Prof. Zheng.
- I performed the experiments on the comparison with state-of-the-art methods. Prof. Cai, Prof. Zheng and Prof. Cham provided suggestions on the experiments.

Chapter 5 is published as Deng, T., Cai, J., Cham, T.-J. and Zheng J. (2018) Shading-based Surface Recovery Using Subdivision-based Representation. In Computer Graphics Forum. ACM.

The contributions of the co-authors are as follows:

- Prof. Cai and Prof. Zheng provided the initial project direction and edited the manuscript drafts.
- I wrote the drafts of the manuscript. The manuscript was revised together with Prof. Cham.
- I performed the experiments on the shading recovery and surface refinement. Prof. Cai, Prof. Zheng and Prof. Cham provided suggestions on the experiments.

Chapter 6 is published as Deng, T., Cai, J., Cham, T.-J. and Zheng J. (2018) SubdSH: Subdivision-based Spherical Harmonics Field for Real-time Shading-based Depth Refinement. In Proceedings of IEEE International Conference on Visual Communications and Image Processing. IEEE.

- Prof. Cai and Prof. Zheng provided the initial project direction and edited the manuscript drafts.
- I wrote the drafts of the manuscript. The manuscript was revised together with Prof. Cham.
- I implemented the shading-based shading recovery software on GPU and performed the experiments on the shading recovery and surface refinement. Prof. Cai, Prof. Zheng and Prof. Cham provided suggestions on the experiments.

29-Jan-2019

.....

Date

.....

Deng Teng

Acknowledgements

I thank all who in one way or another contributed in the completion of this thesis.

First, I would like to express my sincere gratitude to my supervisor Prof. Jianfei Cai for his patience, encouragement and continuous support provided throughout my Ph.D study and research. I have been very lucky to have him to be my supervisor and become a researcher under his excellent guidance.

Besides, I would like to express my gratitude to my co-supervisor Prof. Tat-jen Cham, who provides me invaluable supports and advises, and offers me the opportunity to work in 3D reconstruction. I also would like to thank Prof. Jianmin Zheng, who guides me with his profound knowledge and inspires me with a lot of innovative research ideas.

I would like to thank my friends and colleagues in Institute for Media Innovation for their supporting and helping me with all the experiments. I treasure the time we went through to show all the fancy demos and the discussions we had that enlightens me in my research.

Most importantly, I would like to thank my family for their love and kindly support. Especially, I thank my wife, Li Hui, for continued support and encouragement. Your companionship over the years has been invaluable.

Contents

Abstract	i
Acknowledgements	vii
Contents	viii
List of Figures	xi
List of Tables	xiv
1 Introduction	1
1.1 Background	1
1.2 Objective and Research Scope	2
1.3 Summary of Contributions	5
1.4 Organization	6
2 Literature Review	7
2.1 Camera Calibration	7
2.1.1 Pinhole Camera Calibration	7
2.1.2 Depth Camera Calibration	8
2.1.3 Multiple RGB-D Camera Calibration	11
2.2 Depth Image Enhancement	12
2.3 Image-based 3D Reconstruction	13
2.3.1 Multi-view stereo reconstruction	13
2.3.2 Photometric stereo reconstruction	14
2.3.3 Combined MVS and PS methods	15
2.3.4 Surface refinement on RGB-D	16
2.3.5 Image Formation Model	17
3 Registration of Multiple RGB-D Cameras via Local Rigid Transformations	19
3.1 Overview	19
3.2 Proposed Approach	21
3.2.1 Computation of Transformation Coefficients	23
3.3 Results and Discussion	24
3.3.1 Experimental Setup	24
3.3.2 Rendering	26

3.3.3	Evaluation and discussion	26
3.3.4	Additional results	29
3.4	Conclusion	32
4	Multiple Consumer-grade Depth Camera Registration Using Everyday Objects	33
4.1	Overview	33
4.2	Proposed Method	34
4.2.1	Shape Detection and Localization	36
4.2.1.1	Generating Shape Database	36
4.2.1.2	Shape Detection	37
4.2.2	Pose and Depth Distortion Refinement	38
4.2.2.1	Camera Pose Initialization	38
4.2.2.2	Generating Correspondences	39
4.2.2.3	Optimization	39
4.3	Experiments	41
4.3.1	Experimental Setup	41
4.3.1.1	Baseline Methods	41
4.3.1.2	Dataset Construction	42
4.3.1.3	Parameters	43
4.3.2	Experimental Results	43
4.3.3	Discussion on Grid Resolution	44
4.4	Conclusion	45
5	Shading-based Surface Recovery Using Subdivision-based Representation	49
5.1	Overview	49
5.2	Subdivision Representation for Overall Illumination Vectors	50
5.2.1	Smooth Illumination Vector Function	51
5.2.2	Adding Independent Variables	53
5.2.2.1	Modified Subdivision Rules with Additional Independent Vectors	53
5.2.2.2	Update of Subdivision Coefficient Matrices	55
5.3	Subdivision-based MVS Surface Reconstruction	56
5.3.1	Subdivision-based Surface Model	56
5.3.2	Optimization	57
5.3.3	Numerical Computation	59
5.4	Experiments	60
5.4.1	Evaluation on Subdivision-based Overall Illumination Model	60
5.4.2	Evaluation on Surface Reconstruction	65
5.5	Conclusion	66
6	SubdSH: Subdivision-based Spherical Harmonics Field for Real-time Shading-based Depth Refinement	70
6.1	Overview	70
6.2	New Lighting Representation	72
6.2.1	Image Formation	72
6.2.2	SubdSH: Subdivision Representation for SH Illumination Field	73

6.3	Real-time Shading-based Depth Refinement	75
6.3.1	Lighting Estimation	75
6.3.2	Depth Map Refinement	78
6.4	Experimental Results	79
6.4.1	Quantitative Evaluation on Synthetic Data	80
6.4.2	Qualitative Evaluation on Real Data	87
6.4.3	Limitations	87
6.5	Conclusion	87
7	Conclusion and Future Works	91
7.1	Conclusion	91
7.2	Future Directions	92
	List of Publications	94
	 Bibliography	 95

List of Figures

1.1	Three models of consumer-grade depth sensors.	3
2.1	An illustration of computing a 3D point from its corresponding 2D pixel and depth using a pinhole camera model.	9
3.1	Single rigid transformation methods for multiple RGB-D cameras obtained with the optical calibration and with the geometric ICP algorithm generally provide a correct registration only for a certain region, such as the background (green, top row) or the foreground (red, bottom row), but not all the regions. In contrast, proposed method (right) provides a correct registration for the whole captured scene.	20
3.2	Registration of two textured geometries acquired from two RGB-D cameras (Kinects) by a single 3D rigid transformation (e.g., obtained by the popular ICP algorithm) with (a) the internal parameters of the RGB-D cameras provided by the manufacturer, and with (b) the internal parameters computed by Herrera’s calibration toolbox [33]. (c) The proposed 3D registration field with the internal parameters provided by the manufacturer. This comparison shows that, in contrast to single 3D rigid transformation-based methods, our proposed approach provides a visually appealing alignment even when using low accuracy internal (e.g., manufacturer’s) parameters.	22
3.3	Given parameter (u, v, w) for sample point, $\mathbf{x}$, 7 <i>lerp</i> operations are performed to compute the amount of translation at $\mathbf{x}$ from the eight translation coefficients $\mathbf{t}_{ijk}$. The amount of rotation is computed accordingly, by performing 7 <i>slerp</i> operations given the quaternions $\mathbf{r}_{ijk}$	23
3.4	Comparison of the average reprojection error (in pixels) in both color images obtained by a global registration and the proposed method at $3 \times 3 \times 3$ locations uniformly distributed in the scene. For an easier visualization, the (x, y, z) locations are illustrated on a 3D grid.	25
3.5	Novel viewpoints rendered by a virtual camera. The full motion of the virtual camera is available in the accompanying video.	28
3.6	Results obtained by the proposed approach on a dynamic scene.	30
3.7	Results obtained by the proposed approach on a dynamic scene acquired by a moving setup.	30
3.8	Top Left: Representative result with three Kinects via Manu. parameters. Top Right: color coded visualization of the overlap of the three Kinects via Manu. parameters. Bottom Left: Representative result with three Kinects via proposed parameters. Bottom Right: color coded visualization of the overlap of the three Kinects via proposed parameters.	31

4.1	Proposed Framework.	35
4.2	An illustration of different consistency constraints of a scene.	36
4.3	Visual comparisons of different methods on the depth data pre-corrected by [33] on “KinectV1Corner” dataset. Odd rows are the colored geometry with one color per depth camera. Even rows are the color-coded distances.	46
4.4	Visual comparisons of different methods on the depth data pre-corrected by [33] on “KinectV1Wall” dataset. Odd rows are the colored geometry with one color per depth camera. Even rows are the color-coded distances.	47
4.5	Results of the averaged point cloud distance under different grid resolutions.	48
5.1	One iteration of subdivision on both vertex positions and vertex overall illumination vectors.	52
5.2	Loop subdivision. Left: the weights for computing a new edge point in a general case. Middle: the weights for computing a new edge point at boundary. Right: the weights for updating an existing vertex where k is the valence of the vertex and $w = \frac{1}{k} \left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4} \cos \frac{2\pi}{k} \right)^2 \right)$	52
5.3	An illustration on the smoothness constraints of subdivision illumination model with/without adding independent variables for buddha model with “campus4” lighting. The vertices with independent overall illumination vectors are shown in red. The vertices with 1-iteration subdivision smoothness and 2-iteration subdivision smoothness are shown in green and black, respectively.	54
5.4	Examples of colorized light vectors generated from different illumination conditions on the buddha model with “campus4” (left) and “campus32” (right). Self-shadows are more obvious on the mesh illuminated by “campus4” lighting.	62
5.5	A visual comparison between our subdivision illumination methods with and without the proposed adaptive refinement for “buddha” model with “campus4” lighting.	64
5.6	Quantitative results on proposed method with different number of independent overall illumination vectors on “buddha” model with ground-truth normal and “campus4” and “campus32” lighting.	64
5.7	Examples of DTU datasets. Top row: Image from one of the captured views. Bottom row: Rendered ground-truth point cloud.	68
5.8	Examples of captured images for our challenging real-world dataset.	68
5.9	Visual comparisons between [93] and the proposed method on the DTU datasets and our captured dataset. From top to bottom: scan33, scan49, scan74, scan109 and captured datasets.	69
6.1	Shading-based geometry refinement relies on a good approximation of lighting. However, shading cues from challenging lighting conditions (top) are not easily described by a traditional single global lighting model widely used in existing shading-based refinement methods (middle). Our subdivision-based local illumination model can better recover the shading and improve the quality of the refined geometry (bottom).	71
6.2	The image grid can be considered as a quad mesh and the image pyramid corresponds to the inverse operation of generating subdivided meshes from a base mesh.	74

6.3	Catmull-Clark Subdivision Scheme. Left: the mask for computing a new face point. Middle: the mask for computing a new edge point. Right: the mask for computing a vertex point where n_e is the number of adjacent edge points and n_f is the number of adjacent face points, $n = n_e + n_f$, $\beta_1 = \frac{2}{(n_e+n_f)n_e}$ and $\beta_2 = \frac{1}{(n_e+n_f)n_f}$.	74
6.4	Our region-based shading constraint for fast local lighting estimation.	77
6.5	Visual comparison on shading recovery between [91] and our proposed method on <i>Armadillo</i> dataset	82
6.6	Visual comparison on shading recovery between [91] and our proposed method on <i>Roman</i> dataset	83
6.7	Visual comparison on shading recovery between bilinear interpolated local lighting model and proposed method on <i>Statuette</i> .	84
6.8	Visual comparison on depth refinement results between [91] and our proposed method on <i>Hylas</i> dataset. The highlighted regions in (b) and (c) show the wrongly recovered details where convex details are recovered as concave details.	85
6.9	Visual comparison on depth refinement results between [91] and our proposed method on <i>Happy</i> dataset.	86
6.10	Visual comparison on depth refinement results between [91] and our proposed method on NYU dataset. The details on the mattress are better recovered by the proposed method.	88
6.11	Visual comparison on depth refinement results between [91] and our proposed method on NYU dataset. The surface of the cabinet are smooth but the details are well recovered by the proposed method.	89
6.12	Visual comparison on depth refinement results between [91] and our proposed method on our captured dataset. Proposed method handles well on the systematic error from consumer-grade RGB-D cameras.	90

List of Tables

3.1	Calibration residuals by the method of Herrera et al. [33] for two Kinects.	24
4.1	Number of calibration and testing scenes in the constructed datasets. . . .	43
4.2	Results of the averaged point cloud distance (cm) and the number of closest point pairs under different methods. Note that Herrera’s method cannot be applied on Kinect V2 data.	44
5.1	Quantitative results on angular and magnitude errors for overall illumination vector recovery. N_L is the number of independent overall illumination vectors.	63
5.2	Quantitative results on distance errors for surface recovery on synthetic datasets. N_V , N_L and N_P are the number of vertices, independent overall illumination vectors and independent positions, respectively. Acc. is the averaged relative distances (normalized by the diagonal distance of the bounding box) of ground-truth vertices to recovered surfaces.	63
6.1	Averaged time performance on a 640×480 RGB-D image with our parallelized lighting solver implementation.	81
6.2	Quantitative results on mean absolute error (MAE) of recovered shading.	81
6.3	Quantitative results on recovered depth error (in mm).	81

Chapter 1

Introduction

1.1 Background

The process of generating three-dimensional (3D) computer model of a real-world object or scene from two-dimensional images is called *3D Reconstruction*. 3D reconstruction is a core technology and its applications are found in many fields, such as medicine, robotics, film industry, gaming and so on. For instance, the 3D lesion information of the patients improves the accuracy in diagnosis. In robotics, knowing the 3D information of the surrounding environment enables reliable autonomous control of the robots. A natural human computer interaction (HCI) can be achieved by knowing the 3D information of the user. In film industry, obtaining the geometry information enables the modification of the scene properties or a natural and seamless composition of multiple scenes.

On one hand, as humans, we perceive 3D structure of the surrounding world with ease. On the other hand, though extensively studied for about five decades with significant progress, this task remains challenging. Traditional techniques focus more on the passive methods where the shape is obtained from the original appearance of the object and there is no interference with the reconstructed object. These methods are in general very slow and applicable to certain circumstances. In contrast with passive methods, active methods refer to the 3D reconstruction techniques that interfere with the objects. They normally require the acquisition of depth maps, which can be obtained from devices with, for example, structured light patterns or time-of-flight lasers.

Nowadays, thanks to the advent of affordable consumer-grade RGB-D cameras, active methods become more and more popular. RGB-D sensors are a specific type of sensing devices that augment the conventional color information (RGB) with depth (D) information. Microsoft Kinect is probably the most popular one among them. Firstly

introduced in November 2010 as an attempt to broaden the audience for Xbox 360, the first-generation Kinect (Kinect V1) revolutionizes the way people play games and how they entertain. The device features a color camera, a depth sensor and a multi-array microphone, providing full-body 3D motion capture, facial recognition and voice recognition capabilities. With the leading technology on human body language understanding, it enables user to control and interact with their computer/console without the traditional game controller. Compared to its previous RGB-D sensors in the market, which usually cost around \$10,000 each, Kinect only costs around \$200. After the great success of Kinect V1, more consumer-grade depth sensors are released in the market such as Intel RealSense and the second-generation Kinect (Kinect V2), as shown in Figure. 1.1. Lightweight consumer-grade depth sensor product like Structure Sensor, allowing to work with mobile platforms (such as tablets), offers more potential to users to work with 3D data. A lot of research attentions have been drawn to the affordable consumer-grade RGB-D cameras. A number of interesting and exciting applications are developed based on the device, such as 3D object and scene scanning [37], 3D telepresence [50, 60, 61], and real-time novel view synthesis [49]. In the field of 3D reconstruction, many recent techniques have been proposed to obtain real world geometry in real-time with ease [66, 67, 91, 105].

In general, a 3D reconstruction framework based on RGB-D sensors consists of four steps: sensor calibration, sensor registration, initial data fusion and surface refinement. Sensor calibration step refers to the computation of the intrinsic parameters of color and depth cameras. The depth distortion correction parameters are obtained in this step as well. Sensor registration aims to find the proper transformation to warp the captured 3D data into a common coordinate system (e.g. world coordinate system). Depending on the scenario of using RGB-D sensors, sensor registration is slightly different. In a setup of capturing static scenes through a moving RGB-D sensor, sensor registration is to find the pose of each frame. Alternatively, it is to find the extrinsic parameters of each RGB-D sensor when dealing with multiple static RGB-D sensors to capture a dynamic environment. Once the sensor registration is done, initial data fusion is performed to generate a rough shape of the reconstructed object/scene. Many advanced 3D reconstructions contain a final step of surface refinement to improve the 3D reconstruction quality of the initial obtained shapes.

1.2 Objective and Research Scope

Although the consumer-grade RGB-D sensors bring easy and convenient ways for 3D reconstruction, it also brings in new challenges including the following aspects:

(A) Microsoft Kinect V1

(B) Microsoft Kinect V2

(C) Intel Realsense

FIGURE 1.1: Three models of consumer-grade depth sensors.

- Initially designed for gesture recognition, these sensors lack accuracy on the captured geometries. The generated depth maps usually contain depth distortions. The internal computation and processing of the depth information are usually hidden to the end users, making it more difficult to design a proper parametric depth correction model. The depth errors consist of systematic errors, which could be due to the imperfect internal measurements of the RGB-D camera, and high-frequency random noise that may come from the instability of the stereo-matching algorithms.
- Occlusion is another issue in 3D reconstruction when dealing with RGB-D sensors. This can be simply solved by adding more cameras from different viewpoints. Then

the problem become how to calibrate the multi-camera system. Most of the calibration and registration of the RGB-D cameras are performed using checkerboard, i.e., a planar calibration target contains alternating black and white squares. It is widely used in the traditional color camera calibration process, which requires corner detection based on visual features. However, these visual features are normally not visible by the depth cameras, making it impossible to directly perform depth camera calibration and registration. Thus, many existing methods make use of the coupled raw infra-red (IR) images, which are assumed to be at the same viewpoint of the depth camera, to perform indirect calibration and registration of the corresponding depth cameras. Since the process of depth computation from IR images are hidden to users, errors are unavoidable if performing RGB-D calibration and registration in this manner.

- Existing calibration methods of multiple RGB-D cameras are difficult for maintenance. Once any of the cameras moves, the whole process of the calibration needs to be reconducted. The use of planar calibration target also has the problem of high calibration error when captured in a gazing angle. It requires expert knowledge for proper positioning of the checkerboard to have good calibration results.

On the other hand, recent progress on surface refinement shows fine surface details can be recovered by the state-of-the-art 3D reconstruction methods. Specifically, SfS is the mostly adopted technique when an initial shape and its color information are provided. The challenges faced in SfS today are listed below:

- Most of the SfS methods deal with an ill-posed problem, since the color changes in images are the joint effects of environmental lighting, surface material and geometry. Various assumptions are made to solve this problem effectively and efficiently. For example, the estimation of the visibility of the lighting sources is ignore in most of the recent shading-based research for speeding up the whole process.
- It is impossible to accurately model the real-world illumination conditions under arbitrary environment, given the light sources can be with any intensity and at any possible locations.
- The computational costs are high. The per-vertex formulations on the shading constraints in the optimization framework largely increase the costs in terms of both speed and memory. Early methods can take up to tens of gigabyte memory and a few hours for processing a single shape.

In this thesis, we mainly focus on two aspects: systematic noise removal in calibration and registration stages for consumer-grade RGB-D sensor; surface detail recovery from

a rough shape through SfS techniques. If we regard the reconstructed 3D surface as digital signals, the former deals with the error in the low-frequency domain while the latter recovers the high-frequency component. Our goal is to improve existing 3D reconstruction framework by designing and developing effective and efficient algorithms that can reduce the systematic error and recover fine surface details.

1.3 Summary of Contributions

The thesis has made several contributions to the 3D reconstruction related research, including a multi-RGBD camera registration technique through local rigid transforms, a multi-depth camera calibration technique using everyday objects, a shading-based MVS surface refinement framework by subdivision-based overall illumination vectors and a real-time shading-based depth refinement system using local lighting models.

- We propose an approach to register data obtained from a multi-hybrid camera setup that uses a non-rigid alignment strategy making the method more robust and independent of technology and manufacturer. Our approach locally computes the parameters that can accurately register the geometry and color streams and interpolate these values across the entire captured volume. This approach provides a more visually appealing alignment than competing techniques, is easy to perform, and is efficient to use in practice.
- We propose an approach for directly registering depth maps obtained from multiple fixed depth cameras utilizing pre-scanned everyday objects. Particularly, high quality pre-scanned 3D shapes of standard furniture are used as calibration targets. A unified framework is presented that jointly computes the optimal extrinsic calibration and depth correction parameters.
- We introduce subdivision-based representations for both lighting and geometry in shape-from-shading. Our proposed approach uses classic subdivision to define the basic smooth lighting function and surface, and introduces additional independent variables into the subdivision to adaptively model sharp changes of illumination and geometry. Compared to previous works, the new model not only preserves the merits of the per-vertex illumination model, but also greatly reduces the number of variables required in surface recovery and intrinsically regularizes the illumination vectors and surface. A variational surface reconstruction model built upon the subdivision representations for lighting and geometry is developed.
- We extend our novel subdivision-based lighting model for shading-based surface recovery that can be performed on a complete scene where the lighting conditions

can vary across different locations. Our key idea is to estimate a smoothly varying lighting field that is described by a few control points, determined through Catmull-Clark subdivisions over the image grid. The lighting estimation process is accelerated by exploiting parallelization on GPU. As a result, we can perform fast shading-based depth refinement under challenging lighting conditions.

1.4 Organization

The rest of the thesis is organized as follows. In Chapter 2, the existing works on the two major problems are reviewed. In Chapter 3, a new method is proposed on the registration of multiple color and depth streams via local rigid transforms. In Chapter 4, a novel method on multiple depth stream distortion correction and registration is proposed using everyday object as calibration target. In Chapter 5, we propose a novel subdivision-based representations for both lighting and geometry with the improvement on the shading-based surface recovery. In Chapter 6, we extend the idea of subdivision-based representation on lighting and apply it to the existing real-time 3D acquisition system. Finally, we conclude the thesis and discuss future works in Chapter 7.

Chapter 2

Literature Review

In this chapter we review the existing works on handling the two major problems. To begin with, we introduce both the calibration and registration on color and depth cameras that are related to our solution to the first problem. Then we review existing methods on the surface recovery as our second solution for improving the 3D geometry acquisition framework.

2.1 Camera Calibration

Camera calibration estimates the parameters of a lens and image sensor of a camera, including intrinsics, extrinsics and distortion coefficients. These parameters can be used for the lens distortion correction, camera pose calculation and 3-D scene reconstruction. In this thesis, we only focus on the study of pinhole camera calibration since most of the existing consumer-grade RGB-D cameras can be modeled as pinhole camera. Calibration methods of other camera models, such as catadioptric and fisheye cameras [75], are not included.

2.1.1 Pinhole Camera Calibration

Pinhole camera calibration is a well-established task in computer vision, where Zhang's technique [100] is probably the most widely used and has been implemented in Bouget's MATLAB toolbox [10] and OpenCV library [11]. Zhang's approach makes use of a planar checkerboard captured in multiple poses for establishing the correspondences needed to estimate intrinsic, extrinsic and distortion parameters. Beside using checkerboard to obtain correspondences, others make use of circular [62] and point [83] targets for similar purpose.

2.1.2 Depth Camera Calibration

Early works on depth sensor calibration focus more the registration between a laser range finder and a paired color camera. Following previous pinhole camera calibration, a common procedure treats the depth camera and color camera as a camera pair and calibrate their intrinsic parameters and the relative pose [87, 98]. These methods make use of checkerboard patterns and its planar geometry to compute the extrinsics. They may not be optimal since they assume the measurement from depth camera is reliable, which may be true for the expensive 3D laser range finders but not applicable to other depth sensor with noisy data. Meanwhile, the requirement of the manual labeling on the planar checkerboard in the range image makes the methods quite troublesome to use.

In general, a consumer-grade depth camera is modeled by a pinhole camera model. As shown in Fig. 2.1, given a pixel point $m = [u, v]^T$ and depth map D , its 3D position $p^c = [x, y, z]^T$ in the camera coordinates can be found by

$$\begin{aligned} x &= \frac{u - c_x}{f_x} z \\ y &= \frac{v - c_y}{f_y} z \\ z &= D(u, v) \end{aligned} \tag{2.1}$$

where (f_x, f_y) denotes the focal length and (c_x, c_y) denotes the principle point of the depth sensor.

Equation 2.1 is the basic equation to convert a depth map to a 3D point cloud. To further improve the accuracy, image distortion and depth distortion can be taken into account.

In terms of image distortions, the radial and tangential distortion are considered. To simplify distortion correction equations, $m = [u, v]^T$ denotes the old pixel point and $m_{corrected} = [u_{corrected}, v_{corrected}]^T$ denotes the corrected pixel point for each kind of image distortion.

The radial distortion models the “barrel” or “fish-eye” effect of a camera. To correct the radial distortion, the following formula can be used:

$$\begin{aligned} u_{corrected} &= u(1 + k_1 r^2 + k_2^4 + k_3 r^6) \\ v_{corrected} &= v(1 + k_1 r^2 + k_2^4 + k_3 r^6) \\ r &= \sqrt{(u - c_x)^2 + (v - c_y)^2} \end{aligned} \tag{2.2}$$

FIGURE 2.1: An illustration of computing a 3D point from its corresponding 2D pixel and depth using a pinhole camera model.

where k_1 , k_2 and k_3 are the radial distortion parameters.

The tangential distortion happens when camera lens is not perfectly aligned with the image plane. It can be corrected via the formulas:

$$\begin{aligned} u_{corrected} &= u + [2p_1xy + p_2(r^2 + 2x^2)] \\ v_{corrected} &= v + [p_1(r^2 + 2y^2) + 2p_2xy] \\ r &= \sqrt{(u - c_x)^2 + (v - c_y)^2} \end{aligned} \quad (2.3)$$

where p_1 and p_2 are the tangential distortion parameters.

Once the corrected pixel point $m_{corrected}$ is obtained, it can be taken into Equation 2.1 to be converted into a 3D point position.

From Equation 2.2 and 2.3, it is obvious that the correction on image distortion is incapable to tackle the depth distortion on depth map D . Thus, one common strategy is to adopt an additional depth correction function, which is normally modeled as a pixel position related function to the original depth reading.

$$z_{corrected} = f_{u,v}^{z-correct}(z) \quad (2.4)$$

where $f_{u,v}^{z-correct}(\cdot)$ is the depth correction function at pixel point m .

Other previous works focus on the calibration of Time-of-flight (ToF) camera, which simultaneously produces an intensity image and a depth map from the same viewpoint. Such property simplifies calibration process of ToF cameras, since the registration between depth camera and color camera is straight forward. Fuchs and Hirzinger [27] perform the simplification via accurate localization of color discontinuities. Their method has a high number of parameters and requires a robotic arm to know the exact pose of the camera to perform calibration. Lindner and Kolb [56] replace the robotic arm by a high-resolution color camera to determine the camera poses. Lichti [54] calibrate a ToF camera using planar calibration objects, making use of radiometric intensity and range for each pixel. However, like previous ToF camera calibration methods, it is not applicable to other setups with individual color and depth cameras, since it requires color and depth frames are taken from the same camera and identical pose. Zhu et al. [103] propose a method to fuse depth from stereo rig and ToF cameras. They use depth from stereo cameras as ground truth but their depth accuracy depends on the stereo triangulation.

Recent years, due to its wide usage in the research communities, consumer-grade RGB-D camera becomes a main subject in camera calibration field. Many calibration techniques are proposed to estimate the camera parameters and improve its depth accuracy. Zhang and Zhang [97] calibrate Kinect V1 via the co-planar property of points on checkerboard and additional correspondences between depth and color images. But their method does not handle depth distortion. Smisek et al. [80] present a detailed procedure to calibration Kinect V1. They calibrate the pose between RGB and IR cameras and study the shift between IR and depth images, which could be the results of depth calculation process inside Kinect V1. Depth distortion is also handled by using a pixel-wise z-correction technique. But their correction method may suffer from the information loss when quantizing depth values into 11-bit integers. Herrera et al. [33] improves the depth correction accuracy by handling the depth distortion on the raw disparity value of the depth map. However, the official sdk for Kinect V1 does not support retrieving disparity maps. The method only works on specific Kinect V1 driver, which limits its usage. Teichman et al. [84] proposed a innovative method to calibrate single depth sensor via Simultaneous Localization and Mapping (SLAM) without using a checkerboard. The underlying principle is to use the close depth, which is more reliable, to correct the obvious distortion on the faraway depth. Though the method is convenient to use, the depth accuracy may not be guaranteed since the method relies on the camera pose estimation from SLAM. Similarly, Zhou et al. [101] propose a more complicated method, which simultaneously optimizes camera poses and depth distortions. But the depth accuracy is depended on the scene captured. It could be problematic when using the parameters from positions where no content is observed.

2.1.3 Multiple RGB-D Camera Calibration

Aforementioned methods are mainly on the calibration of single RGB-D sensor. A straightforward adoption of the single RGB-D sensor calibration methods to a multiple RGB-D camera setup is not guaranteed an accurate result. The optical method is the most commonly used method, due to its convenience on directly processing color/grayscale images. Many existing consumer-grade RGB-D sensors provide the IR image that captured from its depth camera, enabling registration of multiple sensors through the traditional checkerboard-based methods [80, 100]. The optical approach has been followed by many researchers. For example, Maimone and Fuchs [60] adopt the method to register their multi-Kinect setup of a 3D telepresence system. However, ignoring the depth information from multiple depth sensors during the calibration usually results in a poor registration of the 3D geometry of the capture scene.

On the other hand, geometric method is to calibrate multiple RGB-D cameras through geometric information. Iterative Closest Point (ICP) [8, 70] is a commonly used technique that directly register the point clouds from different depth cameras. However, this method highly relies on the initial positions of the point clouds and requires shape with rich geometric features. In case some correspondences of 3D points are known, one can directly estimate the rigid transformation [35]. This geometric approach has been followed by Izadi et al. [37] in the context of a moving Kinect. But it requires a small movement between consecutive frames and it usually fails when it scans an object with less geometric feature or depth noise becomes dominant in the pose estimation process, similar to other geometry-based calibration method.

More recently, Avetisyan et al. [3] proposed a method that makes use of an optical tracking system, providing physical positions of checkerboard corners. It constructs a 3D look up table that maps depth values to a more reliable measurement from the optical system. Similarly, Beck et al. [7] compute a single calibration and registration volume for each camera using correspondences obtained from a tracked checkerboard based on an optical tracking system. Though these methods greatly improve the quality of multiple RGB-D camera registration, their limitation are obvious, since they both require an highly accurate and expensive optical tracking system. The calibration accuracy relies on the optical tracking system and it may not be tedious for normal users to perform the same calibration process.

To sum up, when dealing with multiple depth cameras, a 3D point registration is required to convert points from each camera coordinate to a common world coordinate. Given $\mathcal{N}$ depth cameras and their pose $(\mathbf{R}, \mathbf{T})$, for i -th camera, a 3D point $p_i^c \in P_i^c$ in camera coordinates can be converted to a point $p_i^w \in P_i^w$ into world coordinates via formula:

$$p_i^w = R_i p_i^c + t_i \quad (2.5)$$

where $R_i \in \mathbf{R}$ denote the rotation part and $t_i \in \mathbf{T}$ the translation part of the pose for camera i .

2.2 Depth Image Enhancement

Various works are proposed for depth image enhancement. Model based method [24, 39, 89] impose assumptions on the geometry and fitting planes or quadratic surfaces to each region of the images. But these images are usually taken under urban environment. Other methods, however, make use of local geometric conditions. Some assume that edges in

both color and depth channels likely coincide. Yang et al. [48] upsample a low-resolution depth image using an aligned color image using a joint bilateral upsampling (JBU) filter. Variants of joint-bilateral or multilateral filtering for depth upsampling [21, 57, 71, 72] can achieve real-time performance. There are also global optimization methods that take care of the connections of local geometric conditions. Diebel and Thrun [20] computed the upsampled depth through an optimization of a Markov-Random Field (MRF), using the texture derivatives of the color image as the smoothness term. Park et al. [69] extend the framework by depth upsampling to color image resolution by enforcing consistent discontinuities, with additional regularization. Sparseness caused by the local smoothness of the measure shapes is used as global information. Li et al. [53] learn a mapping function from a training data set and enhance the depth map via sparse coding algorithm. Kiechle et al. [46] adopt a bimodal co-sparse analysis model that captures the interdependency of registered intensity and depth information.

The limitation of the depth image enhancement is that the refined depths are usually not metrically accurate and their recovered geometric details were usually incorrect.

Recent attempts have been made to enhance range images through deep networks [82], but continue to encounter inaccuracies in recovering geometric details.

2.3 Image-based 3D Reconstruction

Modeling the 3D shape of real objects or scenes from photographs is a classic problem that extensively studied over the past three decades. Its goal can be described as "*given a set of photographs of an object or a scene, estimate the most likely 3D shape that explains those photographs, under the assumptions of known materials, viewpoints, and lighting conditions*" [29]. The challenge comes from the assumption that materials, viewpoints and lighting conditions are known. But it is generally not true, leading to an ill-posed problem since there exists multiple combination of geometry, materials, viewpoints and lighting conditions that produce the same images.

Two methods, Multi-view stereo (MVS) and photometric stereo (PS), have been quite successful to perform the image-based 3D reconstruction. These two methods are reviewed in this section.

2.3.1 Multi-view stereo reconstruction

Multi-view stereo (MVS) reconstruction refers to a group of techniques that extract geometry using stereo correspondence from two or more images from different viewpoints.

Seitz et al. [76] have an excellent survey on MVS algorithms. According to the survey, existing MVS methods are classified into four categories. The first category is the volumetric methods [79, 81, 88], where final shape is determined by the occupancy of the voxels that refined from the initial volume of the whole scene. The second category is the surface evaluation methods [28, 85, 96]. These methods evolve a rough initial shape (e.g. sphere) to a fine shape based on shape consistency or other constraint between multi-view images. The third category is the depth fusion methods [13, 63], in which depth map are computed under each viewpoint and merged later to generate the 3D model. The forth category is the region growing methods [30, 31]. Seed surfaces are initially found by triangulation of the matched 3D surface points and then expanded gradually. The final 3D shape is calculated from all the expanded seed surfaces.

The limitations of the MVS methods are mainly from the key steps of finding point correspondences. Even with the increase of image resolution and quality of digital cameras, it is still hard to find point correspondences between all points in multi-view images, especially for the texture-less surface. Absence of high-frequency shape details is the common issue when adopting solely the MVS algorithms for 3D reconstruction.

2.3.2 Photometric stereo reconstruction

Photometric stereo (PS) methods estimate 3D shape through recovering surface normals when observing the shape under different lighting conditions. Herzmann et al. [34] estimate object geometry and its reflectance property based on examples placed in the same scene with known geometry and similar reflectance. Basri et al. [6] propose to perform PS algorithms under complex unknown illumination. Wu et al. formulate the PS problem using Markov Random Field and solve it through optimization. Wu et al. [92] adopt a low rank matrix completion method to recover surface normal as well as shadow and specular information. To overcome the limitation of assuming simple reflectance property, attempts [2] are made to recover surface normal, shape and spatially varying bidirectional reflectance distribution function (BRDF), which is used to describe complex reflectance properties. Recent research attempt to loosen the assumptions and targets at unknown illuminations [25, 43].

While PS methods in general require multiple images of the objects under various illumination, a special case where the data is a single image is known as shape from shading (SfS). Oxholm et al. [68] estimate shape from a single image under natural lighting conditions, but the lighting was obtained from a calibration object. Yu et al. [95] recovered relative albedo from noisy depth information and separated shading from albedo, followed by enhancing surface normals from the estimated shading. Barron and Malik [5]

used a learned mixture model of shapes and illumination to recover intrinsic properties of a scene. To model natural lighting as a combination of global and local components, Han et al. [32] used a quadratic function and per-vertex scalars respectively. Depth information was then recovered from the refined surface normals using the estimated lighting via the algorithm by Nehab et al. [64].

The drawbacks of PS methods are obvious. It is hard to perform 3D reconstruction on moving or deformable objects since objects are still with different lighting conditions. Meanwhile, compared to MVS methods that allow to reconstruct the full 3D shape, PS methods usually recovers a partial geometry of the object from a fixed viewpoint.

2.3.3 Combined MVS and PS methods

It can be seen from the above review that MVS methods can generate a rough 3D shape but in general loss of high-frequency details. On the other hand, however, PS methods can recover shape details but are usually from a fixed viewpoint. The complementary nature of the MVS and PS methods suggests improvements can be done from combining the two methods. Several works using this strategy are proposed for 3D reconstruction.

In fact, this hybrid technique has been long studied from previous research. Fua et al. [26] propose to minimize the error function composed of shading, surface smoothness and stereo constraint with smooth varying albedo properties. Samaras et al. [74] perform surface reconstruction with piece-wise constant albedo map, which is estimated from stereo fitting. The shape and illumination are then estimated from a deformable SfS algorithm. Jin et al. [41, 42] combine different MVS and PS constraints into their frameworks to estimate Lambertian surface through surface evolution and variational minimization methods. Joshi et al. [44] recover a depth map and corresponding normals from images captured from different views under different illumination directions. Esteban et al. [23] present a method to obtain a closed surface reconstruction from multiple viewpoints, under varying lighting conditions. Although good quality of the reconstructed shape can be obtained, [44] and [23] require a large number of the images from different viewpoint and various illuminations that are configured in a experimental setup (e.g. single light point source in dark room). Yoshiyasu and Yamazaki [94] propose a topology adaptive evolution method based on shading and silhouette cues from photographs captured by a moving camera with flash. The method is still limited by the requirement of a dark room, so it cannot handle general cases. Wu et al. [90] propose to use spherical harmonic function to model general illumination and refine 3D shape using shading cues from an initial shape obtained from MVS method. The drawbacks of this approach are the high computational costs of both visibility computation and the vertex

refinement through nonlinear energy optimization. Zhou et al. [102] integrate photometric stereo information from multiple viewpoints into a single framework. It requires a complex acquisition system that controls both light and camera positions. Xu et al. [93] propose to formulate the shading-based surface recovery problem using a new concept called overall illumination vector, which considers the lighting and albedo. A total variation (TV) based constraint is adopted in their optimization framework. Langguth et al. [52] propose an integrated optimization scheme that effectively combines the energies of MVS and SfS based on image gradients. Kim et al. [47] perform surface refinement using online images that taken under different illuminations, which are modeled by a per-image lighting model.

2.3.4 Surface refinement on RGB-D

Recent years, more research interests are in 3D reconstruction using RGB-D data. A typical setup is to reconstruct the environment by a moving RGB-D sensor using implicit surface representations. Signed distance fields (SDF) are the most commonly used model, where 3D surfaces are represented by voxels. Many successful RGB-D surface reconstruction algorithms are based on this technique [37, 65]. Choi et al. [15] propose a robust optimization and use only geometry for high-quality pose alignment. Dai et al. [16] show large-scale scenes can be reconstructed in real time through their proposed global optimization technique. Although SDF-based fusion methods greatly reduce depth noise, they rely on observation of the surfaces from different viewpoint. Errors in the pose estimation or large baselines usually lead to noticeable reconstruction error.

Many works are done to improve the 3D shape obtained by the RGB-D sensor using shading cues. Wu et al. [91] reformulated the inverse rendering optimization into a highly parallelized algorithm that can be efficiently solved on GPU, and their system was able to achieve real-time performance on shading-based depth recovery. Or-El et al. [67] improved this system by handling spatially varying albedo through Retinex theory [51]. They also added independent, spatially changing variables in the lighting function to account for local lighting variations. They further improved their framework to recover shapes with specularities [66], but it required the light source position to be known. Recently, shading-based refinement was also applied to 3D models represented by volumetric signed distance fields. Zollhöfer et al. [105] extended Wu's framework [91] to perform shading-based refinement on the volumetric representation after fusing multiple RGB-D images. Maier et al. [59] followed the same pipeline while jointly optimizing camera poses and used a trilinear-based spatial varying SH to model lighting. Unlike previous systems [91, 105], this optimization was done without GPU acceleration.

2.3.5 Image Formation Model

We review the most related image formation models from previous works. Assuming the light sources are infinitely distant and all objects in the scene are non-emitters, the image irradiance can be defined by [45]

$$B(v, \omega_o) = \int_{\Omega} L(\omega_i) V(v, \omega_i) \rho(\omega_i, \omega_o) \max(\omega_i \cdot \mathbf{n}(v), 0) d\omega_i \quad (2.6)$$

where $B(x, \omega_o)$ is the reflected radiance, v is the spatial location, $\mathbf{n}$ is the surface normal. ω_i and ω_o are the incident and outgoing angles, respectively. Ω denotes the domain of all the possible directions. $L(\omega_i)$ represents the incident lighting. $V(v, \omega_i)$ is a binary visibility function and $\rho(\omega_i, \omega_o)$ is the bidirectional reflectance distribution function (BRDF) of the surface.

If we further assumes Lambertian surface, where $\rho(\omega_i, \omega_o)$ is a constant and depending on the materials of the surface, the image formation can be:

$$I_o(v) = \int_{\Omega(v)} \rho(v) I_i(v, \omega) \max(\omega \cdot \mathbf{n}(v), 0) V(v, \omega) d\omega \quad (2.7)$$

where $\rho(v)$ is the albedo of v , ω is the incident direction, $I_i(v, \omega)$ is the incident radiance along ω , $\mathbf{n}(v)$ is the unit surface normal at v , $\Omega(v)$ represents a hemisphere of incident directions at v , and $V(v, \omega)$ stands for a binary visibility function of vertex v to direction ω .

The Lambertian reflectance model is widely adopted by many SfS methods. In [90], Spherical Harmonics (SH) are used to perform lighting and geometry estimation. Due to the orthogonality of the SH basis, the image irradiance equation can be simplified as

$$I_o(v) = \int_{\Omega(v)} L_a(\omega_i) T(v, \omega_i) d\omega_i = \sum_{k=1}^{n^2} l_k t_k \quad (2.8)$$

where $T(v, \omega_i) = V(v, \omega_i) \max(\omega_i \cdot \mathbf{n}(v), 0)$ and $L_a(\omega_i) = \rho L(\omega_i)$ is the incident illumination scaled by the albedo ρ . l_k and t_k are the SH coefficients of lighting L_a and visibility T . $n - 1$ is the order of the SH.

In many real-time SfS algorithms [91, 105], to speed up the process and take into account the scene may consist of multiple materials, spatial variant per-vertex albedo is added and visibility is neglected and thus

$$I_o(v) = \rho(v) \sum_{k=1}^{n^2} l_k H_k(\mathbf{n}(v)) \quad (2.9)$$

where $H_k(\mathbf{n}(v))$ are the SH basis function taking a unit surface normal $\mathbf{n}(v)$ as input. In practice, second or third order SH is adopted for efficiency thus only the low frequency part of the lighting is estimated. This may be true for distant lighting sources, but it cannot handle more general cases under challenging unknown lighting.

Besides SH, a concept called *overall illumination vectors* is proposed to simplify the task [93]. Specifically, let $\Omega'(v)$ denote the subset of Ω for which $\omega \cdot \mathbf{n}(v) > 0$ and $V(v, \omega) = 1$. The overall illumination vector at v is defined as

$$L(v) = \int_{\Omega'(v)} \rho(v) I_i(v, \omega) d\omega \quad (2.10)$$

and hence Eq.2.7 is simplified to

$$I_o(v) = L(v) \cdot \mathbf{n}(v). \quad (2.11)$$

It is worth pointing out that $L(v)$ represents an overall effect of all incident lights at v . Also $L(v)$ implicitly includes visibility and albedo. Thus, shading cues depend only on the surface normal and $L(v)$, which significantly simplifies the process of surface reconstruction.

However, the per-vertex overall illumination vector representation introduces many additional variables since each vertex v of the reconstructed mesh has a vector $L(v)$ and the number of the vertices is usually very large. Moreover, the discrete characteristic of the per-vertex overall illumination vectors makes them lack intrinsic coherence or constraints across the neighboring vertices.

Chapter 3

Registration of Multiple RGB-D Cameras via Local Rigid Transformations

3.1 Overview

Despite that textured geometry obtained by each RGB-D camera is visually appealing, even state-of-the-art methods have difficulties in correctly combining the textured geometries obtained by several RGB-D cameras via a rigid transformation. Our experiments support this observation and show that even with a very carefully executed registration procedure using state-of-the-art methods, the geometry and color do not align consistently everywhere. As illustrated in Figure 3.1, it is common that the alignment might be satisfying in some areas but not in others.

The reasons are complex. First, multiple RGB-D camera setups require not only geometry alignment but also combined geometry and texture alignment which makes the task challenging. Second, current RGB-D cameras consist of a color camera, an IR camera and an IR projector. They require a careful calibration but calibrating such a complex device is a complicated task and thus highly error prone. Small calibration errors can result in erroneous 3D data that, in turn, cannot be rigidly aligned to 3D data acquired by another device, as illustrated in Figure 3.1. While these calibration errors can be visually acceptable when only one RGB-D camera is used, they become immediately apparent in multiple RGB-D camera setups resulting in severe visual artifacts. Third, the proprietary implementation of these cameras contains additional processing, especially on the depth stream, that are unknown to the user.

FIGURE 3.1: Single rigid transformation methods for multiple RGB-D cameras obtained with the optical calibration and with the geometric ICP algorithm generally provide a correct registration only for a certain region, such as the background (green, top row) or the foreground (red, bottom row), but not all the regions. In contrast, proposed method (right) provides a correct registration for the whole captured scene.

In this chapter, we propose an approach to register data obtained from a multi-hybrid camera setup that uses a non-rigid alignment strategy making the method more robust and independent of technology and manufacturer. Our approach locally computes the parameters that can accurately register the geometry and color streams and interpolate these values across the entire captured volume. This approach provides a more visually appealing alignment than competing techniques, as illustrated in Figure 3.2, is easy to perform, and efficient to use in practice.

3.2 Proposed Approach

While in theory, a single global rigid transformation for each device should be sufficient to register multiple data streams to the same coordinate system, in practice, even using state-of-the-art calibration methods, the alignment from this global registration is inaccurate. Therefore, rather than assuming a global rigid transformation, we model the registration in a spatially variant way. Therefore, the transformation to transform a point x from camera A into camera B is a function for x , i.e.,

$$x' = R_x x + T_x \quad (3.1)$$

In this work, we present the field of translations, $\mathbf{T}_x$, and the field of rotations, $\mathbf{B}_x$, by first placing a rectangular regular grid of $n \times m \times o$ vertices within camera A.

We represent the field of translations, $\mathbf{T}_x$, and the field of rotations, $\mathbf{R}_x$, by first placing a rectangular regular grid of $n \times m \times o$ vertices within camera A. Then, for each vertex $\mathbf{v}_{ijk}$, where $i = 1, \dots, n$, $j = 1, \dots, m$, and $k = 1, \dots, o$, we estimate a rotation coefficient $\mathbf{r}_{ijk}$ and a translation coefficient $\mathbf{t}_{ijk}$, which are tuned to accurately transform $\mathbf{v}_{ijk}$ into camera B. The computation of these transformation coefficients is conducted as a preprocess and is discussed in more detail in Section 3.2.1. Then, a linear basis is imposed on the grid to reconstruct the smooth rotation and translation field, both of which are $C^{(0)}$ continuous.

Due to the linear basis, the evaluation of the amount of rotation $\mathbf{r} := \mathbf{R}_x$ and translation $\mathbf{t} := \mathbf{T}_x$ boils down to first identifying the voxel which contains $\mathbf{x}$. Then, a parameter (u, v, w) is computed from the eight corresponding voxel vertices $\mathbf{v}_{ijk}$. Note that u , v , and w vary between 0 and 1. By referring to Figure 3.3, $\mathbf{r}$ and $\mathbf{t}$ are computed by first interpolating the respective transformation coefficients pair-wise along u . The resulting values are then interpolated along v . And finally, the resulting two values are interpolated along w , yielding $\mathbf{r}$ and $\mathbf{t}$, respectively.

FIGURE 3.2: Registration of two textured geometries acquired from two RGB-D cameras (Kinects) by a single 3D rigid transformation (e.g., obtained by the popular ICP algorithm) with (a) the internal parameters of the RGB-D cameras provided by the manufacturer, and with (b) the internal parameters computed by Herrera's calibration toolbox [33]. (c) The proposed 3D registration field with the internal parameters provided by the manufacturer. This comparison shows that, in contrast to single 3D rigid transformation-based methods, our proposed approach provides a visually appealing alignment even when using low accuracy internal (e.g., manufacturer's) parameters.

FIGURE 3.3: Given parameter (u, v, w) for sample point, $\mathbf{x}$, 7 *lerp* operations are performed to compute the amount of translation at $\mathbf{x}$ from the eight translation coefficients $\mathbf{t}_{ijk}$. The amount of rotation is computed accordingly, by performing 7 *slerp* operations given the quaternions $\mathbf{r}_{ijk}$.

While trilinear interpolation is used to interpolate translation quantities, rotations have to be interpolated over the sphere to achieve constant-speed motion. This is achieved by the *slerp* operation, defined as

$$\text{slerp}(t, \mathbf{r}_0, \mathbf{r}_1) = \frac{\sin((1-t)\alpha)}{\sin(\alpha)} \mathbf{r}_0 + \frac{\sin(t\alpha)}{\sin(\alpha)} \mathbf{r}_1 \text{ with } t \in [0, 1] \quad (3.2)$$

which linearly interpolates between the two quaternions $\mathbf{r}_0$ and $\mathbf{r}_1$, respectively, and where $\cos(\alpha) = \mathbf{r}_0 \cdot \mathbf{r}_1$. More information on the *slerp* operation is provided in [77].

The next section discusses the computation of the two 3D lattices of rotations and translations, and proposes an algorithm which guarantees that our method is always at least as accurate and in most cases significantly more accurate in registering the geometry of multi device data as methods that use a global rigid transformation.

3.2.1 Computation of Transformation Coefficients

The choice of the coefficients $\mathbf{r}_{ijk}$ and $\mathbf{t}_{ijk}$ significantly affects the quality of alignment of the camera data, hence, accurate computation of these coefficients is key. We first sample the scene with a checkerboard that is observed by both cameras.

A 3D location is determined for each checkerboard corner which is local to the respective camera, i.e., for each checkerboard corner we have a pair of points $(\mathbf{x}, \mathbf{x}')$, where $\mathbf{x}$ is the corner as seen in camera A , and $\mathbf{x}'$ is its corresponding corner as seen in camera B . We propose an adaptive scheme to compute $\mathbf{r}_{ijk}$ and $\mathbf{t}_{ijk}$ as follows:

1. Consider a set of nested cubic regions with increasing sizes centered all around $\mathbf{v}_{ijk}$.

2. For each region l , construct a set $\mathcal{C}^l$ containing checkerboard corner correspondences, $(\mathbf{x}, \mathbf{x}')$, where $\mathbf{x}$ is located within the respective region.
3. For each of these regions, estimate the rotation $\mathbf{r}_{ijk}^l$ and translation $\mathbf{t}_{ijk}^l$ using the rigid transformation estimation method in [35].
4. Pick $\mathbf{r}_{ijk}^l$ and $\mathbf{t}_{ijk}^l$ that produces the smallest reprojection error. The reprojection errors are evaluated on the pairs of corners in the respective set $\mathcal{C}^l$.

By construction, the reprojection error obtained with our method is never higher than the reprojection error obtained using the transformation computed from all the samples. In fact, in the result section we demonstrate that our reprojection error is frequently much lower.

3.3 Results and Discussion

3.3.1 Experimental Setup

To compare our approach with existing methods, we build a setup composed of two rigidly attached Kinects. To reduce the interference due to the IR projectors, we attach a small vibrator to each Kinect [12].

As discussed in previous sections, each RGB-D camera must be internally calibrated to obtain appropriate textured geometry, and then the transformations between the RGB-D cameras are needed to register the textured geometries in a common coordinate system. For the internal calibration, we use the parameters provided by the manufacturer as well as obtained from Herrera’s state-of-the-art toolbox [33]. The toolbox provides very low reprojection/residual errors in both the color images and depth space for more details), namely, less than one pixel and less than one kdu (Kinect disparity units, see [33]). For completeness, the exact values are available in Table 3.1.

To compute the transformation between the RGB-D cameras, we compare against both optical and geometric approaches. For the optical approach, we apply the popular Camera Calibration Toolbox for Matlab [10], in a similar way to Maimone and Fuchs [60]. To avoid biasing the registration for a certain part of the scene, we acquire data of a

TABLE 3.1: Calibration residuals by the method of Herrera et al. [33] for two Kinects.

	Color	Depth
Kinect1	0.260 ± 0.004 px	0.7749 ± 0.0006 kdu
Kinect2	0.637 ± 0.014 px	0.7306 ± 0.0006 kdu

FIGURE 3.4: Comparison of the average reprojection error (in pixels) in both color images obtained by a global registration and the proposed method at $3 \times 3 \times 3$ locations uniformly distributed in the scene. For an easier visualization, the (x, y, z) locations are illustrated on a 3D grid.

checkerboard visible in both cameras at many different orientations and locations in the whole volume of interest. For the geometric approach, we apply ICP [8] and acquire several 3D geometries by moving the setup at different poses. Since ICP depends on the initial solution, we give the best possible initial alignment manually and then run the ICP method available in Point Cloud Library.

To apply our approach, we use the same checkerboard data as acquired for the optical approach and run the algorithm (Section 3.2.1). The grid covers the volume of interest and its size is set to $3 \times 3 \times 3$ meters with $20 \times 20 \times 20$ regular voxels.

3.3.2 Rendering

To render the scene from a virtual camera position, the depth map of each RGB-D camera is tessellated and vertices of the triangles are transformed by either a single rigid transformation, or by the proposed approach. These triangle meshes are textured accordingly. The textured geometries of each RGB-D camera are projected into the virtual view and the color of the pixel in the final view is the average of the projected textures. If a pixel is visible by only one camera, we directly use the color observed by that camera. This type of blending is chosen intentionally to better visualize the quality of the alignment.

The proposed method is computationally inexpensive and runs in real-time on the GPU, taking about 18ms per frame for two RGB-D cameras at full resolution (i.e., 640×480 depth maps). The execution time linearly scales with the number of devices by pairwise registration. An important observation is that the execution time does not depend on neither the volume size nor the grid resolution: First, the regularity of the grid allows for efficient lookup of the coefficients for a given query point, where a lookup is performed in constant time. Second, the time to compute the transformation of a query point does not depend on the coefficient values (eight) but the number of coefficients, and thus runs in constant time as well.

3.3.3 Evaluation and discussion

In the following, the global 3D rigid transformation is computed by ICP only, due to space limitations and as it yields comparable quality to the optical approach. A visual comparison between a global transformation and the proposed rigid transformation field is shown in Figure 3.2. It can be seen that our approach leads to visually appealing images in which the textures are correctly aligned.

To quantitatively measure the results, we consider the reprojection error in the images. We prefer this measurement over the 3D registration error simply because the reprojection reflects the visual quality.

We apply the following procedure to compute the reprojection error. The N corners of a checkerboard are observed by each of the M RGB-D cameras. Their 2D coordinates are obtained by corner extraction in each color images and their 3D coordinates are provided by the depth sensor of each RGB-D camera. These 3D corners can be reprojected in the image plane of any RGB-D camera. In noise-free cases, these reprojected points lie at the exact same location as the extracted corner in the image. However because of noise, these points do not perfectly coincide and the distance between them is not zero. We compute the registration error in the image space as the mean distance between each corresponding corners. The reprojection error from the k -th RGB-D camera to the j -th RGB-D camera is:

$$\frac{\sum_{i=1}^N \|p_i^j - p_i^{k \rightarrow j}\|}{N} \quad (3.3)$$

where p_i^j is the 2D location of the i -th corner point extracted in the color camera of the j -th RGB-D camera and $p_i^{k \rightarrow j}$ represents p_i^k which is reprojected in the color image of the j -th RGB-D camera. If $k = j$, the error is zero, so we do not consider self-reprojection.

Summing over each pair of RGB-D cameras, the overall reprojection error is computed as:

$$\frac{\sum_{j=1}^M \sum_{k=1, k \neq j}^M \sum_{i=1}^N \|p_i^j - p_i^{k \rightarrow j}\|}{M(M-1)N} \quad (3.4)$$

A single rigid transformation obtains an average reprojection error of 2.5 px (with a std. dev. of 0.8). In contrast, our method provides an average of 0.8 px (with a std. dev. of 0.2), which is an improvement of about 68%.

We further analyze the error distribution within the volume of interest. We compute the average reprojection error (see Eq. 3.4) of the checkerboard corners located inside a local volume of size $20 \times 20 \times 20$ cm and centered at each grid vertex, using both the proposed method and a global rigid transformation. Results are illustrated in Figure 3.4. For a better visibility, we display the error computed on a uniform $3 \times 3 \times 3$ subsampling of the grid. It shows that our reprojection always outperforms the global registration approach. It also illustrates the fact that the error obtained by the global registration greatly varies with respect to the location in the scene, in agreement with the results in Figure 3.1.

In addition, we study the reprojection accuracy with respect to the grid resolution. The average reprojection error is 1.3 px for $5 \times 5 \times 5$ grid, 1.1 px for $10 \times 10 \times 10$ grid, 0.8 px

FIGURE 3.5: Novel viewpoints rendered by a virtual camera. The full motion of the virtual camera is available in the accompanying video.

for $20 \times 20 \times 20$ grid and 0.8 px for $40 \times 40 \times 40$ grid. This indicates that the accuracy increases with the grid resolution until converging to 0.8 px. For all the results shown in this chapter, we use a $20 \times 20 \times 20$ grid (same accuracy as a $40 \times 40 \times 40$ grid but with less coefficients in memory).

3.3.4 Additional results

In the context of free-viewpoint video, the pose (location and orientation) of the virtual camera can be controlled interactively by the user or via a head/eye tracking system, for example in combination with autostereo displays [60].

To study the quality of the results in this context, we move the virtual camera between the physical cameras and render the scene at these new viewpoints. Still images of the novel viewpoints are available in Figure 3.5 and we invite the readers to view the full camera motion in the accompanying video. It can be seen that our method can provide visually appealing results for a range of virtual camera’s pose. Note that our goal is not to compete with systems dedicated to free-viewpoint videos such as [49, 104], but to provide a visually appealing alignment that can be then combined with free-viewpoint video techniques such as silhouette refinement and advanced blending (see [104]).

Our approach can also be applied to dynamic scenes. To illustrate this, we acquire sequences with moving objects. We process each frame sequentially by directly applying the method described in Section 3.2. Figure 3.6 illustrates a representative result obtained with a person walking. This example demonstrates that our method can correctly deal with dynamic scenes, in contrast to volumetric fusion-based methods [37]. In an additional experiment, we capture a dynamic scene with an acquisition setup installed on a mobile platform. Results are available in Figure 3.7 and show that our approach can also be apply in a mobile acquisition context.

Our approach can deal with more than two Kinects. In practice, we applied our approach pairwise with respect to a given Kinect considered the reference device. We used a single calibration volume and grid and ran the method of Section 3.2.1 for each pair of devices, i.e., the coefficients of the grid vertices encode the rotation and translation associated to the current pair. Figure 3.8 shows a representative result obtained by a setup composed of three Kinects.

FIGURE 3.6: Results obtained by the proposed approach on a dynamic scene.

FIGURE 3.7: Results obtained by the proposed approach on a dynamic scene acquired by a moving setup.

FIGURE 3.8: Top Left: Representative result with three Kinects via Manu. parameters. Top Right: color coded visualization of the overlap of the three Kinects via Manu. parameters. Bottom Left: Representative result with three Kinects via proposed parameters. Bottom Right: color coded visualization of the overlap of the three Kinects via proposed parameters.

3.4 Conclusion

Registration of multiple RGB-D camera setups is a notoriously sensitive and delicate procedure despite the variety of available methods. The registration between RGB-D cameras which typically consists of a (single) rigid transformation often fails in practice to faithfully align all the depth and color streams consistently everywhere inside the capture volume, as observed in many experiments. We address this issue by proposing a practical and general approach to register the depth and color streams of multi RGB-D camera setups. Our approach estimates a smooth field of rigid transformations between RGB-D cameras with $C(0)$ continuity on a regular grid within the captured volume. In terms of both visual quality and measurable reprojection errors, we demonstrate that our method provides better results than state-of-the-art methods.

Chapter 4

Multiple Consumer-grade Depth Camera Registration Using Everyday Objects

4.1 Overview

Consumer-grade depth cameras have two major limitations: they only have a restricted field of view and can only recover a depth field rather than true 3D data. One approach to overcome these problems would be to move the camera around [37] under the assumption of a static scene; but if real-time wide-coverage 3D data is needed, such as in 3D telepresence [60] and real-time novel view synthesis [49], then multiple depth cameras are required. The challenge then becomes registering and calibrating multiple depth cameras, given those known problems that such cameras have systematic distortions in depth measurements and are in general very noisy.

Extrinsic calibration may be conducted on a multi-camera rig, to determine the relative poses of the depth cameras with respect to a common reference frame. One approach is to fall back on classical image-based (i.e. optical) calibration techniques utilizing a checkerboard and color or raw IR imagery. Doing so would effectively calibrate the cameras through passive stereo triangulation, while completely ignoring the depth sensing mechanisms. However, 3D data subsequently obtained from the cameras will register poorly as the systematic depth distortions are not accounted for. An alternative approach is to compute the rigid transform that will geometrically align 3D data separately sensed in the multiple cameras. Although doing so does not correct for the depth distortions, this optimization procedure will typically distribute 3D registration errors more evenly.

While depth distortion correction methods have been proposed for a single depth camera [3, 33, 40, 60, 80], combining such methods with extrinsic calibration in a sequential or piecemeal manner remains unsatisfactory and will still lead to significant misalignment, as the depth correction for individual devices tends to be sensitive to errors. Instead, we propose a *unified* framework in which extrinsic calibration and depth distortion correction are determined together.

For short experiments, an initial one-off calibration may be satisfactory. However, if accurate registration must be retained for an extended period of time, then in practice recalibration must be conducted at multiple intervals, due to mechanical displacement of the cameras (intentional, accidental, or environmental temperature variations, etc.) or simply sensor drift. This can pose a major practical problem if the calibration process is tedious, such as requiring tens or even hundreds of checkerboard observations for each recalibration. We thus seek to find a more practical method of maintaining registration, such as replacing checkerboards with everyday objects found in the scene, such as regular furniture.

Therefore, in this chapter, we propose a new approach for registering depth maps obtained from multiple fixed depth cameras using a non-rigid alignment strategy. High quality pre-scanned 3D shapes of standard furniture are used as calibration targets, rather than checkerboards; these scans may be obtained offline using a moving depth camera together with a structure from motion algorithm such as KinectFusion [37]. The key idea of the proposed approach is to utilize the high-quality geometry of the pre-scanned shape as both a reference and a conduit for registering the dynamic 3D data obtained from the multiple static depth cameras.

4.2 Proposed Method

In this section, we describe the details of the proposed method. Fig. 4.1 shows the steps in our proposed framework. During the pre-processing stage, we generate a pre-scanned shape database and obtain backgrounds for each depth camera. When performing the registration, we capture scenes with the pre-scanned shapes at different locations. Pre-scanned shape detection and localization are performed to find poses of the detected shapes. Correspondences are found using the poses of pre-scanned shapes and camera poses and depth correction function are computed through an optimization framework.

FIGURE 4.1: Proposed Framework.

FIGURE 4.2: An illustration of different consistency constraints of a scene.

4.2.1 Shape Detection and Localization

There are two steps before we can use a 3D shape in our proposed calibration framework. First, we need to obtain the geometry of the 3D shape. Then, the pose of the shape should be detected given a depth map. Their details are in the following.

4.2.1.1 Generating Shape Database

Common 3D shapes in the indoor environment, e.g. chairs, are scanned via the interactive scanning technique - KinectFusion [37]. The targeting objects are placed in a controlled environment so that most parts are visible. To ensure the quality of 3D shapes, we only use the depth within the most reliable working range of a consumer-grade depth sensor. A small amount of manually editing is performed on the 3D shapes to separate the targeting shapes from its surrounding environment. An example of a pre-scanned 3D shape is shown in Fig. 4.2. A more precise 3D shape, which could be either provided by the manufacturer or downloaded from the Internet, is expected to improve our calibration results. However, our experiments show that a pre-scanned object using a KinectFusion is already good enough for calibrating consumer-grade depth sensors.

4.2.1.2 Shape Detection

We adopt the approach of Drost et al.[22] for recognising and locating the 6DoF pose of a pre-scanned shape. This approach performs a Generalised Hough Transform [4] via Point-Pair Features (PPFs). PPF is a four-dimensional asymmetric descriptor computed from the relative positions and the normals of a pair of local points m_1 and m_2 :

$$F(m_1, m_2) = (\|d\|_2, \angle(n_1, d), \angle(n_2, d), \angle(n_1, n_2)) \quad (4.1)$$

where $d = m_1 - m_2$, n_1 and n_2 are the normals of m_1 and m_2 , respectively. For more details, please refer to [22]. A global description for each 3D shape in the database is generated by discretising PPFs with similar values and sorting them in search data structures, similar to the procedures in [73].

For a depth camera i , given a depth map $\mathcal{D}_i$, we first generate a point cloud $\mathcal{P}_i$ and compute a normal map $\mathcal{N}_i$. In order to speed up and improve the accuracy of the shape recognition, we segment each frame into foreground and background pixels using background subtraction. We assume the targeting shapes are inside the foreground and thus we limited our search within the foreground pixels. We treat the objects that are not likely moving within a session as backgrounds. For example, floors, walls and some heavy furniture are seen as background since they are rarely being moved in an indoor scene. Thus, we capture the background by removing all the movable objects in the scene. In practice, we take 20 frames on the background scene and use the averaged frame $\mathcal{D}_i^0$ as the background frame. To detect the foreground pixels on the depth map, we simple check the difference on the depth between the captured frame and the background frame, i.e.

$$|\mathcal{D}_i(u, v) - \mathcal{D}_i^0(u, v)| > Threshold. \quad (4.2)$$

We generate the foreground cloud $\mathcal{P}_i^f$ and extract its PPFs. By matching with the pre-computed PPFs of pre-scanned shapes, we are able to find the initial shape poses $\hat{T}_{i,s}$ w.r.t. camera i . We then use ICP to refine shape poses $\hat{T}_{i,s}$ to get more accurate poses $T_{i,s}$. The pseudo-code of the shape detection and pose estimation algorithm is provided in Algorithm 1.

Algorithm 1 Shape Detection Algorithm For Each Depth Camera

```

for  $i \leftarrow$  each depth frame  $\mathcal{D}_i$  do
  Generate foreground cloud  $\mathcal{P}_i^f$ 
  Initialize processing cloud  $\mathcal{P}_{tmp} \leftarrow \mathcal{P}_i^f$ 

  for each pre-scanned shape  $s$  do
    // find initial shape pose through PPF framework
    Generate  $ppf(\mathcal{P}_{tmp})$ 
    Retrieve shape cloud  $\mathcal{P}_s$  and  $ppf(\mathcal{P}_s)$  from database
    Initialize shape pose  $\hat{T}_{i,s}$  from  $ppf(\mathcal{P}_{tmp})$  and  $ppf(\mathcal{P}_s)$ 

    // refine shape pose through ICP
    Generate shape cloud  $\mathcal{P}_s \leftarrow \hat{T}_{i,s}\mathcal{P}_s^{db}$ 
     $\Delta T \leftarrow$  ICP between  $\mathcal{P}_{tmp}$  and  $\mathcal{P}_s$ 
    Refine object pose  $T_{i,s} \leftarrow (\Delta T)\hat{T}_{i,s}$ 
    // verify shape pose
    if 1/3 visible part of refined shape cloud  $T_{i,s}\mathcal{P}_s^{db}$  overlaps with  $\mathcal{P}_{tmp}$  then
      Save shape pose  $T_{i,s}$ 
    end if

    // remove detected points
    Update  $\mathcal{P}_{tmp} \leftarrow (\mathcal{P}_{tmp} - T_{i,s}\mathcal{P}_s^{db})$ 
  end for
end for

```

4.2.2 Pose and Depth Distortion Refinement

As illustrated in Fig. 4.2, given a scene, we divide it into pre-scanned objects and other overlapped regions in the context of object detection results.

4.2.2.1 Camera Pose Initialization

With the shape detection results from Section 4.2.1, we can easily compute the camera poses. Given camera i , camera j and shape s , we can get shape poses $T_{i,s}$ and $T_{j,s}$ in camera i and camera j , respectively. Thus, the relative pose of camera j with respect to camera i can be computed as

$$T_{i,j} = T_{i,s}(T_{j,s})^{-1}. \quad (4.3)$$

After selecting reference camera, we can obtain the initial camera pose T_i^0 by averaging poses over different shapes and frames. Note that for simplicity we use T_i , T_j and T_s to represent the poses of camera i , camera j and object s , respectively, w.r.t. the world coordinates.

4.2.2.2 Generating Correspondences

After obtaining the initial poses for cameras and pre-scanned shapes, two types of correspondences are extracted to serve as the inputs for our optimization. Let C_i denote the mapping function between the original depth values and the values after the depth correction for camera i . In order to compute C_i , we extract dense correspondences from the scene cloud $\mathcal{P}_i$ and the shape cloud $\mathcal{P}_s$ using the initial camera pose T_i^0 and the object pose $T_{i,s}$. Given the initial poses for depth cameras and 3D shapes, we establish two types of correspondences: shape-to-camera correspondences, denoted by $\mathcal{K}^{shape}$, and camera-to-camera correspondences, denoted by $\mathcal{K}^{cam}$.

We denote points $p \in \mathcal{P}_i$, $q \in \mathcal{P}_j$ and $o \in \mathcal{P}_s$ as a point in the scene cloud of camera i , a point in the scene cloud of camera j and a point in the object cloud of shape s , respectively. The object-to-camera correspondences $\mathcal{K}_{i,s}^{shape}$ are obtained by find the nearest neighbor between the object cloud $\mathcal{P}_s$ and the scene cloud $\mathcal{P}_i$ after warping to the world coordinates. Likewise, the camera-to-camera correspondence $\mathcal{K}_{i,j}^{cam}$ are obtained also by performing nearest neighbor search between the scene clouds $\mathcal{P}_i$ and $\mathcal{P}_j$ in the world coordinates. Note that any pair of two corresponding points with large distance or large normal difference is considered as outliers and thus removed from the correspondence set.

4.2.2.3 Optimization

We obtain our refined poses and depth correction through an optimization framework. To compute $\mathcal{T} = \{T_i\}$ and the depth correction function $\mathcal{C} = \{C_i\}$, we minimize the following objective function:

$$E(\mathcal{T}, \mathcal{C}) = E^{shape} + \lambda E^{cam} + \lambda_{fid} E^{fid} + \lambda_{lap} E^{lap}, \quad (4.4)$$

where the first, second, third and fourth terms are the shape-to-camera term, the camera-to-camera term, the fidelity term and the smoothness term, respectively, and λ , λ_{fid} and λ_{lap} are trade-off parameters. The shape-to-camera term E^{shape} evaluates the alignment between observed shape and pre-scanned shape. We want to correct depth to preserve the shape of the high quality pre-scanned shape from the database. Thus, we define the term as

$$\begin{aligned}
E^{shape}(\mathcal{T}, \mathcal{C}) &= \sum_{i,s} \sum_{(p,o) \in \mathcal{K}_{i,s}^{shape}} ((p' - o')^\top n'_p)^2, \\
p' &= T_i C_i(p), \\
o' &= T_s o.
\end{aligned} \tag{4.5}$$

The camera-to-camera term E^{cam} evaluates the alignment between observed data from different cameras. We aim to reduce the distances between overlapping regions from different cameras by minimizing

$$\begin{aligned}
E^{cam}(\mathcal{T}, \mathcal{C}) &= \sum_{i,j} \sum_{(p,q) \in \mathcal{K}_{i,j}^{cam}} ((p' - q')^\top n'_p)^2, \\
p' &= T_i C_i(p), \\
q' &= T_j C_j(q).
\end{aligned} \tag{4.6}$$

Here p' , q' and o' are the transformed points to world coordinates for p , q and o , respectively. n'_p is the normal of p in world coordinates. Essentially, (4.5) and (4.6) minimize the point-to-plane distances to enforce the merging of two nearby corresponding surfaces.

In (4.4), we also introduce a fidelity term E^{fid} to avoid large deformation and a Laplacian term E^{lap} to impose smoothness on the depth correction function, i.e.

$$\begin{aligned}
E^{fid}(\mathcal{C}) &= \sum_{v \in \mathcal{V}} \|\Delta C(v)\|^2, \\
E^{lap}(\mathcal{C}) &= \sum_{v \in \mathcal{V}} \left\| \Delta C(v) - \frac{\sum_{u \in \mathcal{N}_v} \Delta C(u)}{M} \right\|^2, \\
\Delta C(v) &= C(v) - C^0(v),
\end{aligned} \tag{4.7}$$

where C^0 is the original depth mapping without any depth correction and thus $\Delta C(v)$ can be treated as a correction vector for vertex v . $\mathcal{N}_v$ is the neighborhood of v and M denotes the number of neighboring vertices of $v \in \mathcal{V}$.

To reduce the computation complexity, we adopt a trilinear interpolation scheme for the depth correction function C to reduce the number of parameters in the framework. Given any point p , we compute its depth correction by

$$C(p) = \sum_k \psi_k(p) C(v_k), \tag{4.8}$$

where $v_k \in \mathcal{V}$ is a depth correction sample vertex over a regular 3D lattice $\mathcal{V}$ and $\psi_k(p)$ are the trilinear interpolation coefficients.

For solving the optimization problem, we minimize $E(\mathcal{T}, \mathcal{C})$ using the Gauss-Newton method. Let θ be the vector that includes all the parameters. At iteration k , it is updated as

$$\theta^k = \theta^{k-1} + \Delta\theta. \quad (4.9)$$

$\Delta\theta$ can be computed by solving the following linear system:

$$\mathbf{J}^\top \mathbf{J} \Delta\theta = -\mathbf{J}^\top \mathbf{r}, \quad (4.10)$$

where $\mathbf{r} = \mathbf{r}(\theta)$ is the residual vector and $\mathbf{J}$ is its Jacobian of θ .

4.3 Experiments

In this section, we conduct various experiments to evaluate the effectiveness of the proposed calibration framework.

4.3.1 Experimental Setup

For simplicity, we conduct experiments on a setup of two depth sensors with one everyday object - chair, although our framework supports more than two depth sensors and more than one objects. We adopt the first method in our experiments. Also, we test on both Kinect V1 and Kinect V2 sensors to show that our framework is general for different types of depth sensors.

4.3.1.1 Baseline Methods

We compare our method with three available baseline methods for multi-depth sensor registration: optical method, geometrical method and Deng's method [18]. Both the optical and the geometrical methods generate a single rigid transform between the two depth cameras.

- Optical method: We use the implementation of Zhang's method [100] in OpenCV Library to extract checkerboard correspondences from color images, based on

which we are able to generate camera poses with a reprojection error around 1.0 pixel.

- Geometrical method: We compute the 3D positions of checkerboard correspondences and directly calculate the camera poses using the 3D correspondences.
- Deng’s method [18]: It generates a set of local rigid transforms at the grid vertices. The resolution of the local extrinsic grid is set as $12 \times 12 \times 12$. Deng’s method achieves state-of-the-art performance for multi-depth camera registration.

We also adopt Herrera’s method [33] as a pre-processing option to correct the depths of each individual depth camera, which achieves state-of-the-art performance for single Kinect V1 depth correction. We use their toolbox to jointly calibrate Kinect V1 and an external color camera with a resolution of 1600×1200 . We are able to generate results with a reprojection error less than 0.9 pixel and depth error less than 0.8 kdu (Kinect Disparity Unit, see [33]). Note that Herrera’s method can only be used in Kinect V1 data and cannot be applied on Kinect V2 data since Kinect V2 does not provide disparity maps that is required by Herrera’s method.

4.3.1.2 Dataset Construction

To the best of our knowledge, there is no benchmark dataset available for calibrating multi-depth camera systems. Thus, we construct a few datasets using different depth sensors with calibration targets placed at different locations.

For Kinect V1, both disparity maps and depth maps with a resolution of 640×480 are acquired, since Herrera’s method [33] for single Kinect V1 calibration requires disparity maps. For Kinect V2, depth maps are acquired at a resolution of 512×424 . Although our method does not require color images, we acquire them at a resolution of 1280×960 as the input for the baseline methods to find checkerboard correspondence. To avoid the interference issue with two Kinect V1, we separately capture the static scenes for each device. For our method, we also capture the background scene, which is used to perform foreground segmentation.

Table 4.1 shows the three datasets we have captured. Two datasets “KinectV1Corner” and “KinectV1Wall” are captured by two rigidly fixed Microsoft Kinect V1 with in-between angles of around 90 and zero degrees, respectively. “KinectV2Wall” is taken by two rigidly fixed Microsoft Kinect V2 with zero degree angle. Each dataset consists of three types of captured frames: shape observation frames for calibration, checkerboard observation frames for calibration, testing frames. The shape observation frames are used

TABLE 4.1: Number of calibration and testing scenes in the constructed datasets.

Dataset	Calibration (Shape)	Calibration (Checkerboard)	Testing
KinectV1Corner	8	8	6
KinectV1Wall	10	20	12
KinectV2Wall	20	20	8

as the inputs of our proposed method for calibration, and the checkerboard observation frames are the inputs of the baseline methods for calibration, since all the baseline methods rely on checkerboard corners for registering depth sensors. All the performance evaluations are performed on the test frames.

4.3.1.3 Parameters

For our method, we also use a $12 \times 12 \times 12$ depth correction grid in a $4m \times 4m \times 4m$ space, the same as that in Deng’s method [18]. We empirically set the parameters $\lambda = 1$, $\lambda_{fid} = 0.02$ and $\lambda_{lap} = 0.01$.

4.3.2 Experimental Results

To evaluate the registration performance, we use the metric of point cloud distance, which is the average distance between closest points of the two point clouds. Such metric measures how close the two registered point clouds are. A smaller average distance indicates a better data registration. Specifically, the average point cloud distance is defined as

$$Distance = \frac{\sum_{(p,q) \in K} \|p - q\|_2}{N_K} \quad (4.11)$$

where p and q are two corresponding points from the two point clouds, K denotes the set of closest point pairs for the scene and N_K is the number of closest point pairs. In our experiment, we treat closest point pairs with a distance larger than 5 cm as outliers and exclude them from the evaluation.

Table 4.2 shows the quantitative results of different methods. It can be seen that our proposed method outperforms all the baseline methods in terms of the average point cloud distance with similar number of closest point pairs. Applying Herrera’s method [33] to pre-correct depth data in general helps on reducing the average point cloud distance, while our method can further reduce the distance based on the pre-corrected depth.

TABLE 4.2: Results of the averaged point cloud distance (cm) and the number of closest point pairs under different methods. Note that Herrera’s method cannot be applied on Kinect V2 data.

Method	KinectV1Corner		KinectV1Wall		KinectV2Wall	
	Dist.	# of pairs	Dist.	# of pairs	Dist.	# of pairs
Raw Depth + Optical	1.801	176.7k	1.576	223.8k	1.161	188.0k
Raw Depth + Geometrical	1.295	200.7k	1.585	229.0k	1.092	187.5k
Raw Depth + Deng [18]	1.491	211.0k	1.237	233.2k	1.081	180.1k
Raw Depth + Proposed	1.079	208.2k	1.090	231.8k	0.965	190.1k
Herrera [33] + Optical	1.972	189.3k	1.712	215.2k	-	-
Herrera [33] + Geometrical	1.211	208.8k	1.387	227.4k	-	-
Herrera [33] + Deng [18]	1.379	212.9k	1.214	230.9k	-	-
Herrera [33] + Proposed	0.903	209.3k	1.065	233.7k	-	-

We also provide the visual comparisons in Figure 4.3 and Figure 4.4, which shows both the geometry and the color-coded distances rendered at a virtual view. We can observe that our proposed method has in general closer distances over most of the regions, especially for the floors and walls. However, the performances of the other methods are limited by the checkerboard positions, thus resulting in a larger distance for floors and walls. In particular, “KinectV1Wall” dataset is captured by a relatively easy setup with two depth sensors pointing to a similar direction. The geometrical method on the pre-corrected depth with geometrical extrinsic calibration actually has a good performance. The visual result is further improved by using local extrinsic calibrations [18]. However, these checkerboard-based baseline methods have limitation when registering geometries, since checkerboard only provides sparse point observations which may be greatly affected by depth noise. Our proposed method takes into account the surface distance which is usually more stable and we impose the consistency constraints between the pre-scan object and the observed geometry.

On the other hand, “KinectV1Corner” dataset is captured by a setup with two depth cameras facing almost 90 degrees. It greatly constrains the observation angle of a checkerboard, which may cause problems in corner extractions and increase the possibility of getting invalid depth for some depth sensors. In our method, though observed from different angles, using a pre-scanned shape can still provide reliable correspondences. Therefore, our results are much better than all the baseline methods for “KinectV1Corner” dataset.

4.3.3 Discussion on Grid Resolution

We also investigate the performance effect of different grid resolutions under our proposed framework using “KinectV1Corner” and “KinectV1Wall” datasets. Fig. 4.5 shows

the results with both raw depth and pre-corrected depth using Herrera’s method. It can be seen that the $12 \times 12 \times 12$ resolution has the best performance. Further increasing the resolution makes the average point cloud distance larger. This is mainly because the number of correspondences in each grid cell becomes less when the grid resolution becomes higher, where the depth noise becomes easier to affect the depth correction of grid vertices.

4.4 Conclusion

In this chapter, we have developed a multi-depth sensor registration method using easy-to-obtain everyday objects. Particularly, we use a common chair as an example and pre-scan it. Then, we apply state-of-the-art method to detect and locate the chair to generate correspondences between the pre-scan and the observation. We have proposed an optimization framework that reserves the geometry of the pre-scanned shape and minimizes the distance between the overlapped region of two depth cameras. Experimental results show that our method significantly outperforms state-of-the-art methods in terms of the average point cloud distance as well as achieving much better visual quality. Our method will be very useful for multi-depth sensor systems since it enables the capability of multi-depth sensor online calibration.

FIGURE 4.3: Visual comparisons of different methods on the depth data pre-corrected by [33] on “KinectV1Corner” dataset. Odd rows are the colored geometry with one color per depth camera. Even rows are the color-coded distances.

FIGURE 4.4: Visual comparisons of different methods on the depth data pre-corrected by [33] on “KinectV1Wall” dataset. Odd rows are the colored geometry with one color per depth camera. Even rows are the color-coded distances.

(A) “KinectV1Corner” dataset

(B) “KinectV1Wall” dataset

FIGURE 4.5: Results of the averaged point cloud distance under different grid resolutions.

Chapter 5

Shading-based Surface Recovery Using Subdivision-based Representation

5.1 Overview

In chapter 3 and chapter 4, two methods for correcting the systematic depth error from the calibration and registration of the depth capturing devices are introduced. However, these methods can only help register point clouds captured by multiple depth cameras but cannot solve the problem of recovering surface details of the captured objects or scenes. In this chapter, we exploit shape-from-shading (SfS) for surface detail recovery. We start from an initial shape that obtained from Multi-view stereo methods and adopt our proposed methods on refining the surface details. SfS [25, 36, 43, 99] and MVS [76], are two popular techniques for reconstruction. MVS and SfS are often combined to achieve good surface reconstruction. Basically, these methods make use of the relations among 3D shapes, surface reflectance and lighting conditions to reconstruct geometry and extract surface detail. Thus, the surface reconstruction often involves estimation of underlying unknown lighting, surface visibility and material property of the object, which is usually very difficult.

While many of previous studies have assumptions on the illumination such as distant light sources and low order spherical harmonic (SH) lighting, new approaches have been proposed for handling arbitrary illumination and albedo [5, 47, 105]. Recently, Xu et al. [93] introduced a concept called “overall illumination vector” defined at each vertex of the reconstructed mesh. The overall illumination vector models the joint effect of illumination, visibility and albedo. For reconstructing Lambertian surfaces, recovering

the overall illumination vectors is sufficient. The overall illumination vectors have many advantages. For example, they are flexible in modeling the effects of complicated general illumination. They avoid estimation of the visibility function and surface albedo. However, the use of overall illumination vectors requires a great number of additional variables in surface reconstruction, which limits the number of vertices to be used for the reconstructed surface and thus affects the accuracy of the reconstruction. In addition, the discrete nature of the overall illumination vectors at vertices makes these vectors lack intrinsic coherence, which does not reflect the reality in general.

This chapter presents a solution to these problems. The key ideas behind the solution are in three aspects. First, instead of considering the overall illumination vectors only at the vertices of the surface mesh, we define these vectors as a vector-valued function defined over the surface. That is, when we reconstruct the surface of a 3D object from multi-view images, the overall illumination vectors can be viewed as an attribute of the reconstructed surface, which models the overall illumination effect including the underlying lighting, self-shadowing, occlusion and albedo. The per-vertex overall illumination vectors can be viewed as the samples of this function at vertices.

Second, how do we model or represent this vector-valued function? Considering the complexity of the reconstructed shape and piecewise smoothness of overall illumination [93], subdivision schemes is a good choice since they can model smooth surfaces with arbitrary topology. We propose to adapt Loop subdivision [58] to represent this function and the reconstructed surface as well. The advantage of the subdivision-based representation is that it can use a small number of vectors to define the function and the smoothness ensured by the subdivision scheme automatically regularizes the illumination vectors.

Third, note that Loop subdivision produces surfaces that are smooth everywhere. In many application scenarios, however, sharp illumination changes or sharp geometry features are often observed in overall smooth illumination or surfaces. Using smooth functions or shapes to model sharp features is not very effective or economic, so we introduce a specific design to allow the inclusion of sudden changes in overall smooth functions.

5.2 Subdivision Representation for Overall Illumination Vectors

To overcome the abovementioned problems, we propose to model the overall illumination vectors as a vector-valued function defined over the mesh and view per-vertex overall illumination vectors as samples of the vector function at the vertices of the mesh. Note

that subdivision surfaces have the advantages of defining smooth, arbitrary topological shapes using a rather coarse control mesh. Our idea is to use a subdivision scheme to model the vector function. Specifically, we adopt Loop subdivision to model the base function over a triangular mesh and propose a mechanism called “adding independent variables” to model sharp features. The details are elaborated in the following two subsections. The method will also be adapted for modeling the reconstructed surface in the next section. It is also worth pointing out that other advanced subdivision methods [9, 55, 86] can be adapted to our framework as well. We choose Loop subdivision for its simplicity.

5.2.1 Smooth Illumination Vector Function

We start with a coarse mesh $\mathcal{M}^0$ called the base mesh and each vertex v^0 of $\mathcal{M}^0$ has an overall illumination vector $L(v^0)$. After performing h iterations of Loop subdivision on both $\mathcal{M}^0$ and $L(v^0)$, $\mathcal{M}^0$ becomes $\mathcal{M}^h$ whose vertices are denoted by v^h and $L(v^0)$ becomes $L(v^h)$.

Fig. 5.1 illustrates one iteration of Loop subdivision [58] from the l th level to the $(l+1)$ th level. For three vertices v_p^l , v_q^l and v_r^l at the l th level, we denote by v_p^{l+1} , v_q^{l+1} and v_r^{l+1} the corresponding vertices at the $(l+1)$ th level. By adding three new edge vertices v_s^{l+1} , v_t^{l+1} , v_u^{l+1} , the triangle (v_p^l, v_q^l, v_r^l) at l th level is subdivided into four sub-triangles at $(l+1)$ th level, $(v_p^{l+1}, v_s^{l+1}, v_t^{l+1})$, $(v_s^{l+1}, v_q^{l+1}, v_u^{l+1})$, $(v_r^{l+1}, v_t^{l+1}, v_u^{l+1})$ and $(v_s^{l+1}, v_u^{l+1}, v_t^{l+1})$. Geometrically, the masks for computing new positions or vectors are given in Fig. 5.2.

Let $L(V^h)$ denote the vector consisting of all the overall illumination vectors $L(v_i^h)$, and $L(V^0)$ denote the vector consisting of all the overall illumination vectors $L(v_i^0)$. $L(V^h)$ can be expressed as a linear combination of $L(v_i^0)$:

$$L(V^h) = \Psi(V^0, V^h)L(V^0) \quad (5.1)$$

where $\Psi(V^0, V^h)$ is the accumulated subdivision coefficient matrix from V^0 to V^h , and it can be computed recursively by

$$\Psi(V^0, V^{l+1}) = \Psi(V^l, V^{l+1})\Psi(V^0, V^l), \quad l = 1, 2, \dots, h-1 \quad (5.2)$$

where $\Psi(V^l, V^{l+1})$ is the subdivision coefficient matrix from level l to level $l+1$.

In this way, the overall illumination vectors for $\mathcal{M}^h$ can be determined by those for $\mathcal{M}^0$ and they are implicitly correlated by the subdivision rules. Also the number of vertices of $\mathcal{M}^0$ is roughly $1/4^h$ of that of $\mathcal{M}^h$. The subdivision based representation provides a very compact representation for the overall illumination vectors.

FIGURE 5.1: One iteration of subdivision on both vertex positions and vertex overall illumination vectors.

FIGURE 5.2: Loop subdivision. Left: the weights for computing a new edge point in a general case. Middle: the weights for computing a new edge point at boundary. Right: the weights for updating an existing vertex where k is the valence of the vertex and $w = \frac{1}{k} \left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4} \cos \frac{2\pi}{k} \right)^2 \right)$.

5.2.2 Adding Independent Variables

Due to visibility change and other factors, the illumination may vary sharply in some regions (see Fig. 5.3). However, the overall illumination vectors represented by Loop subdivision are smooth everywhere. To model sharp features, we propose an adaptive refinement method to amend the classic subdivision scheme, aiming to not only preserve the advantages of compact representation and smooth representation with the subdivision, but also accurately model sharp illumination change. Our idea is to convert some generated overall illumination vectors at vertices near the regions with sharp illumination change to independent variables. Specifically, given initial independent illumination vector set $\mathbf{L}^0 = \{L(v^0)\}$, we find a new independent illumination vector set $\mathbf{L}' = \mathbf{L}^0 \cup \mathbf{L}_s^1 \cup \mathbf{L}_s^2 \cup \dots \cup \mathbf{L}_s^h$, where $\mathbf{L}_s^l$ is the set of selected overall illumination vectors $L(v^l)$ on subdivision mesh $\mathcal{M}^l$, $0 < l \leq h$, which are converted to be independent variables.

We perform the refinement in a recursive way from level 0 to level h . Let $L'(V^s)$ denote the vector consisting of all the independent overall illumination vectors $L(v_i^s)$. At each level l , we will first update the existing subdivision rules and identify the additional independent overall illumination vector set $\mathbf{L}_s^{l+1}$. Afterwards, we obtain a new variable set $\mathbf{L}^{l+1} = \mathbf{L}_s^{l+1} \cup \mathbf{L}^l$ and its corresponding accumulated coefficient matrix $\Psi'(V^s, V^{l+1})$ from V^s to V^{l+1} . Therefore, a two step scheme is performed at each level for the refinement.

5.2.2.1 Modified Subdivision Rules with Additional Independent Vectors

We modify the subdivision rules based on the observed intensities $I(v)$, since lighting and visibility changes directly affect the observed intensities. Let m_{ij} indicate the similarity of two vertices v_i and v_j . When two adjacent vertices have similar observed intensities, they are tagged as similar vertices and $m_{ij} = 1$; otherwise $m_{ij} = 0$. For simplicity, we use a threshold ε to determine whether the observed intensities are similar:

$$m_{ij} = \begin{cases} 1 & \|I(v_i) - I(v_j)\| < \varepsilon \\ 0 & \text{otherwise.} \end{cases} \quad (5.3)$$

There are two steps in Loop subdivision, in which we perturb existing vertices (vertex points) and insert new vertices (edge points).

Vertex points. We modify the rule for perturbing existing vertices by updating the valance $k(v_i) = \sum_{v_j \in NN(v_i)} m_{ij}$ where $NN(v_i)$ denotes the neighbourhood of v_i on

(A) Ground-truth Intensity

(B) Proposed w/o adding independent vectors

(C) Proposed with adding independent vectors

FIGURE 5.3: An illustration on the smoothness constraints of subdivision illumination model with/without adding independent variables for buddha model with “campus4” lighting. The vertices with independent overall illumination vectors are shown in red. The vertices with 1-iteration subdivision smoothness and 2-iteration subdivision smoothness are shown in green and black, respectively.

mesh $\mathcal{M}^l$. The weights are computed from k using the standard formulation in Loop subdivision. For $k(v_i) = 0$, the vertex v_i will be assigned to itself, e.g. $v_i^{l+1} = v_i^l$.

Edge points. We perform the additional lighting variable selection in each edge point generation step. For an ordinary subdivision operation that an inserting point is derived from the four neighbouring vertices, as shown in Fig. 5.2, we check the similarity of the observed intensities $I(v)$ between inserting point v and its neighbour $v_{nn} \in \{v_{a_1}, v_{a_2}, v_{b_1}, v_{b_2}\}$.

If $I(v)$ is similar to all the observed intensity $I(v_{nn})$ for neighbouring vertices v_{nn} , we keep the original weight and approximate $L(v^{l+1})$ using Loop subdivision.

If $I(v_{a_1})$ and $I(v_{a_2})$ are close to $I(v)$ while only one of $I(v_{b_1})$ or $I(v_{b_2})$ is close to $I(v)$, we change ordinary Loop subdivision scheme to a boundary case to compute $L(v^{l+1})$.

All other cases are classified as a complex illumination region. For these regions, we add a new lighting variable $L(v_i^{l+1})$ at the vertex v_i into the set $\mathbf{L}_s^{l+1}$.

For a boundary case subdivision operation, as shown in Figure 2, we refine the overall illumination vectors in a similar manner. Since only two neighbouring vertices $v_{nn} \in \{v_{a_1}, v_{a_2}\}$ are related in this case, we evaluate whether their observed intensities are close to $I(v)$ by thresholding using Eq. 5.3. If $I(v_{nn})$ is similar to both $I(v_{a_1})$ and $I(v_{a_2})$, we keep the original subdivision coefficients. Otherwise, we add a new lighting variable $L(v_i^{l+1})$ into $\mathbf{L}_s^{l+1}$.

5.2.2.2 Update of Subdivision Coefficient Matrices

Once we obtain the modified rules and the additional independent lighting variable set $\mathbf{L}_s^{l+1}$ at level $l + 1$, we update the total independent overall illumination vector set $\mathbf{L}^{l+1}$ and subdivision coefficient matrix $\Psi(V^l, V^{l+1})$ accordingly. For the modified rules, we can change their corresponding weights in $\Psi(V^l, V^{l+1})$ directly. Then, we get the updated coefficient matrix $\tilde{\Psi}(V^s, V^{l+1})$ through a similar recursive manner as in Eq. 5.2 without considering the independent vectors, thus

$$\tilde{\Psi}(V^s, V^{l+1}) = \Psi(V^l, V^{l+1})\Psi'(V^s, V^l). \quad (5.4)$$

For a new independent vector at vertex v_i^{l+1} , we add the overall illumination variable $L(v_i^{l+1})$ into $L'(V^s)$ and expand the matrix $\tilde{\Psi}(V^s, V^{l+1})$ by one column and modify its

corresponding i -th row, so that $L(V^{l+1})$ can be expressed as

$$\begin{aligned}
 L(V^{l+1}) &= \Psi'(V^s, V^{l+1})L'(V^s) \\
 &= \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} & 0 \\ w_{21} & w_{22} & \dots & w_{2n} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ w_{i-1,1} & w_{i-1,2} & \dots & w_{i-1,n} & 0 \\ 0 & 0 & \dots & 0 & 1 \\ w_{i+1,1} & w_{i+1,2} & \dots & w_{i+1,n} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ w_{d1} & w_{d2} & \dots & w_{dn} & 0 \end{bmatrix} \begin{bmatrix} L(v_1^s) \\ L(v_2^s) \\ \vdots \\ L(v_n^s) \\ L(v_i^{l+1}) \end{bmatrix} \quad (5.5)
 \end{aligned}$$

where $L(v^s) \in \mathbf{L}^l$ are the lighting variables from previous subdivision level l .

We get the final $\Psi'(V^s, V^{l+1})$ and $L'(V^s)$ after adding all the independent vectors, after which we can proceed to the next subdivision level. In this way, we can have sufficient independent variables to model piecewise smooth overall illumination field.

5.3 Subdivision-based MVS Surface Reconstruction

We now consider the problem of reconstructing the surface from multiple images taken from different viewpoints under general, unknown illumination. Similar to the frameworks of [90] and [93], we assume that an initial MVS mesh has been generated by an existing MVS method and the mapping between the mesh and the multiview images has been established. What remains is to use the shading cues to adjust the positions of vertices of the mesh to recover high-frequency surface details.

5.3.1 Subdivision-based Surface Model

In both [90] and [93], the positions of all the vertices are the variables in the optimization process. This results in high memory cost during the optimization and prevents the methods from processing larger meshes. On the other hand, we notice that the geometry of a smooth region on the surface can be represented by the positions of a few control vertices using a subdivision scheme. Hence, we also use our adaptive subdivision-based approach introduced in Section 5.2 for the positions of vertices.

Let $P(V^h)$ denote the vector consisting of all the vertex positions $P(v_i^h)$, and $P(V^0)$ denote the vector consisting of all the vertex positions $P(v_i^0)$. $P(V^h)$ can be expressed

as a linear combination of $P(v_i^0)$

$$P(V^h) = \Psi(V^0, V^h)P(V^0), \quad (5.6)$$

where $\Psi(V^0, V^h)$ is the accumulated subdivision coefficient matrix from V^0 to V^h . Similar to the proposed lighting model, a simple Loop subdivision-based parameterization imposes a strong smoothness on the geometry that makes some high-frequency surface details impossible to be recovered. Thus, we also apply the adaptive refinement on $P(V^h)$,

$$P(V^h) = \Psi_P(V^s, V^h)P(V^s), \quad (5.7)$$

where $\Psi_P(V^s, V^h)$ is the updated accumulated subdivision coefficient matrix and V^s is the set of new control vertices generated using the same method described in Section 5.2.2 with the threshold ϵ replaced by a smaller threshold ϵ_P .

5.3.2 Optimization

To simplify the description, we assume that the initial MVS mesh is a semi-regular triangular mesh. Denote the mesh by $\mathcal{M} = \{V, E, F\}$ where $V = \{v_i\}$ is the set of vertices, E is the set of edges and F is the set of faces. If the initial MVS mesh is not semi-regular, it can be re-meshed into a semi-regular one. Suppose $\mathcal{M}$ is topologically obtained from $\mathcal{M}^0 = \{V^0, E^0, F^0\}$ by h iterations of subdivision described in Section 5.3. We have the corresponding overall illumination vectors $L(V^{s_1})$ serving as the control overall illumination vectors and vertex positions $P(V^{s_2})$ as the control vertex positions. The overall illumination vectors $L(V)$ and positions $P(V)$ on mesh $\mathcal{M}$ are the linear combination of $L(V^{s_1})$ and $P(V^{s_2})$, with

$$L(V) = \Psi(V^{s_1}, V)L(V^{s_1}) \quad (5.8)$$

and

$$P(V) = \Psi(V^{s_2}, V)P(V^{s_2}) \quad (5.9)$$

where $\Psi(V^{s_1}, V)$ and $\Psi(V^{s_2}, V)$ are the accumulated weight matrices for $L(V^{s_1})$ and $P(V^{s_2})$, respectively. Our goal now is to compute $L(V^{s_1})$ and $P(V^{s_2})$ of $\mathcal{M}$.

Considering a vertex v_i , we use $I_k(v_i)$ to represent the intensity value of vertex v_i in its corresponding k -th image. Let $c_i = \frac{\sum_{k=1}^m I_k(v_i)}{m}$ where m is the number of multi-view images containing v_i . Then according to (2.11), we have

$$c_i = L(v_i) \cdot \mathbf{n}(v_i) \quad (5.10)$$

This suggests that we introduce **shading terms**:

$$E_{sh} = \frac{1}{N} \sum_{i=1}^N \|L(v_i) \cdot \mathbf{n}(v_i) - c_i\|^2 + \frac{1}{N_e} \sum_{(v_i, v_j) \in E} \|(L(v_i) \cdot \mathbf{n}(v_i) - L(v_j) \cdot \mathbf{n}(v_j)) - (c_i - c_j)\|^2 \quad (5.11)$$

where N and N_e are the numbers of vertices and edges of the mesh, respectively. The first term of E_{sh} is the intensity error measuring the difference between the computed diffuse reflection and the average of the captured intensities. The second term of E_{sh} is the gradient error measuring the difference between the gradients of the computed reflected radiance and the average of the captured intensities [90]. The gradient error imposes another constraint on the normal changes making the model more stable. The normal $\mathbf{n}(v_i)$ can be expressed as a function of v_i and its surrounding vertices. Thus the shading term is a function of v_i and $L(V^{s_1})$.

We next introduce two geometry-oriented **fidelity terms**:

$$E_f = \frac{1}{N} \sum_{i=1}^N (\|v_i - v_i^{in}\|^2 + \|\mathbf{n}(v_i) - \mathbf{n}_i^{in}\|^2) \quad (5.12)$$

where the superscript "in" indicates the vertices and the normals of the input mesh, and **Laplacian term**:

$$E_{lap} = \frac{1}{N} \sum_{i=1}^N \|v_i - \bar{v}_i\|^2 \quad (5.13)$$

where $\bar{v}_i$ is the average of all the 1-ring neighboring vertices of v_i . The fidelity term is introduced to prevent the adjusted vertices v_i and their normals $\mathbf{n}(v_i)$ from deviating from their counterparts of the initial MVS mesh too much. The Laplacian term is computed as the squared sum of the Laplacian of all vertices, which helps avoid generating singular triangles in the mesh updating process and make the updated mesh smooth.

It is observed in our experiments that the smoothness assured by Loop subdivision is not good enough to recover the overall illumination vectors. Therefore, we add a simple weighted smoothness term to further enforce the similarity of nearby overall illumination vectors in the optimization process, which is defined as

$$E_{Ldif} = \frac{1}{N_e} \sum_{(v_i, v_j) \in E} \omega^{ij} \|L(v_i) - L(v_j)\|^2 \quad (5.14)$$

with weights $\omega^{ij} = \exp(-(\frac{\|I(v_i) - I(v_j)\|^2}{2\sigma^2}))$.

By combining all these terms together, we have our overall model:

$$\min \left\{ \frac{\alpha}{2} E_f + \frac{\beta}{2} E_{sh} + \frac{\eta}{2} E_{lap} + \frac{\gamma}{2} E_{Ldif} \right\} \quad (5.15)$$

where α , β , η and γ are the tradeoff factors. In this model, the overall illumination control vectors $L(V^{s_1})$ and the control positions $P(V^{s_2})$ are the variables to be determined.

5.3.3 Numerical Computation

To solve model (5.15), we alternatively solve two sub-problems. The first one is to solve for lighting while fixing geometry; and the second one is to solve for geometry while keeping the lighting unchanged. The initial MVS model serves as the initialization. The process iterates until convergence.

- (1) **Lighting**: In this step, we assume that the vertex positions v_i are fixed and only the overall illumination vectors $L(v^{s_1})$ is treated as unknowns. The minimization problem is simplified to be

$$\min \left\{ \frac{\beta}{2} E_{sh} + \frac{\gamma}{2} E_{Ldif} \right\} \quad (5.16)$$

This is a least-square problem. The objective function can be rewritten as $\frac{\beta}{2} E_{sh} + \frac{\gamma}{2} E_{Ldif} = \sum_{i=1}^n \|A_i L(V^{s_1}) - b_i\|^2$. By substituting Eqn. 5.8, the problem becomes

$$\min \{\|\mathbf{A}L(V^{s_1}) - \mathbf{b}\|^2\}, \quad (5.17)$$

where $\mathbf{A} = \begin{bmatrix} A_1 \\ \vdots \\ A_n \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$, which is equivalent to solving the linear system

$\mathbf{A}^T \mathbf{A} L(V^{s_1}) = \mathbf{A}^T \mathbf{b}$. The control overall illumination vectors are then obtained as $L(V^{s_1}) = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b}$.

- (2) **Geometry**: In this step, $L(v^{s_1})$ is fixed and we find new positions $P(v^{s_2})$ for position control vertices so that the new mesh better matches the intensity captured in the multi-view images. The problem becomes

$$\min \left\{ \frac{\alpha}{2} E_f + \frac{\beta}{2} E_{sh} + \frac{\eta}{2} E_{lap} \right\}. \quad (5.18)$$

Due to the non-differentiability of the unit normal vectors, we solve this problem using Levenberg-Marquardt algorithm. We rewrite the objective function as $\frac{\alpha}{2} E_f + \frac{\beta}{2} E_{sh} +$

$\frac{\eta}{2}E_{lap} = \sum_{i=1}^n f_i^2(P(V^{s_2}))$. We compute $P(V^{s_2})$ by an iterative process. At each iteration, $P(V^{s_2}) = P(V^{s_2}) + \Delta P(V^{s_2})$, where $\Delta P(V^{s_2})$ is computed from

$$\mathbf{J}^T \mathbf{J} \Delta P(V^{s_2}) + \mu \text{diag}(\mathbf{J}^T \mathbf{J}) = -\mathbf{J}^T \mathbf{f} \quad (5.19)$$

with the Jacobian matrix

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial P(V^{s_2})} \\ \vdots \\ \frac{\partial f_n}{\partial P(V^{s_2})} \end{bmatrix}, \quad (5.20)$$

$\mathbf{f} = (f_1, \dots, f_n)^T$, and the diagonal matrix $\text{diag}(\mathbf{J}^T \mathbf{J})$ consisting of the diagonal elements of $\mathbf{J}^T \mathbf{J}$ and the damping factor μ , which is adjusted at each iteration to accelerate the convergence.

5.4 Experiments

We have conducted several experiments to validate the proposed method. We first study the effectiveness of the subdivision-based illumination model compared to the per-vertex overall illumination model [93] on synthetic dataset with ground-truth geometry. Then we perform shading-based shape recovery on challenging real-world datasets with rich geometrical details. For each example, we normalize the vertex positions using the diagonal distance of the bounding box. Then we empirically set the parameters $\alpha = 1.0e + 5$, $\beta = 1.0e + 2$, $\eta = 1.0e + 4$, $\gamma = 10$ and $\sigma = 5$. We choose thresholds $\varepsilon = (0.4)I_{max}$ and $\varepsilon_P = (0.1)I_{max}$, where I_{max} is the maximum of the observed intensities. Our experiments are conducted on a standard PC with 3.70GHz Intel Core i7 CPU and 16GB of memory.

5.4.1 Evaluation on Subdivision-based Overall Illumination Model

We validate our lighting model by computing the distribution of $L(v)$ on the synthetic data under general and natural lighting conditions from lighting probe images of [17]. A light probe image is an omnidirectional, high dynamic range image that records incident illumination conditions at a particular point in space. Such images are captured under general and natural illumination conditions. We use the light probe image “campus” to illuminate a semi-regular mesh “buddha”. We first generate a base mesh with 2K vertices by a mesh simplification process via edge collapse, followed by performing three times Loop subdivision to obtain a semi-regular mesh with 128K vertices. We extract light sources using the MedianCut method of HDR Shop from each light probe image. During light source extraction, we set the number of extracted light source to be 4 and

32, which leads to two lighting configurations, “campus4” and “campus32”. Fig. 5.4 shows the meshes of “buddha” illuminated by the two light sources.

We compare our method with the per-vertex based overall illumination model in [93] and three other baselines, i.e. (a) per-vertex model with the smoothness constraint E_{Ldif} replacing the original TV-based constraint; (b) a simple subdivision-based model; (c) a subdivision-based model with the additional smoothness constraint by E_{Ldif} . The quantitative results are shown in Table 5.1, where we evaluate both angular and magnitude errors on the recovered overall illumination vectors. The illumination vectors directly computed from the light sources are used as our ground-truth for lighting. Note that we use both ground-truth surface normals and MVS normals for the evaluation since our proposed lighting model is used for surface recovery which generally does not have accurate input surfaces. To generate normals that are likely different from those of the accurate surface, we apply Laplacian smoothing 5 times to over-smooth the “buddha” model for the computation of the initial normals.

Table 5.1 shows that our subdivision-based illumination model greatly reduces the number of variables required to model the overall illumination vectors. Similar results are observed on experiments with ground-truth normals and MVS normals. The number of independent overall illumination vectors drops from 128K for the per-vertex overall illumination vector of [93] to a few thousands with the proposed subdivision-based illumination model.

Table 5.1 also shows that the smoothness assured by Loop subdivision is not good enough for lighting recovery. That is, the results generated from the subdivision-based method without any additional smoothness constraint have large angular and magnitude errors. The weighted smoothness term E_{Ldif} achieves a similar effect as the TV term used in [93]. For solving the TV-based approach, an augmented Lagrangian(ALM)-based solver that has an additional loop during the optimization is required. Thus, the proposed method is faster than the TV-based solution. In particular, for the “buddha” dataset, the computational time for the overall illumination vector recovery using the TV-based method and the proposed method are about 5 minutes and 1 minute, respectively.

On the other hand, it can be seen from the results that the proposed adaptive refinement on the subdivision-based overall illumination model improves the accuracy of the recovered overall illumination vectors. As shown in Fig. 5.3, our proposed adaptive refinement scheme adds independent overall illumination vectors at the sharp illumination change regions, while still preserving overall subdivision smoothness, avoiding over-parametrization on the lighting variables. Fig. 5.5 visualizes the improvement of the refinement on the “buddha” model with “campus4” lighting. This suggests that a straightforward application of the subdivision constraints on the illumination model has

FIGURE 5.4: Examples of colorized light vectors generated from different illumination conditions on the buddha model with “campus4” (left) and “campus32” (right). Self-shadows are more obvious on the mesh illuminated by “campus4” lighting.

TABLE 5.1: Quantitative results on angular and magnitude errors for overall illumination vector recovery. N_L is the number of independent overall illumination vectors.

		GT Normal		MVS Normal	
	N_L	Ang. (deg)	Mag. (%)	Ang. (deg)	Mag. (%)
Buddha model with “Campus4” lighting					
per-vertex L [93]	128K	14.139	7.620	17.480	8.736
per-vertex L + Ldif smooth	128K	13.170	8.475	17.478	9.964
subdiv-based L only	2K	18.451	12.840	31.594	51.113
subdiv-based L + Ldif smooth	2K	13.235	7.820	17.103	10.146
subdiv-based L + Ldif smooth + refine. (Proposed)	6.1K	12.701	7.155	16.034	8.323
Buddha model with “Campus32” lighting					
per-vertex L [93]	128K	16.232	6.549	21.290	8.790
per-vertex L + Ldif smooth	128K	16.318	5.710	21.253	6.870
subdiv-based L only	2K	19.914	8.916	38.837	39.346
subdiv-based L + Ldif smooth	2K	14.646	6.585	21.359	11.218
subdiv-based L + Ldif smooth + refine. (Proposed)	2.9K	14.382	6.042	20.391	7.574

TABLE 5.2: Quantitative results on distance errors for surface recovery on synthetic datasets. N_V , N_L and N_P are the number of vertices, independent overall illumination vectors and independent positions, respectively. Acc. is the averaged relative distances (normalized by the diagonal distance of the bounding box) of ground-truth vertices to recovered surfaces.

	Scan33				Scan49				Scan74				Scan109			
	$N_V(K)$	N_L	$N_P(K)$	Acc.(%)	$N_V(K)$	N_L	$N_P(K)$	Acc.(%)	$N_V(K)$	N_L	$N_P(K)$	Acc.(%)	$N_V(K)$	N_L	$N_P(K)$	Acc.(%)
Input	-	-	-	1.52	-	-	-	1.54	-	-	-	1.81	-	-	-	1.30
[90]	128	25	128	1.31	128	25	128	1.63	128	25	128	1.46	128	25	128	1.28
[93]	128	128K	128	1.20	128	128K	128	1.29	128	128K	128	1.30	128	128K	128	1.05
Proposed-L	128	3.5K	59	1.15	128	4K	43	1.26	128	3.8K	39	1.23	128	2K	40	1.01
Proposed-H	512	11K	200	1.14	512	11K	152	1.22	510	11K	140	1.22	508	8K	147	0.99

FIGURE 5.5: A visual comparison between our subdivision illumination methods with and without the proposed adaptive refinement for “buddha” model with “campus4” lighting.

FIGURE 5.6: Quantitative results on proposed method with different number of independent overall illumination vectors on “buddha” model with ground-truth normal and “campus4” and “campus32” lighting.

difficulty to recover the hard shadow regions and might result in a wrongly smoothed shading effect.

Fig. 5.6 demonstrates the effects on different numbers of independent vectors with different thresholds ε . A larger ε leads to a smaller number of independent vectors. Increasing the number of independent vectors can better satisfy the shading constraints, while it weakens the subdivision smoothness, resulting in large errors in the recovered overall illumination vectors. On the other hand, if too strong subdivision smoothness is imposed when only a small number of independent vectors are added, the light estimation results also become worse due to insufficient degrees of freedom for modeling the overall illumination changes over the mesh.

5.4.2 Evaluation on Surface Reconstruction

Since the proposed illumination model is mainly used for diffuse surface recovery, we test our method on surface recovery for various real-world datasets. We select four scans from DTU MVS Data Set [1], which contains various models including stuffed toys, animal statues and groceries. These datasets are with rich geometric details and non-uniform albedos. High quality 3D scanned point clouds are provided by the DTU dataset and we use them as ground-truth for the shape recovery evaluation. We adopt MVS methods [38] to generate an initial mesh as an input for later shading-based refinement. We adopt the same procedure for generating a semi-regular mesh as that in Section 5.4.1. We compare with the 4th order SH-based method [90] (our implementation) and the per-vertex illumination model [93] on the same semi-regular mesh input. We also run our proposed method on the input mesh with a higher resolution but using the number of variables similar to that used in [93]. Particularly, the low resolution input meshes contain $\sim 128\text{K}$ vertices and the high resolution input meshes contain $\sim 512\text{K}$ vertices through three times Loop subdivision from a base mesh with 2K and 8K vertices, respectively. Note that we only modify the topology of the initial input MVS meshes and their input geometries are unchanged. This is achieved by projecting the generated semi-regular meshes back to the corresponding input MVS mesh. We adopt a GPU-accelerated implementation to preform the projection, which takes less than 1 minute for each mesh. The baseline method of firstly performing shading-based refinement on a low resolution mesh followed by subdividing the refined mesh to a high resolution mesh is omitted in this experiment, since direct mesh subdivision introduces geometry distortions.

Our quantitative results are shown in Table 5.2. It can be observed that the proposed method significantly reduces the number of overall illumination variables used while

improves the accuracy of the recovered surfaces compared to the per-vertex illumination model [93]. For visual comparisons, we also perform a test on a captured real-world scene using the same experimental settings, as shown in Fig. 5.8. In this example, 25 images with a resolution of 1280×1024 are captured under challenging natural lighting with shadows. Since the ground-truth is in the form of point cloud, which in general has an unsatisfying rendering quality, the input mesh with color is visualized as the reference for surface details. Visual comparison is provided in Fig. 5.9 with highlighted regions showing better surface details recovered from our proposed method on both DTU datasets and our captured real-world example.

Computational cost: The proposed model significantly reduces the computational time. Wu’s method [90] takes half an hour to estimate the lighting and visibility functions for a mesh with 30K vertices. The computational time increases when dealing with a larger mesh. In our case, it only takes $2 \sim 7$ minutes to compute the overall illumination vectors on meshes with 512K vertices. The averaged computational time for the surface refinement are approximately 5 hours, 2 hours and 1 hour for an input mesh with 128K vertices for [90], [93] and our method, respectively.

Meanwhile, the proposed lighting model requires $2.5\% \sim 5\%$ memory cost compared to [93]. In addition, our proposed adaptive subdivision-based scheme on vertex positions reduces the number of vertices to about 40%, enabling the possibility of shading-based geometry refinement on larger meshes.

Limitations: Although the proposed method achieves memory efficiency and is capable to handle illumination modeling for Lambertian surface that contains a large number of vertices, our current shading-based refinement framework may not handle non-Lambertian effects well, such as reflective or transparent objects. Our current implementation also has other drawbacks. First, the number of base mesh vertices shouldn’t be too small. Otherwise the semi-regular mesh fails to be projected to the input mesh which may introduce large geometric distortions. Second, the running time of the entire framework is still relatively long in our current implementation. It would be interesting to use a parallel scheme for acceleration on GPU by utilizing the hierarchical structure of the semi-regular mesh.

5.5 Conclusion

We have described a subdivision-based variational approach for recovering surface details of objects from multi-view images captured under general unknown illuminations. Adaptive subdivision schemes are proposed to model both lighting and the reconstructed

surfaces. This reduces the number of variables used in the optimization procedure and achieves memory efficiency. It also enhances the regularization of the lighting. The experimental results demonstrate that the proposed method can well recover illumination and surface details of Lambertian objects with less computational costs.

FIGURE 5.7: Examples of DTU datasets. Top row: Image from one of the captured views. Bottom row: Rendered ground-truth point cloud.

FIGURE 5.8: Examples of captured images for our challenging real-world dataset.

FIGURE 5.9: Visual comparisons between [93] and the proposed method on the DTU datasets and our captured dataset. From top to bottom: scan33, scan49, scan74, scan109 and captured datasets.

Chapter 6

SubdSH: Subdivision-based Spherical Harmonics Field for Real-time Shading-based Depth Refinement

6.1 Overview

From previous chapter, we present a method to perform shading-based surface refinement using subdivision-based lighting model on semi-regular meshes. To extend this work to RGB-D data, in this chapter, we introduce our novel shading-based surface refinement method that improves our previous subdivision-based lighting model. The key differences are that the new method runs in real-time by exploiting the parallelism on GPU and works with color and depth streams from consumer-grade RGB-D sensors.

In particular, most of the environmental lighting used for the shading-based surface refinement on consumer-grade RGB-D camera is typically modeled as either simple distant lighting [67, 91, 105] or light sources at known positions [66]. As a result, test examples were usually scenes containing only foreground objects. This was the case until only recently, when Robert et al. [59] adopted a spatially-varying local lighting model to represent real world illumination in which lights were not placed at infinity. However, its trilinear-based technique is coarse and solved on CPU. We have not yet encountered a technique that can estimate more complex environmental lighting in real-time, which can be integrated into online shape recovery systems [91].

(A) Input Intensity

(B) Shading recovered from global lighting model [91]

(C) Shading recovered from proposed lighting model

FIGURE 6.1: Shading-based geometry refinement relies on a good approximation of lighting. However, shading cues from challenging lighting conditions (top) are not easily described by a traditional single global lighting model widely used in existing shading-based refinement methods (middle). Our subdivision-based local illumination model can better recover the shading and improve the quality of the refined geometry (bottom).

In this chapter, we extend our previous subdivision-based lighting model and propose a novel lighting model for shading-based surface recovery on RGB-D images that can be performed on a complete scene where the lighting conditions can vary across different locations. To achieve this, we seek to estimate the local lighting based on *local* Spherical Harmonics (SH) rather than global ones. Our key idea is to estimate a smoothly varying lighting field that is described by a few control points, determined through Catmull-Clark subdivision [14] over the image grid. The lighting estimation process is accelerated by exploiting parallelization on GPU. As a result, we are able to perform fast shading-based depth refinement under challenging lighting conditions, as shown in Fig. 6.1.

6.2 New Lighting Representation

The computation of shading cues involves lighting and surface normal. In this section we introduce a new representation for the unknown lighting, which facilitates our lighting estimation and depth refinement of RGB-D images.

6.2.1 Image Formation

Similar to previous work [59, 67, 91, 105], we assume that our surfaces are Lambertian and the incident lighting is parameterized using spherical harmonics (SH). The reflected irradiance B at pixel (i, j) is computed as

$$B(i, j) = \rho(i, j) \sum_{m=1}^{b^2} l_m H_m(n(i, j)) \quad (6.1)$$

where $\rho(i, j)$ is the albedo, b is the number of SH bands, l_m is the lighting parameters and H_m is the SH basis evaluated at unit normal $n(i, j)$.

Consider an RGB-D image and denote by $D(i, j)$ the depth at pixel (i, j) . The 3D position $p(i, j)$ of the point corresponding to pixel (i, j) can be computed from the pin-hole camera model:

$$p(i, j) = \begin{bmatrix} D(i, j)(i - c_x)/f_x \\ D(i, j)(j - c_y)/f_y \\ D(i, j) \end{bmatrix} \quad (6.2)$$

where f_x and f_y represent the focal lengths of the camera and (c_x, c_y) is the principal point. The normal $n(i, j)$ of the surface at point $p(i, j)$ can be estimated from neighbouring points: $n(i, j) = \frac{\tilde{n}(i, j)}{\|\tilde{n}(i, j)\|}$ with

$$\tilde{n}(i, j) = (p(i, j - 1) - p(i, j)) \times (p(i - 1, j) - p(i, j)). \quad (6.3)$$

The global lighting model described in (6.1) has been widely adopted in many shading-based geometry refinement algorithms [91, 105], which have demonstrated promising results in recovering surface detail. However, this global lighting model has an obvious shortcoming in that it works only for simple lighting environments where lights are assumed to be distant and always visible, because the lighting is modeled by a single set of SH coefficients. In order to overcome this problem, our initial idea is to introduce a per-pixel local SH lighting representation, i.e.,

$$B(i, j) = \rho(i, j) \sum_{m=1}^{b^2} l_m(i, j) H_m(n(i, j)), \quad (6.4)$$

where $\{l_1(i, j), \dots, l_{b^2}(i, j)\}$ is a vector consisting of the local lighting parameters at pixel (i, j) . This is equivalent to estimating the local environmental lighting distribution at each pixel. One advantage of this representation is that visibility and self-occlusion are implicitly handled once the per-pixel lighting is computed. But it should be noted that this representation often leads to over parameterizations, which may lead to non-unique lighting solutions.

6.2.2 SubdSH: Subdivision Representation for SH Illumination Field

Let $L(i, j) = [l_1(i, j), l_2(i, j), \dots, l_{b^2}(i, j)]^T$ be a vector representing the local illumination. We can consider these as discrete samples from a vector-valued function $L(s, t) = [l_1(s, t), l_2(s, t), \dots, l_{b^2}(s, t)]^T$ with continuous parameters (s, t) . If function $L(s, t)$ is determined, the local lighting vector $L(i, j)$ becomes known. Therefore we need to find an appropriate representation or model for function $L(s, t)$.

Our second idea is motivated by Land’s Retinex Theory [51], which assumes that illumination typically changes the surface brightness gradually, leading to small image gradients. Thus strong gradients usually imply changes in surface material, which is independent of lighting. Land’s Retinex Theory has been used in many works to separate surface albedo and shading [52, 66]. For our purpose, we will assume that our lighting function $L(s, t)$ is a smooth function, but remove the strong assumption that the lighting is distant or consists of directional lights. Moreover, we propose to use Catmull-Clark subdivision [14] to model the vector function $L(s, t)$. In particular, the image grid is a special quadrilateral mesh. The inverse operation of building an image pyramid correspond to the process of generating subdivided meshes at finer levels, as shown in Fig. 6.2.

To simplify notation, let u denote a point on the image grid. Then $L(s, t)$ is $L(u)$. The subdivision-based lighting has the advantage of compact representation, i.e. the lighting

FIGURE 6.2: The image grid can be considered as a quad mesh and the image pyramid corresponds to the inverse operation of generating subdivided meshes from a base mesh.

FIGURE 6.3: Catmull-Clark Subdivision Scheme. Left: the mask for computing a new face point. Middle: the mask for computing a new edge point. Right: the mask for computing a vertex point where n_e is the number of adjacent edge points and n_f is the number of adjacent face points, $n = n_e + n_f$, $\beta_1 = \frac{2}{(n_e + n_f)n_e}$ and $\beta_2 = \frac{1}{(n_e + n_f)n_f}$.

field $L(u)$ can be described by only a few control points. We start with the base mesh $\mathcal{M}^0$ and each vertex contains an SH coefficient vector $L(u^0)$. Through h iterations of Catmull-Clark subdivision on both $\mathcal{M}^0$ and $L(u^0)$, $\mathcal{M}^0$ becomes $\mathcal{M}^h$ whose grids are denoted by u^h and $L(u^0)$ becomes $L(u^h)$. Fig. 6.3 shows the masks for one iteration of Catmull-Clark subdivision from a coarse to a finer level.

Let $L(U^0)$ and $L(U^h)$ denote the vectors containing all the SH lighting coefficients $L(u^0)$ and $L(u^h)$, respectively. $L(U^h)$ can be expressed as a linear combination of $L(U^0)$:

$$L(U^h) = \Psi(U^0, U^h)L(U^0) \quad (6.5)$$

where $\Psi(U^0, U^h)$ is the accumulated coefficient matrix for subdivision from U^0 to U^h . Based on the subdivision rules, the accumulated coefficient matrix can be obtained recursively. That is, for $l = 2, 3, \dots, h$, we have

$$\Psi(U^0, U^l) = \Psi(U^{l-1}, U^l)\Psi(U^0, U^{l-1}) \quad (6.6)$$

where $\Psi(U^{l-1}, U^l)$ is the subdivision coefficient matrix from level $l-1$ to level l , which can be derived from the masks given in Fig. 6.3.

6.3 Real-time Shading-based Depth Refinement

We now incorporate our proposed lighting representation into a real-time shading-based depth refinement framework similar to the one in [91]. Our input is the initial depth D^i with its corresponding color image I_c . We want to output a refined depth D . To solve the problem, we take a hierarchical strategy. Specifically, we build an image pyramid with the RGB-D data and proceed in a coarse-to-fine manner. At each pyramid level, we adopt a two-step procedure by which we first solve for the spatially varying lighting and then refine the estimated depths. We alternate these two steps until the finest image level is reached. The details of the two steps are given below.

6.3.1 Lighting Estimation

To estimate the lighting coefficients $L(U^0)$, we minimize the difference between the rendered image B and the capture intensity I^h at level h , which is formulated as follows:

$$E_L = \sum_{i=1}^{W^l \times H^l} (B(u_i) - I^l(u_i))^2 \quad (6.7)$$

where (W^l, H^l) is the image size at current level l . This is a least-squares problem, whose solution can be obtained by solving the linear system:

$$A \cdot (\Psi(U^0, U^h) \cdot L(U^0)) = I^l \quad (6.8)$$

where

$$A = \begin{bmatrix} H_0(n(u_1)) & \cdots & H_{b^2}(n(u_1)) \\ H_0(n(u_2)) & \cdots & H_{b^2}(n(u_2)) \\ \vdots & \cdots & \vdots \\ H_0(n(u_{W^l \times H^l})) & \cdots & H_{b^2}(n(u_{W^l \times H^l})) \end{bmatrix}. \quad (6.9)$$

Wu et al. [91] estimated the lighting coefficients at only a few coarse levels because they could easily obtain enough observations to find the global SH lighting coefficients. However, in our case, we need to maintain $\Psi(U^0, U^h)$ and compute $A^T A$ and $A^T I$ on-the-fly, which slows down the lighting estimation process. Therefore, we choose to adapt a strategy used in [59]. We solve for the lighting coefficients $L(U^0)$ locally, based on the neighboring shading constraints, as our real-time solution.

Let $H(u_i) = [H_0(n(u_i)), \dots, H_{b^2}(n(u_i))]^T$. We then have $L(u_i) \cdot H(u_i) = I(u_i)$. We perform the local lighting estimation based on the similarity between lighting coefficients between current pixel $u_i^l = (i, j)$ and its neighboring pixels $NN^{l+1}(u_i^l) = \{u_j^{l+1} = (2i + \Delta i, 2j + \Delta j)\}$ in the finer level $l + 1$ with $|\Delta i| \leq \sigma_r$ and $|\Delta j| \leq \sigma_r$. Since we assume our lighting field is smooth, then $L^l(u_i^l) \approx L^{l+1}(u_j^{l+1})$. This means that instead of having a pixel-wise shading constraint, we can have a regional shading constraint:

$$L^l(u_i^l) \cdot \sum_{u_j \in NN^{l+1}(u_i^l)} H^{l+1}(u_j) = \sum_{u_j \in NN^{l+1}(u_i^l)} I^{l+1}(u_j). \quad (6.10)$$

We design our parallelized solution based on a bottom-up approach to aggregate the local sums of neighboring SH basis and intensities, as shown in Fig. 6.4. In this way, we can have the sampled sums of SH bases H_Σ^0 and intensities I_Σ^0 at the base level. For each local SH lighting at pixel (i, j) in the base image, we reformulate the problem into a local least squares problem:

$$E_L(L(u^0)) = \sum_{u_i \in NN(u^0)} (L(u^0) \cdot H_\Sigma^0(u_i) - I_\Sigma^0(u_i)) \quad (6.11)$$

where $NN(u^0)$ represents a local region around u^0 at the base mesh, and introduces overlapping samples between neighboring vertices to further ensure lighting smoothness.

FIGURE 6.4: Our region-based shading constraint for fast local lighting estimation.

Thus we solve the following linear system in parallel:

$$A_{\Sigma} \cdot L(u_i^0) = I_{\Sigma}^0, \quad (6.12)$$

where A_{Σ} is constructed in a way similar to Eqn. 6.9 using H_{Σ}^0 , and I_{Σ}^0 contains all the sampled intensity sums.

After obtaining $L^0(U^0)$, we use Eqn. 6.5 to evaluate the SH coefficients for all pixels at the current level, and we obtain the albedo image $I_a(u) = (\frac{I_r(u)}{S(u)}, \frac{I_g(u)}{S(u)}, \frac{I_b(u)}{S(u)})$ with $S(u) = L(u) \cdot H(u)$.

6.3.2 Depth Map Refinement

Once lighting and albedo are obtained, we then refine the depth D through an energy optimization process based on shading cues. Similar to [91], our data term measures the gradient error between the computed diffuse reflection and the image intensities:

$$E_{sh} = w^r (\nabla B^r - \nabla I^r)^2 + w^c (\nabla B^c - \nabla I^c)^2 \quad (6.13)$$

where ∇B^r and ∇B^c are the horizontal and vertical shading gradient, and ∇I^r and ∇I^c are the horizontal and vertical intensity gradients, and w^r and w^c are the masks. Both masks are set to zero for albedo boundary edges, which are detected from the albedo image I_a based on distances in RGB color space by a user-defined threshold.

The second term in our energy function is a Laplacian term, which helps avoid generating singular points:

$$E_{lap} = \|p - \bar{p}\|_2^2 \quad (6.14)$$

where p is the point position that is computed from Eqn. 6.2 and $\bar{p}$ is the average of all the 1-ring neighboring vertices of p .

For the depth fidelity term, we propose to use a novel term that is based on the edge-preserving range image refinement [72], which can be efficiently implemented on GPU. Unlike constraining based on input depth, we restrict the refined depth to the upsampled depth from the coarser level:

$$E_{fid} = (D - \tilde{D}^l)^2. \quad (6.15)$$

Here $\tilde{D}^l$ is generated from a dual resolution bilateral filter:

$$\begin{aligned}\tilde{D}_p^l &= \frac{\sum_{q \in S} w_{pq} \tilde{D}_q^{l-1}}{\sum_{q \in S} w_{pq}} \\ w_{pq} &= s(\|p - q\|)r(I_p^l - I_q^{l-1}) \\ \tilde{D}^0 &= D^0\end{aligned}\tag{6.16}$$

where D^0 is the depth at the coarsest pyramid generated from D^i , $s(\cdot)$ and $r(\cdot)$ are the spatial and range kernels for the bilateral filtering. With the new fidelity term, we should be able to remove the systematic noise on the refined depth introduced by D^i . We rely on the shading-based refinement to progressively recover the surface details.

By combining all these terms, we obtain our overall energy function:

$$E(D) = \lambda_f E_{fid} + \lambda_s E_{sh} + \lambda_r E_{reg}\tag{6.17}$$

where λ_f , λ_s and λ_r are the tradeoff factors. This is a nonlinear optimization problem. We solve it using the state-of-the-art GPU solver Opt [19].

6.4 Experimental Results

We tested our proposed shading-based depth refinement framework on both synthetic and real datasets. Our subdivision-based lighting model and the shading-based depth refinement framework were quantitatively evaluated on synthetic dataset where groundtruths are available. We then performed the shading-based depth refinement on a real world RGB-D dataset downloaded from the Internet, and several RGB-D sequences under challenging lighting conditions were also captured to evaluate our framework. We also compared our framework to our implementation of [91]. We empirically set the parameters $\sigma_r = 2$, $\lambda_f = 100$, $\lambda_s = 1$ and $\lambda_r = 400$. Our experiments were conducted on a desktop with 3.00GHz Intel Core i7 CPU and 32GB of memory with a NVIDIA GeForce GTX TITAN X graphic card.

We used an image pyramid with four levels to solve our lighting and refined depth. For speed purposes, we used first order SH to model our local lighting. Therefore, for an image with a resolution of 640×480 , we have $80 \times 60 \times 4 = 19.2K$ lighting parameters which is $\sim 2K$ times the number of variables in [91], which used only 9 parameters for the second-order global SH lighting. However, the average compute time for the lighting estimation is almost the same. With un-optimized GPU codes, we were able to solve the spatial variant lighting on an image with aforementioned resolution in around 1.32 ms. A detailed averaged time performance on solving lighting coefficients is reported in

Table 6.1. Our GPU-accelerated Catmull-Clark subdivision takes around $2 \sim 3$ ms to generate the per-vertex SH coefficients from the coarsest level to the finest level.

6.4.1 Quantitative Evaluation on Synthetic Data

We built our synthetic dataset from 3D objects in Stanford 3D¹ and Blendswap repositories². We rendered 5 scenes using 3DS Max under challenging lighting that consisted of directional, omni and spot light sources. To better evaluate our lighting estimation result, we used objects with uniform albedo. Background was also added with foreground objects for casting shadows. Examples of our rendered scenes are shown in Fig. 6.1, Fig. 6.5 and Fig. 6.6. For each model, we generated a color image and a depth map at the same resolution of 640×480 . Gaussian noise was added to the depth map to mimic the behavior of consumer-grade RGB-D cameras. We compared our method with the state-of-the-art system [91] on both lighting and depth recovery.

Lighting Estimation. We evaluated the accuracy of our lighting model by comparing the errors on the recovered shading. Since we built our dataset using uniform albedo, we computed the recovered shading error as $\epsilon_s = \frac{1}{N} \sum \|B(i, j) - I(i, j)\|$. The results are presented in Table 6.2, which shows that our subdivision-based SH lighting model significantly outperforms the single, global illumination model with much smaller shading errors. Our visual results are shown in Fig. 6.5 and Fig. 6.6, where it is clear that our method is able to recover spatial varying illumination that changes across the image, not only a global one.

We also tested the baseline method with bilinear interpolated local coefficients that is similar to the local lighting model in [59]. The recovered shading was almost identical to our proposed method. However, the bilinear interpolated method created visible jagged artifacts on the recovered shading as shown in Fig. 6.7, while our proposed subdivision-based local lighting model has smooth shading in the same area.

Depth Refinement. We tested the results on depth refinement. We exported the geometry of the entire scene from 3ds Max and used it as our groundtruth. We computed the 3D points using (6.2) and evaluated the averaged point-to-surface distances. The quantitative results are shown in Table 6.3, in which we also report the results from SIRF [5]. It can be seen that proposed method recovered shapes that are closest to the groundtruth surfaces. We obtain an averaged 5% improvement compared with the results from [91]. Visual comparisons are provided in Fig. 6.8 and Fig. 6.9 with highlighted regions comparing the refined details. Interestingly, high frequency details can still be

¹<http://graphics.stanford.edu/data/3Dscanrep/>

²<https://www.blendswap.com/>

TABLE 6.1: Averaged time performance on a 640×480 RGB-D image with our parallelized lighting solver implementation.

Preprocess	Sampling	Compute $A^T A$ and $A^T I$	Solve SH	Total
0.15ms	0.70ms	0.45ms	0.02ms	1.32ms

TABLE 6.2: Quantitative results on mean absolute error (MAE) of recovered shading.

Dataset	Wu et al. [91]	Proposed
<i>Happy</i>	22.37	7.28
<i>Armadillo</i>	17.54	7.42
<i>Statuette</i>	28.02	8.91
<i>Roman</i>	22.78	7.06
<i>Hylas</i>	34.33	5.02

TABLE 6.3: Quantitative results on recovered depth error (in mm).

Dataset	SIRF [5]	Wu et al. [91]	Proposed
<i>Happy</i>	10.01	2.89	2.81
<i>Armadillo</i>	32.99	4.01	3.90
<i>Statuette</i>	6.95	2.54	2.41
<i>Roman</i>	7.74	2.47	2.40
<i>Hylas</i>	2.23	1.71	1.53

(A) Input Intensity

(B) Recovered shading from [91]

(C) Recovered shading error from [91]

(D) Recovered shading from proposed

(E) Recovered shading error from proposed

FIGURE 6.5: Visual comparison on shading recovery between [91] and our proposed method on *Armadillo* dataset

(A) Input Intensity

(B) Recovered shading from [91]

(c) Recovered shading error from [91]

(D) Recovered shading from proposed

(E) Recovered shading error from proposed

FIGURE 6.6: Visual comparison on shading recovery between [91] and our proposed method on *Roman* dataset

(A) Input Intensity

(B) Bilinear interpolated

(C) Proposed

FIGURE 6.7: Visual comparison on shading recovery between bilinear interpolated local lighting model and proposed method on *Statuette*.

FIGURE 6.8: Visual comparison on depth refinement results between [91] and our proposed method on *Hylas* dataset. The highlighted regions in (b) and (c) show the wrongly recovered details where convex details are recovered as concave details.

(A) Input Intensity

(B) Refined geometry from [91]

(C) Error on refined geometry from [91]

(D) Refined geometry from proposed

(E) Error on refined geometry from proposed

FIGURE 6.9: Visual comparison on depth refinement results between [91] and our proposed method on *Happy* dataset.

recovered in the framework of [91]. However, because of error in the estimated lighting, some details are likely to be wrongly refined, e.g. a convex surface in groundtruth being recovered as a concave one.

6.4.2 Qualitative Evaluation on Real Data

We conducted qualitative experiments on real data from both NYU dataset [78] and our captured RGB-D sequences. These datasets were captured from large scenes under varying illuminations using consumer-grade RGB-D cameras which introduced a lot of noise. The results are shown in Figure 6.10, 6.11 and 6.12. Our framework was able to generate smooth depths while still perserving surface details on the noisy RGB-D data. More examples are available in our supplementary material.

6.4.3 Limitations

Although the proposed method relaxes the lighting assumption for shading-based geometry refinement, it cannot handle specularities since the Lambertian assumption still holds in our current framework. In addition, hard shadow regions may lead to artificial creases since we assumed a smooth lighting field. In addition, when we deal with noisy data, more iterations are required for convergence in our depth estimation. Our current per-frame based depth refinement implementation requires $50 \sim 100$ ms for refining depth.

6.5 Conclusion

We have described a subdivision-based local illumination model for recovering surface details on RGB-D images under challenging unknown lighting. The proposed local lighting model can be effectively solved on GPU through parallel scheme. We also build a real-time depth-refinement system by integrating our GPU-based lighting solution. The experimental results confirm that the proposed method can recover challenging lighting and thus improve the depth refinement results. Future work includes adopting a data-driven approach through deep neuron networks to improve the depth refinement performance and speed.

(A) Input Color

(B) Input Depth

(c) Depth recovered using [91]

(D) Depth recovered using proposed

FIGURE 6.10: Visual comparison on depth refinement results between [91] and our proposed method on NYU dataset. The details on the mattress are better recovered by the proposed method.

(A) Input Color

(B) Input Depth

(c) Depth recovered using [91]

(D) Depth recovered using proposed

FIGURE 6.11: Visual comparison on depth refinement results between [91] and our proposed method on NYU dataset. The surface of the cabinet are smooth but the details are well recovered by the proposed method.

FIGURE 6.12: Visual comparison on depth refinement results between [91] and our proposed method on our captured dataset. Proposed method handles well on the systematic error from consumer-grade RGB-D cameras.

Chapter 7

Conclusion and Future Works

7.1 Conclusion

In this thesis, we studied two related topics for improving 3D reconstruction qualities in the state-of-the-art 3D acquisition frameworks: consumer-grade RGB-D camera calibration and registration, and shading-based surface refinement. The former aims to reduce the systematic error, e.g. depth distortions, from the consumer-grade depth sensors. The latter targets to recover high quality surface details noisy and rough initial shapes using color information.

For the calibration and registration of multiple RGB-D camera, we introduced a new technique of registering multiple RGB-D cameras through a local rigid transformation. Registration of multiple RGB-D camera setups is a notoriously sensitive and delicate procedure despite the variety of available methods. The registration between RGB-D cameras which typically consists of a (single) rigid transformation often fails in practice to faithfully align all the depth and color streams consistently everywhere inside the capture volume, as observed in many experiments. We address this issue by proposing a practical and general approach to register the depth and color streams of multiple RGB-D camera setups. Our approach estimates a smooth field of rigid transformations between RGB-D cameras with $C(0)$ continuity on a regular grid within the captured volume. In terms of both visual quality and measurable reprojection errors, we demonstrate that our method provides better results than state-of-the-art methods.

In the meantime, we also have developed a multi-depth sensor registration method using easy-to-obtain everyday objects. Particularly, we use a common chair as an example and pre-scan it. Then, we apply state-of-the-art method to detect and locate the chair to generate correspondences between the pre-scan and the observation. We have proposed

an optimization framework that reserves the geometry of the pre-scanned shape and minimizes the distance between the overlapped region of two depth cameras. Experimental results show that our method significantly outperforms state-of-the-art methods in terms of the average point cloud distance as well as achieving much better visual quality. Our method is very useful for multi-depth sensor systems since it enables the capability of multi-depth sensor online calibration.

For surface detail refinement, we have described a subdivision-based variational approach for recovering surface details of objects from multi-view images captured under general unknown illuminations. Adaptive subdivision schemes are proposed to model both lighting and the reconstructed surfaces. This reduces the number of variables used in the optimization procedure and achieves memory efficiency. It also enhances the regularization of the lighting. The experimental results demonstrate that the proposed method can well recovery illumination and surface details of Lambertian objects and use a much smaller number of variables.

At last, we improved our subdivision-based local illumination model for recovering surface details on RGB-D images under challenging unknown lighting. The proposed local lighting model can be effectively solved on GPU through parallel scheme. We also build a real-time depth-refinement system by integrating our GPU-based lighting solution. The experimental results confirm that the proposed method can recover challenging lighting and thus improve the depth refinement results.

7.2 Future Directions

In terms of multiple RGB-D camera calibration and registration, an interesting future direction could be a self-calibration algorithm that adopts our proposed technique using everyday objects. After the initial calibration and registration process, the setup can be used to capture scenes containing various shapes, which can be later used to optimize the RGB-D camera parameters. By iterating this process, we may be able to design a self-calibration multiple RGB-D camera acquisition setup that can be easily deployed and maintained at any desired locations. In the meantime, it may be worth to integrate existing 3D reconstruction algorithms with high quality texture mapping into our calibration methods. We can make use of the visual features on the scanned objects for considering the visual correspondences for performing a joint color and depth camera system calibration.

For out shading-based surface refinement, since only Lambertian reflectance model is considered, one extension can be shading-based surface recovery on non-Lambertian

surfaces such as specular, translucent or transparent objects. If we can properly establish the relation between surface normals and different illumination effects such as highlights, reflection and refraction, we may be able to use them as photometric constraints for the surface refinement. On the other hand, regardless of adopting Lambertian reflectance model or the non-Lambertian one, the challenge rises from the various materials observed in the scene with completely different shading properties. Detecting surface materials given a scene is hard to be solved and usually yields strange artefacts like texture-copying effect. With recent breakthrough in the machine learning, it is possible to solve the problem through a data-driven method. Deep neuron networks trained on a few objects with different materials could be used to provide prior information on the surface materials given observed images or scenes. Similarly, challenging lighting may also be estimated adopting deep networks.

List of Publications

1. **Deng, T.**, Li, H., Cai, J., Cham, T.-J. and Fuchs, H. (2013) Kinect shadow detection and classification. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 708-713. IEEE.
2. **Deng, T.**, Bazin, J.-C., Martin, T., Kuster, C., Cai, J., Popa, T., and Gross, M. (2014). Registration of multiple rgbd cameras via local rigid transformations. In *Proceedings of International Conference on Multimedia and Expo*, pages 1-6. IEEE.
3. Kuster, C., Bazin J.-C., Öztireli, C., **Deng, T.**, Martin, T., Popa, T. and Gross, M. (2014) Spatio-temporal geometry fusion for multiple hybrid cameras using moving least squares surfaces." In *Computer Graphics Forum*, 33(2):1-10.
4. Dao, N.-L., **Deng, T.**, and Cai, J. (2014) Fast and automatic body circular measurement based on a single kinect. In *Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA)*, pages 1-4. IEEE.
5. **Deng, T.**, Cai, J., Cham, T.-J. and Zheng J. (2017) Multiple Consumer-grade Depth Camera Registration Using Everyday Objects. *Image and Vision Computing*, 62:1-7.
6. Tan, F., Fu, C.-W., **Deng, T.**, Cai, J. and Cham, T.-J. (2017) FaceCollage: A Rapidly Deployable System for Real-time Head Reconstruction for On-The-Go 3D Telepresence. In *Proceedings of the ACM on Multimedia Conference*, pages 64-72. ACM.
7. **Deng, T.**, Cai, J., Cham, T.-J. and Zheng J. (2018) Shading-based Surface Recovery Using Subdivision-based Representation. In *Computer Graphics Forum*.
8. **Deng, T.**, Cai, J., Cham, T.-J. and Zheng J. (2018) SubdSH: Subdivision-based Spherical Harmonics Field for Real-time Shading-based Depth Refinement. In *Proceedings of IEEE International Conference on Visual Communications and Image Processing*. IEEE.

Bibliography

- [1] Aanæs, H., Jensen, R. R., Vogiatzis, G., Tola, E., and Dahl, A. B. (2016). Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, pages 1–16.
- [2] Alldrin, N., Zickler, T., and Kriegman, D. (2008). Photometric stereo with non-parametric and spatially-varying reflectance. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.
- [3] Avetisyan, R., Willert, M., Ohl, S., and Staadt, O. (2013). *Calibration of Depth Camera Arrays*. PhD thesis, (Master thesis), Dept. of computer science, University of Rostock, Germany.
- [4] Ballard, D. H. (1981). Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122.
- [5] Barron, J. T. and Malik, J. (2013). Intrinsic scene properties from a single rgb-d image. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 17–24. IEEE.
- [6] Basri, R., Jacobs, D., and Kemelmacher, I. (2007). Photometric stereo with general, unknown lighting. *International Journal of Computer Vision*, 72(3):239–257.
- [7] Beck, S. and Froehlich, B. (2015). Volumetric calibration and registration of rgb-d sensors. In *Virtual Reality*, pages 151–152. IEEE.
- [8] Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [9] Biermann, H., Martin, I. M., Zorin, D., and Bernardini, F. (2001). Sharp features on multiresolution subdivision surfaces. In *Computer Graphics and Applications, 2001. Proceedings. Ninth Pacific Conference on*, pages 140–149. IEEE.
- [10] Bouguet, J. Y. Camera calibration toolbox for Matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/.

- [11] Bradski, G. et al. The opencv library. <http://opencv.org/>.
- [12] Butler, D. A., Izadi, S., Hilliges, O., Molyneaux, D., Hodges, S., and Kim, D. (2012). Shake’n’sense: reducing interference for overlapping structured light depth cameras. In *SIGCHI Conference on Human Factors in Computing Systems*.
- [13] Campbell, N. D., Vogiatzis, G., Hernández, C., and Cipolla, R. (2008). Using multiple hypotheses to improve depth-maps for multi-view stereo. In *Proceedings of European Conference on Computer Vision*, pages 766–779. Springer.
- [14] Catmull, E. and Clark, J. (1978). Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer-aided design*, 10(6):350–355.
- [15] Choi, S., Zhou, Q.-Y., and Koltun, V. (2015). Robust reconstruction of indoor scenes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 5556–5565. IEEE.
- [16] Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., and Theobalt, C. (2017). Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics*, 36(3):24.
- [17] Debevec, P. (2008). Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *ACM SIGGRAPH 2008 classes*, page 32. ACM.
- [18] Deng, T., Bazin, J.-C., Martin, T., Kuster, C., Cai, J., Popa, T., and Gross, M. (2014). Registration of multiple rgbd cameras via local rigid transformations. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 1–6. IEEE.
- [19] DeVito, Z., Mara, M., Zollöfer, M., Bernstein, G., Theobalt, C., Hanrahan, P., Fisher, M., and Nießner, M. (2016). Opt: A domain specific language for non-linear least squares optimization in graphics and imaging. *arXiv preprint arXiv:1604.06525*.
- [20] Diebel, J. and Thrun, S. (2006). An application of markov random fields to range sensing. In *Conference on Neural Information Processing Systems*, pages 291–298.
- [21] Dolson, J., Baek, J., Plagemann, C., and Thrun, S. (2010). Upsampling range data in dynamic environments. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1141–1148. IEEE.
- [22] Drost, B., Ulrich, M., Navab, N., and Ilic, S. (2010). Model globally, match locally: Efficient and robust 3d object recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 998–1005. IEEE.

- [23] Esteban, C. H., Vogiatzis, G., and Cipolla, R. (2008). Multiview photometric stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):548–554.
- [24] Facciolo, G. and Caselles, V. (2009). Geodesic neighborhoods for piecewise affine interpolation of sparse data. In *Proceedings of IEEE International Conference on Image Processing*, pages 365–368. IEEE.
- [25] Forsyth, D. A. (2011). Variable-source shading analysis. *International Journal of Computer Vision*, 91(3):280–302.
- [26] Fua, P. and Leclerc, Y. G. (1995). Object-centered surface reconstruction: Combining multi-image stereo and shading. *International Journal of Computer Vision*, 16(1):35–56.
- [27] Fuchs, S. and Hirzinger, G. (2008). Extrinsic and depth calibration of tof-cameras. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6. IEEE.
- [28] Fuhrmann, S. and Goesele, M. (2014). Floating scale surface reconstruction. *ACM Transactions on Graphics*, 33(4):46.
- [29] Furukawa, Y., Hernández, C., et al. (2015). Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148.
- [30] Furukawa, Y. and Ponce, J. (2010). Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376.
- [31] Habbecke, M. and Kobbelt, L. (2007). A surface-growing approach to multi-view stereo reconstruction. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.
- [32] Han, Y., Lee, J.-Y., and So Kweon, I. (2013). High quality shape from a single rgb-d image under uncalibrated natural illumination. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1617–1624. IEEE.
- [33] Herrera, C., Kannala, J., and Heikkilä, J. (2012). Joint depth and color camera calibration with distortion correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [34] Hertzmann, A. and Seitz, S. M. (2005). Example-based photometric stereo: Shape reconstruction with general, varying brdfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1254–1264.

- [35] Horn, B. K. (1987). Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*.
- [36] Ikeuchi, K. and Horn, B. K. (1981). Numerical shape from shading and occluding boundaries. *Artificial intelligence*, 17(1-3):141–184.
- [37] 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 *ACM Symposium on User Interface Software and Technology*.
- [38] Jancosek, M. and Pajdla, T. (2011). Multi-view reconstruction preserving weakly-supported surfaces. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3121–3128. IEEE.
- [39] Jia, Z., Chang, Y.-J., Lin, T.-H., and Chen, T. (2011). Dense interpolation of 3d points based on surface and color. In *Proceedings of IEEE International Conference on Image Processing*, pages 869–872. IEEE.
- [40] Jin, B., Lei, H., and Geng, W. (2014). Accurate intrinsic calibration of depth camera with cuboids. In *Proceedings of European Conference on Computer Vision*, pages 788–803. Springer.
- [41] Jin, H., Cremers, D., Wang, D., Prados, E., Yezzi, A., and Soatto, S. (2008). 3-d reconstruction of shaded objects from multiple images under unknown illumination. *International Journal of Computer Vision*, 76(3):245–256.
- [42] Jin, H., Soatto, S., and Yezzi, A. J. (2005). Multi-view stereo reconstruction of dense shape and complex appearance. *International Journal of Computer Vision*, 63(3):175–189.
- [43] Johnson, M. K. and Adelson, E. H. (2011). Shape estimation in natural illumination. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2553–2560. IEEE.
- [44] Joshi, N. and Kriegman, D. J. (2007). Shape from varying illumination and view-point. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1–7. IEEE.
- [45] Kajiya, J. T. (1986). The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150. ACM.
- [46] Kiechle, M., Hawe, S., and Kleinsteuber, M. (2013). A joint intensity and depth co-sparse analysis model for depth map super-resolution. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1545–1552. IEEE.

- [47] Kim, K., Torii, A., and Okutomi, M. (2016). Multi-view inverse rendering under arbitrary illumination and albedo. In *Proceedings of European Conference on Computer Vision*, pages 750–767. Springer.
- [48] Kopf, J., Cohen, M. F., Lischinski, D., and Uyttendaele, M. (2007). Joint bilateral upsampling. *ACM Transactions on Graphics*, 26(3):96.
- [49] Kuster, C., Popa, T., Zach, C., Gotsman, C., and Gross, M. (2011). FreeCam: a hybrid camera system for interactive free-viewpoint video. In *VMV*.
- [50] Kuster, C., Ranieri, N., Agustina, Zimmer, H., Bazin, J.-C., Sun, C., Popa, T., and Gross, M. (2012). Towards next generation 3D teleconferencing systems. In *3DTV-CON*.
- [51] Land, E. H. and McCann, J. J. (1971). Lightness and retinex theory. *J. Opt. Soc. America*, 61(1):1–11.
- [52] Langguth, F., Sunkavalli, K., Hadap, S., and Goesele, M. (2016). Shading-aware multi-view stereo. In *Proceedings of European Conference on Computer Vision*, pages 469–485. Springer.
- [53] Li, Y., Xue, T., Sun, L., and Liu, J. (2012). Joint example-based depth map super-resolution. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 152–157. IEEE.
- [54] Lichti, D. (2008). Self-calibration of a 3d range camera. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37(3):927–932.
- [55] Lin, S., You, F., Luo, X., and Li, Z. (2008). Deducing interpolating subdivision schemes from approximating subdivision schemes. In *ACM Transactions on Graphics*, volume 27, page 146. ACM.
- [56] Lindner, M. and Kolb, A. (2007). Calibration of the intensity-related distance error of the pmd tof-camera. In *Proc. SPIE, Intelligent Robots and Computer Vision*, volume 6764, page 67640W.
- [57] Liu, M.-Y., Tuzel, O., and Taguchi, Y. (2013). Joint geodesic upsampling of depth images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 169–176. IEEE.
- [58] Loop, C. (1987). Smooth subdivision surfaces based on triangles. Master’s Thesis, University of Utah.

- [59] Maier, R., Kim, K., Cremers, D., Kautz, J., and Nießner, M. (2017). Intrinsic3d: High-quality 3d reconstruction by joint appearance and geometry optimization with spatially-varying lighting. In *Proceedings of IEEE International Conference on Computer Vision*, volume 4. IEEE.
- [60] Maimone, A., Bidwell, J., Peng, K., and Fuchs, H. (2012). Enhanced personal autostereoscopic telepresence system using commodity depth cameras. *Computers & Graphics*.
- [61] Maimone, A. and Fuchs, H. (2012). Real-time volumetric 3D capture of room-sized scenes for telepresence. In *3DTV-CON*.
- [62] Mateos, G. (2000). A camera calibration technique using targets of circular features. *Department of Computer Science and Systems, University of Murcia*.
- [63] Merrell, P., Akbarzadeh, A., Wang, L., Mordohai, P., Frahm, J.-M., Yang, R., Nistér, D., and Pollefeys, M. (2007). Real-time visibility-based fusion of depth maps. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1–8. IEEE.
- [64] Nehab, D., Rusinkiewicz, S., Davis, J., and Ramamoorthi, R. (2005). Efficiently combining positions and normals for precise 3d geometry. *ACM Transactions on Graphics*, 24(3):536–543.
- [65] Nießner, M., Zollhöfer, M., Izadi, S., and Stamminger, M. (2013). Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics*, 32(6):169.
- [66] Or-El, R., Hershkovitz, R., Wetzler, A., Rosman, G., Bruckstein, A. M., and Kimmel, R. (2016). Real-time depth refinement for specular objects. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 4378–4386. IEEE.
- [67] Or-El, R., Rosman, G., Wetzler, A., Kimmel, R., and Bruckstein, A. M. (2015). Rgb-d-fusion: Real-time high precision depth recovery. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 5407–5416. IEEE.
- [68] Oxholm, G. and Nishino, K. (2012). Shape and reflectance from natural illumination. In *Proceedings of European Conference on Computer Vision*, pages 528–541. Springer.
- [69] Park, J., Kim, H., Tai, Y.-W., Brown, M. S., and Kweon, I. (2011). High quality depth map upsampling for 3d-tof cameras. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1623–1630. IEEE.

- [70] Pomerleau, F., Colas, F., Siegwart, R., and Magnenat, S. (2013). Comparing ICP variants on real-world data sets. *Autonomous Robots*.
- [71] Richardt, C., Stoll, C., Dodgson, N. A., Seidel, H.-P., and Theobalt, C. (2012). Coherent spatiotemporal filtering, upsampling and rendering of rgbz videos. *Computer Graphics Forum*, 31(2pt1):247–256.
- [72] Riemens, A. K., Gangwal, O. P., Barenbrug, B., and Berretty, R.-P. (2009). Multistep joint bilateral depth upsampling. In *SPIE*, volume 7257, pages 336–347.
- [73] Salas-Moreno, R. F., Newcombe, R., Strasdat, H., Kelly, P. H., and Davison, A. J. (2013). Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1352–1359. IEEE.
- [74] Samaras, D., Metaxas, D., Fua, P., and Leclerc, Y. G. (2000). Variable albedo surface reconstruction from stereo and shape from shading. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 480–487. IEEE.
- [75] Scaramuzza, D., Martinelli, A., and Siegwart, R. (2006). A toolbox for easily calibrating omnidirectional cameras. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5695–5701. IEEE.
- [76] Seitz, S. M., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 519–528. IEEE.
- [77] Shoemake, K. (1985). Animating rotation with quaternion curves. In *Proceedings of IEEE and ACM International Symposium on Mixed and Augmented Reality*.
- [78] Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor segmentation and support inference from rgb-d images. In *Proceedings of European Conference on Computer Vision*, pages 746–760. Springer.
- [79] Sinha, S. N., Mordohai, P., and Pollefeys, M. (2007). Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1–8. IEEE.
- [80] Smisek, J., Jancosek, M., and Pajdla, T. (2013). 3d with kinect. In *Consumer Depth Cameras for Computer Vision*, pages 3–25. Springer.
- [81] Song, P., Wu, X., and Wang, M. Y. (2010). Volumetric stereo and silhouette fusion for image-based modeling. *The Visual Computer*, 26(12):1435–1450.

- [82] Song, X., Dai, Y., and Qin, X. (2016). Deep depth super-resolution: Learning depth super-resolution using deep convolutional neural network. In *Proceedings of the Asian Conference on Computer Vision*, pages 360–376. Springer.
- [83] Svoboda, T., Martinec, D., and Pajdla, T. (2005). A convenient multicamera self-calibration for virtual environments. *PRESENCE: teleoperators and virtual environments*, 14(4):407–422.
- [84] Teichman, A., Miller, S., and Thrun, S. (2013). Unsupervised intrinsic calibration of depth sensors via slam. In *Robotics: Science and Systems*. Citeseer.
- [85] Tylecek, R. and Sara, R. (2010). Refinement of surface mesh for accurate multi-view reconstruction. *The International Journal of Virtual Reality*, 9(1):45–54.
- [86] Umlauf, G. (2005). Analysis and tuning of subdivision algorithms. In *Proceedings of the 21st spring conference on Computer graphics*, pages 33–40. ACM.
- [87] Unnikrishnan, R. and Hebert, M. (2005). Fast extrinsic calibration of a laser rangefinder to a camera. *Technical report, CMU-RI-TR-05-09*.
- [88] Vogiatzis, G., Hernandez, C., Torr, P. H., and Cipolla, R. (2007). Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2241–2246.
- [89] Wang, R., Bach, J., Macfarlane, J., and Ferrie, F. P. (2012). A new upsampling method for mobile lidar data. In *IEEE Workshop on the application of computer vision*, pages 17–24. IEEE.
- [90] Wu, C., Wilburn, B., Matsushita, Y., and Theobalt, C. (2011). High-quality shape from multi-view stereo and shading under general illumination. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 969–976. IEEE.
- [91] Wu, C., Zollhöfer, M., Nießner, M., Stamminger, M., Izadi, S., and Theobalt, C. (2014). Real-time shading-based refinement for consumer depth cameras. *ACM Transactions on Graphics*, 33:3.
- [92] Wu, L., Ganesh, A., Shi, B., Matsushita, Y., Wang, Y., and Ma, Y. (2010). Robust photometric stereo via low-rank matrix completion and recovery. In *Proceedings of the Asian Conference on Computer Vision*, pages 703–717. Springer.
- [93] Xu, D., Duan, Q., Zheng, J., Zhang, J., Cai, J., and Cham, T.-J. (2017). Shading-based surface detail recovery under general unknown illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

- [94] Yoshiyasu, Y. and Yamazaki, N. (2011). Topology-adaptive multi-view photometric stereo. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1001–1008. IEEE.
- [95] Yu, L.-F., Yeung, S.-K., Tai, Y.-W., and Lin, S. (2013). Shading-based shape refinement of rgb-d images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1415–1422. IEEE.
- [96] Zaharescu, A., Boyer, E., and Horaud, R. (2007). Transformesh: a topology-adaptive mesh-based approach to surface evolution. In *Proceedings of the Asian Conference on Computer Vision*, pages 166–175. Springer.
- [97] Zhang, C. and Zhang, Z. (2014). Calibration between depth and color sensors for commodity depth cameras. In *Computer Vision and Machine Learning with RGB-D Sensors*, pages 47–64. Springer.
- [98] Zhang, Q. and Pless, R. (2004). Extrinsic calibration of a camera and laser range finder (improves camera calibration). In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2301–2306. IEEE.
- [99] Zhang, R., Tsai, P.-S., Cryer, J. E., and Shah, M. (1999). Shape-from-shading: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–706.
- [100] Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [101] Zhou, Q. Y. and Koltun, V. (2014). Simultaneous localization and calibration: Self-calibration of consumer depth cameras. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 454–460. IEEE.
- [102] Zhou, Z., Wu, Z., and Tan, P. (2013). Multi-view photometric stereo with spatially varying isotropic materials. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1482–1489. IEEE.
- [103] Zhu, J., Wang, L., , Yang, R., and Davis, J. (2008). Fusion of time-of-flight depth and stereo for high accuracy depth maps. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.
- [104] Zitnick, C. L., Kang, S. B., Uyttendaele, M., Winder, S., and Szeliski, R. (2004). High-quality video view interpolation using a layered representation. *SIGGRAPH*.
- [105] Zollhöfer, M., Dai, A., Innmann, M., Wu, C., Stamminger, M., Theobalt, C., and Nießner, M. (2015). Shading-based refinement on volumetric signed distance functions. *ACM Transactions on Graphics*, 34(4):96.