2017

REAL-TIME CAPTURE AND RENDERING OF PHYSICAL SCENE WITH AN EFFICIENTLY CALIBRATED RGB-D CAMERA NETWORK

Po-Chang Su
University of Kentucky, iamsue110@hotmail.com
Author ORCID Identifier:
https://orcid.org/0000-0001-9862-1056
Digital Object Identifier: https://doi.org/10.13023/ETD.2017.478

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Po-Chang Su, Student

Dr. Sen-Ching S. Cheung, Major Professor

Dr. Cai-Cheng Lu, Director of Graduate Studies

REAL-TIME CAPTURE AND RENDERING OF PHYSICAL SCENE WITH AN EFFICIENTLY CALIBRATED RGB-D CAMERA NETWORK

DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Engineering at the
University of Kentucky

By
Po-Chang Su
Lexington, Kentucky

Director: Dr. Sen-Ching S. Cheung, Professor of Electrical and Computer Engineering
Lexington, Kentucky 2017

ABSTRACT OF DISSERTATION

REAL-TIME CAPTURE AND RENDERING OF PHYSICAL SCENE WITH AN EFFICIENTLY CALIBRATED RGB-D CAMERA NETWORK

From object tracking to 3D reconstruction, RGB-Depth (RGB-D) camera networks play an increasingly important role in many vision and graphics applications. With the recent explosive growth of Augmented Reality (AR) and Virtual Reality (VR) platforms, utilizing camera RGB-D camera networks to capture and render dynamic physical space can enhance immersive experiences for users. To maximize coverage and minimize costs, practical applications often use a small number of RGB-D cameras and sparsely place them around the environment for data capturing. While sparse color camera networks have been studied for decades, the problems of extrinsic calibration of and rendering with sparse RGB-D camera networks are less well understood. Extrinsic calibration is difficult because of inappropriate RGB-D camera models and lack of shared scene features. Due to the significant camera noise and sparse coverage of the scene, the quality of rendering 3D point clouds is much lower compared with synthetic models. Adding virtual objects whose rendering depend on the physical environment such as those with reflective surfaces further complicate the rendering pipeline.

In this dissertation, I propose novel solutions to tackle these challenges faced by RGB-D camera systems. First, I propose a novel extrinsic calibration algorithm that can accurately and rapidly calibrate the geometric relationships across an arbitrary number of RGB-D cameras on a network. Second, I propose a novel rendering pipeline that can capture and render, in real-time, dynamic scenes in the presence of arbitrary-shaped reflective virtual objects. Third, I have demonstrated a teleportation application that uses the proposed system to merge two geographically separated 3D captured scenes into the same reconstructed environment.

To provide a fast and robust calibration for a sparse RGB-D camera network, first, the correspondences between different camera views are established by using a spherical calibration object. We show that this approach outperforms other techniques based on planar calibration objects. Second, instead of modeling camera extrinsic using rigid transformation that is optimal only for pinhole cameras, different view transformation functions including rigid transformation, polynomial transformation,

and manifold regression are systematically tested to determine the most robust mapping that generalizes well to unseen data. Third, the celebrated bundle adjustment procedure is reformulated to minimize the global 3D projection error so as to fine-tune the initial estimates. To achieve a realistic mirror rendering, a robust eye detector is used to identify the viewer's 3D location and render the reflective scene accordingly. The limited field of view obtained from a single camera is overcome by our calibrated RGB-D camera network system that is scalable to capture an arbitrarily large environment. The rendering is accomplished by raytracing light rays from the viewpoint to the scene reflected by the virtual curved surface. To the best of our knowledge, the proposed system is the first to render reflective dynamic scenes from real 3D data in large environments. Our scalable client-server architecture is computationally efficient - the calibration of a camera network system, including data capture, can be done in minutes using only commodity PCs.

KEYWORDS: RGB-D Camera Network; Real-time Capture and Rendering; Virtual Curved Mirror, 3D Telepresence, 3D Interaction.

Author's signature: Po-Chang Su

Date: December 10, 2017

REAL-TIME CAPTURE AND RENDERING OF PHYSICAL SCENE WITH AN EFFICIENTLY CALIBRATED RGB-D CAMERA NETWORK

By
Po-Chang Su

Director of Dissertation: Sen-Ching S. Cheung

Director of Graduate Studies: Cai-Cheng Lu

Date: December 10, 2017

For my family

ACKNOWLEDGMENTS

First, I would like to thank my advisor Dr. Sen-Ching Samson Cheung for teaching me how to do research during my PhD. He always inspires me and guides me to do interesting researches. I would also like to thank all the members in MIA lab. They are very supportive and willing to give opinions on my experiments. I would like to thank Dr. Ju Shen and Changpeng Ti for sharing their experiences on doing research in computer vision. I would also like to thank my committee members, Dr. Kevin Donohue, Dr. Ruigang Yang and Dr. Nathon Jacobs for their insightful suggestions on my research. I would like to thank Dr. Janet Lumpp and Dr. William Smith to help me become a better person. Finally, I would like to thank my family for their unconditional support and encouragement. Part of this material is based upon work supported by the National Science Foundation under Grant No. 1237134.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

High quality capturing and rendering of three dimensional objects and environments are important to many applications in medicine, architecture, manufacturing, education, and entertainment. With the recent explosive growth of virtual and augmented reality platforms, such technologies are crucial in delivering realistic experiences to users. Pokemon Go [10], one of the most successful augmented-reality games, infuses virtual objects in the real world and has drawn an unprecedented attention from the media. The game has achieved 100 million downloads in less than one month and has generated over 200 million US dollars in revenue in the same period [11]. While augmented-reality mixed synthetic objects with real-world scene, virtual reality completely immerses users into a virtual environment using VR headsets [12, 13] and has enjoyed an exponential growth in the gaming market [14]. While gaming is the primary genre of these applications, the possibility of capturing, transmitting, and rendering in real-time reconstructed objects and individuals to enhance communication between users in VR has garnered a great deal of excitement [1]. Instead of seeing your friend in a live video feed on a computer, Holoportation supports high-quality 3D models of human to be captured and transmitted anywhere in the world in real-time [2]. Such technologies open door to remote collaboration and manipulation of objects and environments, without confining to the limited size of a computer display as shown in Figure 1.1.

When capturing static or dynamic scenes for different augmented or mixed reality applications, using multiple networked cameras has many advantages over a single

(a) (b)

Figure 1.1: Immersive applications: (a) A corporative game in a virtual environment [1]; (b) Remote users communicate in a shared 3D space [2].

camera. Single camera suffers from nonintuitive, self-occluding hulls when capturing non-convex articulated 3D shapes like human bodies. The field of view and spatial resolutions of a single camera, especially depth cameras, are often limited. Using Simultaneous Localization and Mapping (SLAM) techniques [15–18] with a moving camera can be used to capture a large static environment but does not work for dynamic scenes. On the other hand, using a stationary camera network can address the limitations in both field of view and dynamic scene capturing. There are already a large body of work using color camera networks for various types of vision processing [19–21]. Camera networks based on depth sensors such as Time-of-Flight (TOF) or Structured Light cameras, however, are not as well-explored. Earlier depth cameras suffer from high measurement noise and low spatial resolution. Due to the recent success of low-cost commodity depth cameras such as Kinect and Xtion Pro Live, there have been significant improvements in performance, thanks to better sensing technology and the addition of a companion high-definition color camera [22–24].

(a)

(b)

Figure 1.2: Virtual mirror applications: (a) Virtual try-on [3]; (b) Behavioral Interventions for children with ASD [4].

In fact, utilizing a pair of color and depth cameras has solved a number of challenging problems. For example, missing depth values and depth misalignment on planar surfaces can be recovered by exploiting the co-located color and depth information [3, 25]. It is natural to extend from a single RGB-D camera to a network of RGB-D cameras, which are beneficial to a myriad of applications including 3D model rendering, body pose tracking and understanding [26–28]. However, there are still significant challenges to calibrate RGB-D camera networks and they will be discussed in Section 1.1.1.

The capability of combining aligned 3D data with virtual objects can make the rendering far more realistic than current AR/VR systems. Currently, physical objects and scenes rendered in virtual world are fairly limited and lack the realism commonly

seen in full computer-generated imagery (CGI) contents. One reason is that reflections of objects on surfaces from different worlds are not accurately rendered. Imagine that a potential home buyer could be rendered on the many reflective surfaces in a new house, giving a far more realistic impression of actually being there. In fact, diverse applications from tourism to manufacturing can benefit from such a technology: from visiting the Hall of Mirrors in the Palace of Versailles or the Cloud Gate at the Millennium Park in Chicago to designing the reflective body of the next generation automobile. In my dissertation, I consider the problem of rendering reflective scene from physical environment on virtual specular surfaces of arbitrary geometry. In fact, the work can be considered as an extension of other simulated-mirror-based systems (SMD), in which a virtual flat mirror is simulated by rendering the mirror image of physical scene, with possible addition of different virtual objects. As shown in Figure 1.2, SMDs are used in virtual try-on systems [3, 29, 30], education [31, 32], health [33, 34], art [35], and even in behavioral interventions for young children with autism spectrum disorders (ASD) [4].

1.1 Problem Statement

In my dissertation, I study two problems in building the next generation 3D environment capturing and rendering systems. The first problem is on how to accurately and rapidly calibrate a network of sparsely-placed RGB-D cameras. The second problem is on how to create virtual specular surfaces that can dynamically render reflection of physical environments. For the first problem, I have developed novel algorithms that only rely on small overlap in fields of views from different camera,

(a) (b) (c)

(d) (e)

Figure 1.3: Capturing a large scene by a network of RGB-D cameras. Cameras are sparsely placed with different poses.

and can achieve global minimization of 3D re-projection error over the entire network. For the second problem, I have built novel real-time realistic rendering system of the reflection of physical scenes over virtual curved mirrors. The physical dimension of the captured space are significantly expanded comparing early works in [3, 29, 36].

1.1.1 Sparse RGB-D Camera Network

The prerequisite in using multiple cameras is to obtain the geometrical relationship that fuses the individual camera perspectives into a unified space. An accurate and robust calibration of multiple cameras is essential for applications that require large-scale captured scene. However, it is challenging to calibrate a network of sparsely-distributed RGB-D cameras as shown in Figure 1.3. There are three main challenges

to this task: first, the captured data from depth cameras often have missing and noisy measurements, particularly on transparent or specular surfaces, and near depth discontinuities. These imperfections can greatly deteriorate the accuracy of the geometric alignment. Second, the sparsity of cameras makes it difficult to locate common scene features needed for calibration across disparate camera views. Adjacent camera views may share more scene features but even small alignment error between adjacent views could accumulate when it is extrapolated to the entire network. Finally, as multiple cameras are often cumbersome to setup and maintain, it is highly desirable to make the calibration procedure robust and easily adaptable to any changes in the camera placement. There have been a number of recent works on RGB-D network calibration [27, 37–39], but as we shall point out in Chapter 2, these approaches are either impractical or prone to errors. To address the problems mentioned above, I propose a fast and robust algorithm for calibrating a sparse network of multiple RGB-D cameras by a spherical object in Chapter 3. Using only commodity hardware, the entire calibration process of a camera network takes only minutes to complete.

1.1.2 Real-Time Rendering of Physical Scenes with Virtual Mirrors

The prerequisite in simulating a reflective surface is to render the visual contents based on a user's perspective. To simulate a large reflective surface that can cope with wide displacement of a viewer, a camera-display system must be able to capture the 3-D world, track the moving viewpoint, render new views and possibly add new visual contents that are compatible with the scene geometry. The depth information provided by the RGB-D cameras can be used to track the user's viewpoint and render

(a)

Figure 1.4: The scene reflected by the curved mirror [5]

the scene dynamically. However, there are a number of technical challenges in the remaining parts of the pipeline. First, as we have mentioned in Section 1.1.1, missing and noisy measurements from depth cameras and limited field of view make rendering large areas and familiar shapes difficult. Second, while a planar reflective surface can be simply reduced to moving the view point to its mirror image [3], curved mirrors, as shown in Figure 1.4, are significantly more difficult due to the aberration effect in which reflected light rays from a single scene point do not converge into an image point. To the best of our knowledge, there are no prior work in modeling and demonstrating real-time reflection of large-scale physical environments on virtual reflective surfaces. To achieve this goal, I present a virtual mirror system that can render physical scenes in real-time on a virtual curved mirror in Chapter 4. Unlike other mirror systems that only have a fixed viewpoint and limited field of view, the proposed framework is capable of rendering reflective scenes based on a viewer's

perspective.

1.2 Contributions of Dissertation

The research work presented in this dissertation addresses the challenges of calibrating a network of sparsely-placed RGB-D cameras and develops a novel system of rendering large 3D physical scene in real-time for virtual curved mirror. The main contributions of my work are as follows:

1. Unlike other approaches that rely on planar calibration objects, the usage of a spherical object overcomes the problem of limited scene features shared by sparsely placed cameras. Specifically, an effective sphere-fitting algorithm is used to identify the moving locations of the sphere center in both the color and depth images. The sphere center can be robustly estimated from any viewpoint as long as a small part of the sphere surface can be observed.

2. Rigid transformation is typically used to represent camera extrinsic and has been shown to be the optimal for the pinhole camera model. An initial estimate of the extrinsic is obtained based on the corresponding locations across different views. The obtained extrinsic is further refined using the simultaneous optimization on the entire network. However, real cameras have imperfection and a more flexible transformation could provide higher fidelity in aligning 3D point clouds from different cameras. I systematically compare a broad range of transformation functions including rigid transformation, intrinsic-extrinsic factorization, polynomial regression, and manifold regression. The experiments

demonstrate that linear regression produces the most accurate calibration results.

3. To provide an efficient calibration procedure and to support real-time 3D rendering and dynamic viewpoints, the proposed algorithm is implemented in a client-and-server architecture where data capturing and part of the 3D processing tasks are carried out at the clients. The architecture is capable of transmitting and processing large volume of RGB-D data across the network so as to reconstruct dynamic large 3D scenes, which are synthesized by seamlessly fusing multiple video streams acquired by different cameras. Based on the designed client-and-server architecture in our RGB-D camera network, the proposed system can bring users into the same, large reconstructed environment by merging geographically separated 3D captured scenes.

4. To achieve a realistic mirror rendering, the proposed system uses eye tracking to capture the viewpoint of the user, and simulate the mirror effect by tracing light rays from the viewpoint to the reconstructed 3D scene after reflecting off the reflective surface. To address the limited field of view issue from a single camera, a calibrated RGB-D camera network is built to provide richer 3D information for reflective scenes rendering. The proposed system is demonstrated through a series of experiments to show the accuracy of the calibration and the quality of the virtual mirror rendering. To the best of our knowledge, the proposed system is the first to render real-time mirror-like reflective scenes from real 3D data in large environments.

1.3 Dissertation Organization

The rest of this dissertation is organized as follows: Chapter 2 reviews previous works of multi-camera calibration, virtual mirror simulation and telepresence. In Chapter 3, I describe in details the proposed system which includes sphere center detection, pairwise camera calibration, and the simultaneous optimization for the RGB-D camera network. A thorough comparison of different view transformations and incorporate a global optimization procedure in refining the results is provided. In Chapter 4, I describe the proposed framework of rendering reflective scenes on a virtual curved mirror in the reconstructed 3D environment by our calibrated RGB-D camera network. The Ray tracing technique to render correct dynamic scenes on virtual mirror surfaces is utilized. In the following section, an application based on remote 3D interaction in our RGB-D camera network system is implemented. Finally, the dissertation is concluded and the future work is discussed in Chapter 5.

Chapter 2 Literature Review

In this chapter, we review previous works related to our proposed physical scene capture and rendering system. The works are categorized into three areas: multi-camera calibration, virtual mirror rendering, and telepresence.

2.1 Multi-Camera Calibration

Extrinsic calibration requires a calibration object visible to different cameras in order to establish correspondences. For color camera calibration, commonly used calibration objects include planar checkerboards [19, 40–42], laser pointers [43–45], circular patterns [46, 47], planar mirrors [48], and other custom-made objects [49]. None of these calibration objects work for depth sensors as they rely on distinctive colors or texture patterns that are not observable to depth sensors. Additionally, they require a dense camera network to obtain accurate camera extrinsics. As such, their calibration procedures tend to be time-consuming and are unable to restore the calibration rapidly in a volatile environment, where cameras may be added, moved, or removed. Instead, objects with significant depth variations need to be used to calibrate depth sensors. For example, a planar calibration pattern with holes were used in [50]. Planar objects were also used by Herrera et al. who utilized the four corners of the calibration plane [51]. Liu et al. instead used a moving stick with one end fixed for RGB-D camera calibration [52]. A common drawback of these approaches is that the sharp depth edges along these objects usually have significant measurement noise on the depth images. Such noise can lead to erroneous correspondences across

different views.

In [6], the authors of [51] improved their earlier scheme by using the planarity constraint defined based on the corners of a checkerboard plane. The use of planar, instead of point, features alleviates the problem associated with depth discontinuities. Similar approaches could also be found in [53] and [54], both of which used planarity constraints to detect the correspondences between the depth images. However, the calibration accuracy is still poor due to the low spatial resolution of depth cameras. To improve the accuracy, Shim et al. used the corresponding 3D positions, rather than 2D features, to optimally calibrate multiple RGB-D cameras [23]. Specifically, they identified the two major sources of depth measurement error to be the changes in scene depth, and the amount of captured infrared light. Based on these two factors, they constructed an error model to optimize the calibration results. On the other hand, the authors did not address the issue of limited common scene features when the cameras are sparsely spaced.

Calibrations without using any specialized reference objects or patterns have also been studied [55–57]. In [55], silhouette extracted from a person was used for calibration. In [56], Carrera et al. calibrated a robotic camera platform by detecting invariant SURF feature correspondences across different views. In [57], the extrinsic was estimated based on point correspondences established from the unstructured motion of objects in the scene. These methods typically have lower precision than those based on reference objects due to imprecise knowledge of the unknown scene features, which can lead to erroneous correspondences from different viewpoints. In [58], Li et al. proposed a method to calibrate multiple cameras based on users joint positions.

The calibration process can be accomplished by aligning skeleton data across different camera views. However, their system has the difficulties of fusing noisy skeleton data from wide-baseline camera network setup.

Besides calibration objects, another key difference between RGB-D cameras and color cameras is the availability of both color and depth information in RGB-D cameras. Appropriate fusion of multiple data channels can potentially achieve more accurate depth measurements and extrinsic calibration [22, 24, 59]. Prasad et al. first demonstrated depth resolution enhancement through color and depth registration by using a novel system with a 2D sensor, a 3D sensor, and an image multiplier [59]. In [22], reference depth images generated by a pair of stereo cameras were used to calibrate a Time-of-Flight (ToF) depth sensor. The depth image quality can also be improved by utilizing both active and passive depth measurements. Hansard et al. used 3D projective transformation to calibrate both the ToF and color cameras [24]. The geometric relation could then be found by aligning range images with parallax reconstructions.

With the advent of low-cost commodity RGB-D cameras, there are now software libraries that can easily align the depth image to the color image for a RGB-D camera [60]. However, a common assumption of all these methods is a close baseline among different cameras so that the transformation among different views can be easily accomplished. Based on [60], the works in [27, 37, 39] calibrated a RGB-D camera network to reconstruct 3D objects. Nevertheless, none of them can accurately reconstruct an entire 3D scene including static background and dynamic foreground as they utilized checkerboard and iterative closest point algorithm to align dense

point clouds of foreground objects. The reconstructed scenes would be misaligned based on their proposed methods.

Increasingly, the topic of calibration of wide-area networks of sparsely-spaced cameras has been investigated [21, 61–63], though the majority of the techniques require special equipments and image features. Kuo et al. used GPS position and images taken by mobile devices to calibrate a fixed camera in a camera network [21]. Ly et al. utilized image of lines to improve the calibration results for multiple cameras with only partially overlapping fields of view [61]. In [62], an active self-calibration of a multi-camera system scheme was proposed to solve the problem of non-overlapping views and occlusion by automatically rotating and zooming each camera. A probabilistic model was used to find the appropriate relative pose during extrinsic calibration. In [63], the authors used large planer scenes such as the floor or ceiling to calibrate cameras with disparate views. The use of pan-tilt-zoom cameras or special scene features limits the types of applications where these techniques can be deployed.

Recently, there have been a number of works using spherical objects for multi-camera calibration [7, 38, 64–66]. In [65, 66], the authors have shown that using a spherical object for calibration could produce better results than using the traditional checkerboard as in [54]. However, the comparisons were done only for cameras separated by a narrow baseline. Yang et al. setup a dense camera network to reconstruct the 3D scene in a small area [64]. The ICP algorithm is utilized to optimize camera extrinsics. However, the comparisons were done only for cameras separated by a narrow baseline. Yang et al. reconstructed the 3D scene in a small area by a dense camera network [64]. The ICP algorithm is utilized to optimize camera extrinsics.

However, the proposed method did not demonstrate live captured 3D data including background can be well extrapolated in large environments. For sparse camera networks, Ruan et al. used a spherical object and estimated the location of the sphere center for extrinsic calibration of multiple depth cameras [38]. However, the sphere detection was not very robust because color information was not used. Furthermore, as the cameras were not time-synchronized, the technique was labor intensive as the sphere needed to be physically moved and affixed to different locations in order to capture enough data for calibration. In [7], we independently proposed a RGB-D camera network calibration scheme based on sphere center detection. Using both the color and depth channels, we developed an automatic noise removal algorithm to robustly identify the sphere and estimate its center location. As our cameras were time-synchronized, a user could simply waive the sphere in the environment once and the data collection process would be done. A drawback of [7] is in its reliance on the simplistic rigid transformation based pairwise camera registration, which is inadequate for non-pinhole cameras and can lead to error accumulation. In this dissertation, we extend the scheme in [7] by first using a more flexible view transformation function to minimize error in registration and then introducing a simultaneous optimization framework to further refine the extrinsic parameters. Comparison of the proposed scheme with our earlier work and other state-of-the-arts schemes discussed here can be found in Section 3.4.

2.2 Virtual Mirror Rendering

Current virtual mirror rendering systems commonly mount a camera near to the display [67–73]. A system that can allow customers to try on virtual customized shoes combining with captured images to increase realistic effects was presented in [67–69]. Jimeno-Morenilla et al. proposed a stereoscopic vision system for users viewing 3D virtual shoes on their feet [68]. Yang et al. developed a two-stage object tracking method to align virtual shoes with moving feet accurately [69]. Similar systems can be found in [70–73], where people can virtually try different clothes and handbags based on virtual mirror rendering on a display. Saakes et al. proposed an interactive system that users can design and put on the virtual clothes on-the-fly in front of the virtual mirror [72]. A projector was used to project the virtual clothes on the users to achieve high-fidelity feedback. In [73], the authors proposed an intelligent virtual mirror system to customize users clothes based on fashion trend and personal preference. The system is capable of learning users preference of wearing and giving recommendations to each individual. However, all of the aforementioned systems cannot generate virtual view with arbitrary perspective for users. To provide free viewpoint functionality, the authors proposed a system that allowed a user to view himself or herself in 360 degree [29, 74, 75]. Multiple RGB cameras were mounted around the user for capturing the movement of human body. However, the background was not rendered on the display, which lacked the essential characteristic of a mirror where the reflective scenes changed based on the user's viewpoint.

Recently, Ju et al. used multiple RGB-D cameras placed in front of a viewer to

render reflections in a virtual plane mirror [3]. While their proposed system can render reflective scenes based on the viewer's perspective, the rendering results on the facial part is poor due to the misalignment of camera views and lack of surface meshing. Also, for a specular surface with arbitrary geometry, there are few viewpoints at which a single perspective image can be obtained [76]. Specifically, light rays emanating from a scene point may be reflected through multiple paths towards the viewpoint, resulting in a so-called multi-viewpoint image. For specific configurations such as a spherical mirror, the center is the only point that can produce a single viewpoint image. As such, using a virtual pinhole camera at the viewpoint to render the mirror image as described in [3] is no longer applicable.

Aside from planar mirror systems, curved mirror surface has also received wide attention among researchers. Techniques to render curved mirror reflections can be generally classified into two groups: graphics based visualization and light field imaging. The former focused on rendering realistic 2D views of curved object reflection of well defined 3D scenes (i.e. geometry, reflectance properties and textures) [77]. The later built novel sensing devices to enhance the capture space by using convex shape mirrors to widen the field of view [78], such as the applications in robotic navigation [79] and omni-directional stereo [80]. Different from previous work, this dissertation has two unique research goals: first, we aim at simulating the visual effect of curved surface reflection rather than using a real curved mirror for capturing; second, instead of using handcrafted 3D models for visualization, we utilize sensing devices to acquire the real world scene as the input source for dynamic curved mirror view rendering. These two key features can benefit many VR and AR applications,

such as teleconferencing, where users can perceive each other in a broader environment with real world illumination [81].

It is not an easy task to achieve curved mirror simulation in the real scene. Compared with the planar mirror, curved mirror has different optical properties. The reflection largely depends on the surface curvature, which does not yield perspective view transformation [82, 83]. To obtain correct reflective scenes, the authors treated each point of the curved mirror as a small planar mirror tangent to the surface and generate independent reflected rays [84–90]. High-fidelity rendering for reflective scenes with curved surfaces can be achieved by using thousands of triangle meshes. In particular, ray tracing based methods can be adopted and often used for mixed reality applications [88–90]. However, a limitation of those works is the failure of considering user's viewpoint change in a dynamic environment. Miguel et al. proposed a forward projection method to efficiently estimate a vertex's reflection point on the quadric surfaces [91]. However, they only demonstrated reflective scenes can be accurately rendered on a sphere shape surface. The global scenes in these works are either handcrafted 3D model or small area captured by a single camera. Their proposed systems cannot generate large-scale real scenes captured by RGB-D cameras – the captured data are often noisy with missing surfaces and the alignments are imperfect due to uncertainty associated with camera pose. The addition of the capturing subsystem also presents significant challenges in maintaining the real-time performance as demanded by an interactive mirror experience. The proposed virtual mirror system is capable of simultaneously capturing and rendering dynamic reflection from a virtual arbitrarily-shaped curved mirror under the real-time constraint. The

multiple well-calibrated RGB-D cameras can cover the entire physical environment for accommodating the movement of the viewer and a significantly convex mirror.

2.3 Telepresence

Current multimedia technologies are transforming the way we communicate and interact in our daily routines. Video chat is a typical example that allows people to have remote conversations and be able to see each other together with the surrounding environments. While 2D video is efficient for remote conversation with the basic communication requirements, it may not be sufficient for activities which demand high engagement and interaction, such as virtual classroom [92] or immersive gaming [93]. As an emerging technique, telepresence aims to integrate spatial immersion and virtual realty to create a perception of being physically present in a shared non-physical environment. To date, many telepresence systems are proposed to offer immersive and realistic experiences to end-to-end users. For example, a telepresence robot is designed to enable teachers to deliver motivating lectures remotely [94]; dynamic scene generation based on perspective views can make the co-space experience more engaging and absorbing [95]. To enhance the awareness of co-location of participants in the same virtual space, audio-visual data and other stimuli are often employed. Therefore, high-speed video transmission and real-time data visualization are crucial to provide immediate feedback [67, 96, 97] for remote 3D interaction system. However, creating the visual perception that dynamically synthesizes realistic virtual environment from the first-person view in real-time has not yet well addressed.

While current telepresence systems produce reasonable 3D interactive experience,

there is still a great potential of improving the rendered visual effects to deliver a more realistic sense of face-to-face interaction. Some of existing work only provide a 3D image with a fixed viewpoint on a 2D display without the ability to offer arbitrary perspective view [98–100]. Such limitation can significantly affect full interaction between users. Early attempts on incorporating user's viewpoint in telepresence systems only put the users in graphics-rendered virtual environment instead of putting them in real scenes [92, 101–103]. To implement a telepresence system that can render real 3D scenes, Maimone et al. built a RGB-D camera network for capturing and reconstructing 3D data [95]. The rendering of visual contents changes based on the viewer's perspective. However, the proposed system only reconstruct a partial region of the environment and users from different places cannot be put into the same environment. To achieve high-quality 3D telepresence, either a cluster of cameras or 3D projectors can be used for the environmental setup [100, 104–106]. However, such hardware installment is expensive and not feasible for general users. By incorporating a network of sparsely placed commodity RGB-D cameras, the proposed 3D telepresence system can reconstruct large dynamic 3D scenes for geographically separated users without compromising to low quality output. With the prevalence of 802.11ad wireless standard, the data transmission in the proposed system can be carried out in real-time.

Chapter 3 RGB-D Camera Network Calibration

The use of multiple RGB-depth cmaeras can capture a wider field of view for acquiring more dynamic 3D information in large environment. In this chapter, we describe in details our RGB-D camera calibration system which includes sphere center detection, pairwise camera calibration, and the global optimization for the camera network. A thorough comparison of different view transformations and incorporated the simultaneous optimization procedure in refining the results is provided.

3.1 Overview

The block diagram in Figure 3.1 shows the basic architecture and data flow of our proposed framework. Each RGB-D camera is controlled by a client process. The server process, which can be run in the same computer as the clients or a separate computer on the network, collect all necessary information from the clients to compute the extrinsic parameters. All the client processes and the server process are time-synchronized using the Network Time Protocol (NTP) with time drift less than 4 milliseconds [107]. While the accuracy of the extrinsic parameters could be measured with respect to ground-truth data, the ultimate test is on how well they can contribute to the 3D reconstruction of real-world scene beyond the ground-truth set. As such, our architecture is designed with this goal in mind and supports real-time 3D rendering – color and depth images are compressed and streamed to the server, which can use the previously computed extrinsic parameters to perform 3D reconstruction in real time. Our framework applies to camera networks consisting of depth cameras and/or

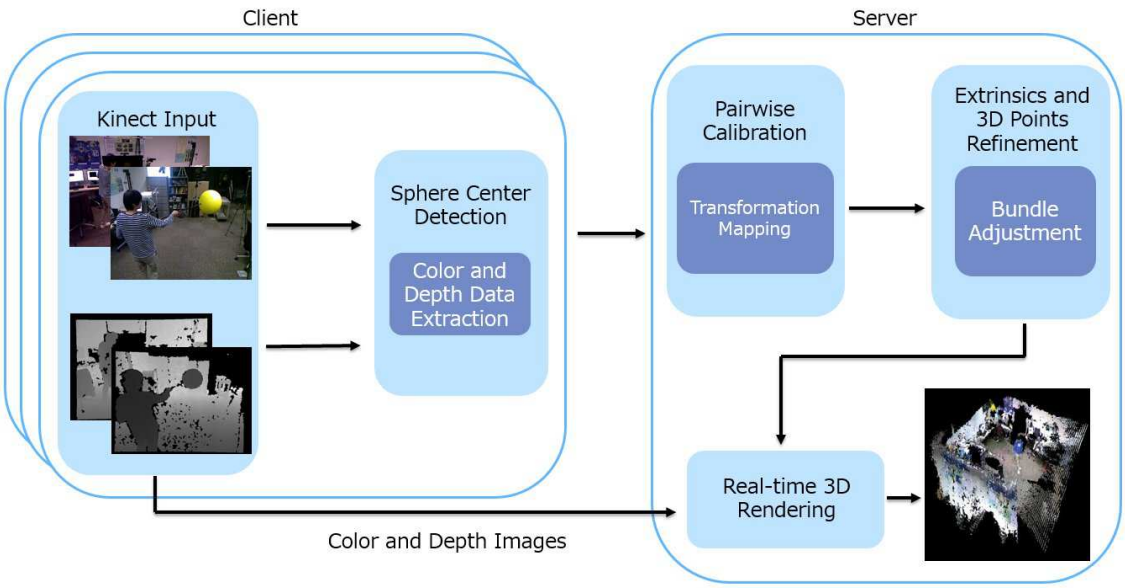


Figure 3.1: This figure provides an overview of our RGB-D cameras calibration framework for real-time 3D rendering in a client-server distributed architecture. On the client side, each of the Kinect camera clients produces a pair of color and depth images. The sphere center detection module uses these raw data to estimate the location of sphere center. During calibration, the estimates are sent to the server, which produces first an rough estimate of the extrinsic camera matrices and the sphere center locations in the world coordinate system. The results are then refined by a simultaneous optimization process to produce optimal extrinsic matrices, which are used to produce real-time rendering results.

co-located depth-color cameras such as Kinect cameras. The main functional blocks in Figure 3.1 are as follows:

Sphere Center Detection: The 3D locations of the center of a moving sphere are estimated from the color and depth images. They are used as visual correspondences across different camera views. There are two reasons for choosing a sphere as a calibration object. First, it is suitable for wide baseline: any small surface patch on the sphere is sufficient to estimate the location of its center. As such, two cameras capturing different sides of the sphere can still use the sphere center as a correspondence. Second, instead of using the error-prone

point or edge features as correspondences, depth measurements of the sphere surface are mostly accurate and the spherical constraint can be used to provide a robust estimate of the center location. This step is independently executed at each camera client. The details of the procedure can be found in Section 3.3.1.

Pairwise Calibration: To provide an initial estimate of the extrinsic parameter of each camera, we perform pairwise calibration to find the view transformation function from each camera to an arbitrarily-chosen reference coordinate system. The server receives from each client the estimated sphere center locations and the associated time-stamps. Correspondences are established by grouping measurements from different cameras that are collected within the time synchronization error tolerance. Then, a system of equations with all correspondences as data terms and parameters of the view transformations as unknowns are solved at the server to provide an initial guess of the transformation functions. Details of this step can be found in Section 3.3.2.

Simultaneous Optimization: The estimated view transformations are then used to bootstrap a pseudo bundle adjustment procedure. This procedure simultaneously adjusts all the extrinsic parameters and the true 3D locations of the sphere center so as to minimize the sum of 3D projection errors across the entire network. Details of this step can be found in Section 3.3.3.

3.2 Problem Formulation

Before we delve into the details of each component, this section formulates the problem of extrinsic calibration of RGB-D camera network and define all the symbols used in this dissertation. A RGB-D sensor consists of a color camera and a depth camera. We first formalized the color camera projection process. Using the coordinate system at the optical center of the color camera as reference, we denote a 3D scene point as $\boldsymbol{X}_c = [X_c, Y_c, Z_c, 1]^T$. The subscript c indicates the usage of the color camera's coordinate system. The color camera project process is modeled by a 3×3 camera projection matrix $\boldsymbol{K}_c$ and a scalar distortion function $L_c(\cdot)$. Specifically, $\boldsymbol{X}_c$ is projected onto the image coordinate $\boldsymbol{x}_c$ on the color camera plane as follows:

$$\boldsymbol{x}_c = \boldsymbol{K}_c \cdot L_c\left(\left\|\begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \end{bmatrix}\right\|\right)\begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \\ 1 \end{bmatrix} \tag{3.1}$$

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

$$\boldsymbol{K}_c = \begin{bmatrix} f_x & \gamma & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

based on the intrinsic parameters of the camera including the focal lengths (f_x, f_y), the principal point (o_x, o_y), and the skew factor γ. $L_c(\cdot)$ is a scalar distortion function that models the radial distortion of the lens, typically expressed as a sixth-degree polynomial [108]. Methods to obtain these intrinsic camera parameters are well documented [40].

The depth camera model projects the 3D scene point $\boldsymbol{X}_d = [X_d, Y_d, Z_d, 1]^T$ with respect to its local coordinate system to two components: the 2D image coordinates $\boldsymbol{x}_d = [x_d, y_d, 1]^T$ and the depth measurement z_d. For the 2D image coordinates, the

projection process is similar to that of the color camera as described in Equation (3.1):

$$\boldsymbol{x}_d = \boldsymbol{K}_d \cdot L_d \left(\left\| \begin{bmatrix} X_d/Z_d \\ Y_d/Z_d \end{bmatrix} \right\| \right) \begin{bmatrix} X_d/Z_d \\ Y_d/Z_d \\ 1 \end{bmatrix} \tag{3.3}$$

with its own camera matrix $\boldsymbol{K}_d$ and distortion function L_d. The depth measurement is related to the actual depth based on the following model:

$$z_d = \frac{1 - \alpha_1 Z_d}{\alpha_0 Z_d} \tag{3.4}$$

where α_0 and α_1 are the parameters that correct depth measurement [109]. To fuse the color and depth information, the color camera and the depth camera need to be calibrated in order to obtain the transformation $\boldsymbol{P}_d$ between the two coordinate systems:

$$\boldsymbol{X}_c = \boldsymbol{P}_d \boldsymbol{X}_d \tag{3.5}$$

$\boldsymbol{P}_d$ is pre-computed using the method in [110].

Consider a network of m RGB-D cameras $\{C_1,\ C_2,\ ...,\ C_m\}$. The goal of the extrinsic calibration is to transform between the local coordinate system of each camera and an arbitrarily chosen world coordinate system. Without loss of generality, we choose the coordinate system of the color camera C_1 to be our world coordinate system. To allow a broad range of extrinsic transformations, we consider the following formulation of the mapping between the 3D point $\boldsymbol{X}_w$ in world coordinates to the 3D point $\boldsymbol{X}_d^{(j)}$ in the j^{th} local depth camera coordinates:

$$h(\boldsymbol{X}_d^{(j)}) = \boldsymbol{P}_j \boldsymbol{X}_w \quad \text{for } j = 1, \ldots, m. \tag{3.6}$$

$\boldsymbol{P}_j$ is the extrinsic matrix for the j^{th} depth camera and $h(\cdot)$ is a data-independent feature mapping that can introduce higher order terms to provide a potentially better fit

of the data. The specific types of $\boldsymbol{P}_j$ and $h(\cdot)$ tested in this dissertation are described in Section 3.3.2. The usages of Equation (3.6) in analysis and synthesis are different. During the analysis stage, we have multiple observations $\boldsymbol{X}_d^{(j)}$ from different cameras of an unknown 3D point $\boldsymbol{X}_w$. The goal is to estimate $\boldsymbol{P_j}^{-1}h(\cdot)$ so as to minimize the overall discrepancies after projecting the image points onto the same world coordinate system. During the synthesis stage, we reverse the above process by using the estimated $\boldsymbol{P_j}^{-1}h(\cdot)$ to project a known 3D point $\boldsymbol{X}_w$ onto each local coordinate system. If the mapping $\boldsymbol{P_j}^{-1}h(\cdot)$ is not invertible, its Moore-Penrose pseudoinverse, denoted as $h^{\dagger}(\boldsymbol{P}_j\cdot)$, will be used. For example, we can compute the color information by relating the local 3D point $\boldsymbol{X}_c^{(j)}$ in the j^{th} color camera coordinates to $\boldsymbol{X}_w$ using the following formula:

$$\boldsymbol{X}_c^{(j)} = \boldsymbol{P}_d^{(j)} h^{\dagger}\left(\boldsymbol{P}_j \boldsymbol{X}_w\right) \quad \text{for } j = 1, \ldots, m. \tag{3.7}$$

Equations (3.1), (3.3), (3.4), (3.6), and (3.7) altogether describe the relationship between an image point $(\boldsymbol{x}_c^{(j)}, \boldsymbol{x}_d^{(j)}, z_d^{(j)})$ in each of the RGB-D cameras and a 3D scene point $\boldsymbol{X}_w$ in the world coordinate system. The problem of extrinsic calibration can now be formulated as follows: *using multiple* $\boldsymbol{X}_w$ *and their corresponding camera image points* $\{(\boldsymbol{x}_c^{(1)}, \boldsymbol{x}_d^{(1)}, z_d^{(1)}), \ldots, (\boldsymbol{x}_c^{(m)}, \boldsymbol{x}_d^{(m)}, z_d^{(m)})\}$ *to optimally compute the extrinsic matrices* $\boldsymbol{P}_j$ *for* $j = 1, 2, 3, \ldots, m$.

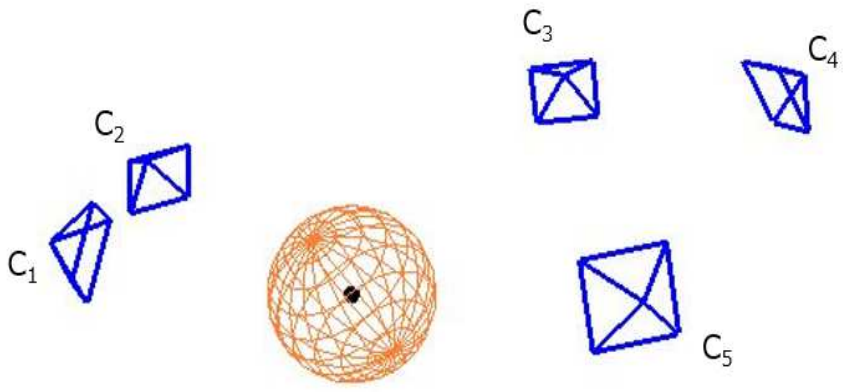


Figure 3.2: Sphere center detection in a camera network

3.3 Proposed Method

3.3.1 Sphere Detection by joint color and depth information

The prerequisite for solving the extrinsic calibration problem as described in Section 3.2 is to establish the correspondence between an image point from each camera and a 3D point in the physical space. Our proposed system uses the center of a spherical calibration object as the target 3D point for calibration. Figure 3.2 illustrates a sphere in the camera network and Figure 3.3 shows our sphere detection process. While the sphere center is not directly visible to any camera, the spherical constraint implies that the observation of a reasonably-size surface patch from any direction can be used to deduce the location of the center. In this section, we describe the algorithm in identifying the calibration object and estimating its center from the captured color and depth images.

To facilitate the detection of the sphere in the color channel, it is painted with a highly distinctive color (see the top row of Figure 3.3). To minimize the effect

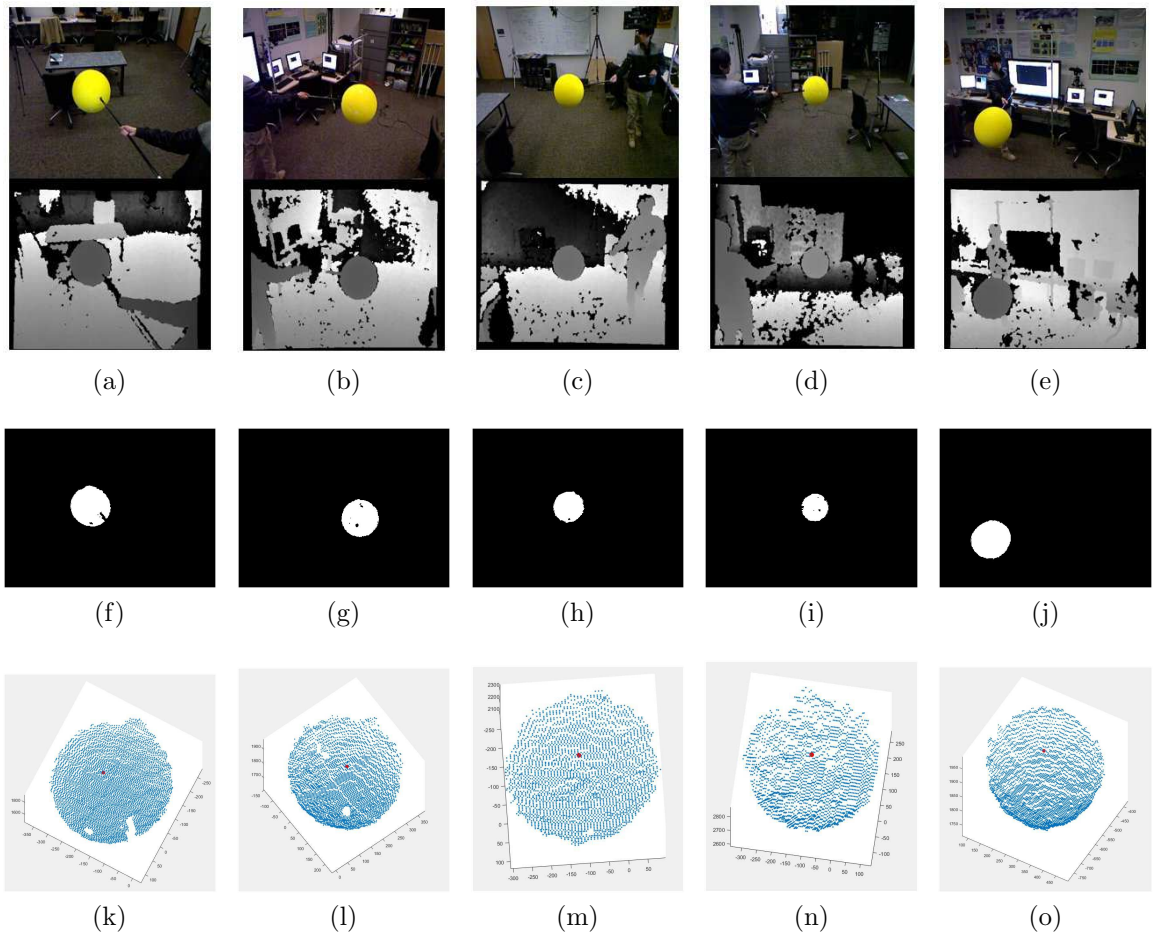


Figure 3.3: Sphere Center Detection: each column shows the process at a different camera in the network. The top row images 3.3a-3.3e are the input RGB and depth images. The middle row images 3.3f-3.3j show the results of detected sphere regions and the bottom row images 3.3k-3.3o represent the initial detected spheres in 3D with the red dots indicating the estimated sphere centers.

of varying illumination, we first convert the RGB image into HSV color space and detects the specific color using a pre-trained Gaussian mixture model classifier in the hue-saturation space. The real-time detection of the sphere is further aided by using a simple background subtraction, and focusing the color search within the foreground region.

The detected color pixels of the sphere are then mapped to the corresponding

depth pixels based on the intrinsic alignment between the two modalities as stated in Equation (3.5). Combining the spatial coordinates and the depth value, we can invert Equations (3.3) and (3.4) to obtain the local 3D coordinates of the detected sphere surface points. As pointed out in Section 1.1.1, depth measurements could be quite noisy. In order to obtain a robust estimate of the center location based on these noisy measurements, we apply a RANSAC procedure by iteratively identifying all the 3D points that satisfy the surface equation of a 3D sphere of a known radius $\bar{r}$ [111]. Specifically, we compute the sphere equation, parameterized by A_1, A_2, A_3, and A_4, by carrying out the following constrained optimization:

$$\min_{A_1 \cdots A_4} \sum_k (x_k^2 + y_k^2 + z_k^2 + A_1 x_k + A_2 y_k + A_3 z_k - A_4) \tag{3.8}$$

subjected to the constraint:

$$\left| \sqrt{(A_1^2 + A_2^2 + A_3^2)/4 - A_4} - \bar{r} \right| \leq \epsilon \tag{3.9}$$

where ϵ is a pre-defined error tolerance in the radius measurement. The estimated sphere center is given by $(-A_1/2, -A_2/2, -A_3/2)$. This estimation is highly robust in our setup for a number of reasons. First, the noisy depth measurements tend to concentrate around the edge of the sphere. However, this has little effect on the estimation of the sphere center location as it is an isotropic quantity. Second, we have chosen a large enough sphere (radius $>$ 100 mm in a 5m $\times$ 5m room) so that the RANSAC procedure typically retains more than a thousand pixels per camera frame for the estimation. Even with a fair amount of occlusion, we have more than sufficient data points to solve for the optimization problem which has only 4 degrees of freedom. The initial detected spheres in 3D with the estimated sphere centers

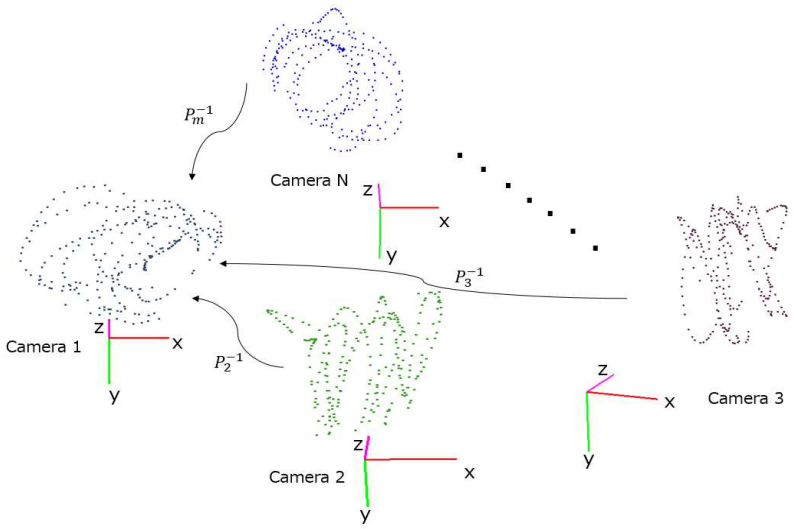


Figure 3.4: Pairwise Calibration: Camera 2 to Camera N are aligned to the reference coordinate (Camera 1)

are shown in Figure 3.3k-3.3o. Repeating the same procedure for n video frames, we obtain the trajectory $\{\boldsymbol{c}_1, ..., \boldsymbol{c}_n\}$ of the estimated sphere centers in the local 3D space.

3.3.2 Extrinsic between Pairwise Cameras

After the locations of the moving sphere center are detected at each camera, we can use them as correspondences to estimate the extrinsic parameters between each camera and the reference camera frame as illustrated in Figure 3.4. The focus on a pair of camera simplifies the optimization but is likely to be suboptimal. As such, this step only produces an initial estimate of the extrinsic parameters which will be later refined in Section 3.3.3. While all cameras are time-synchronized, the sphere may not be simultaneously visible to both cameras in question. Thus, the first step is to filter out those frames in which the sphere is visible to one but not the other. We denote the filtered, time-synchronized trajectories of sphere center

locations in local 3D coordinates at camera pair C_r and C_q as $\{\boldsymbol{C}_1^{(r)}, \boldsymbol{C}_2^{(r)}, ..., \boldsymbol{C}_n^{(r)}\}$ and $\{\boldsymbol{C}_1^{(q)}, \boldsymbol{C}_2^{(q)}, ..., \boldsymbol{C}_n^{(q)}\}$. To keep the calibration effort low, the number of calibration data points could be quite small so the challenge is to use a flexible transformation that can generalize well to the entire scene based the limited training data. Existing approaches almost exclusively focus on using rigid transformation but it is unclear if there are other types of transformations that might be able to produce better results. As such, we have experimentally tested a number of different transformations which are reviewed in the following subsections.

3.3.2.1 Rigid Transformation

The six degrees of freedom rigid transformation is commonly used to describe a relative camera pose in 3D space. For the camera pair (C_q, C_r), the transformation is determined by a rotation matrix $\boldsymbol{R}^{(qr)}$ parameterized by the three rotation angles θ_x, θ_y, and θ_z and a translation vector $\boldsymbol{t}^{(qr)} = [t_x, t_y, t_z]^T$ between the two camera centers. Putting them in the form of Equation (3.7) with C_r as the world (reference) coordinate system, we have $h(\cdot)$ as the identity function and the extrinsic matrix as

$$\boldsymbol{P}_q^{-1} = \begin{pmatrix} \boldsymbol{R}^{(qr)} & \boldsymbol{t}^{(qr)} \\ 0 & 1 \end{pmatrix} \tag{3.10}$$

To compute each unknown parameter, we require at least $n > 6$ correspondences. Our goal is to find $\boldsymbol{R}^{(qr)}$ an $\boldsymbol{t}^{(qr)}$ that minimizes the following cost function:

$$J_{RT}(\boldsymbol{R}^{(qr)}, \boldsymbol{t}^{(qr)}) = \sum_{i=1}^{n} \left\| \boldsymbol{C}_i^{(r)} - \boldsymbol{P}_q^{-1} \boldsymbol{C}_i^{(q)} \right\|^2 \tag{3.11}$$

Due to the orthogonality constraint on the rotation matrix $R^{(qr)}$, we use the least-square based method in [112] by first computing the covariance matrix as follows:

$$\boldsymbol{A} = \sum_{i=1}^{n} [(\bar{\boldsymbol{C}}^{(q)} - \boldsymbol{C}_i^{(q)}) \cdot (\bar{\boldsymbol{C}}^{(r)} - \boldsymbol{C}_i^{(r)})^T] \tag{3.12}$$

where $\bar{\boldsymbol{C}}^{(q)} = \frac{1}{n} \cdot \sum_{i=1}^{n} \boldsymbol{C}_i^{(q)}$ and $\bar{\boldsymbol{C}}^{(r)} = \frac{1}{n} \cdot \sum_{i=1}^{n} \boldsymbol{C}_i^{(r)}$ are the respective centroids of the two correspondence sets. Using Singular Value Decomposition $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$, we can compute the rotation matrix as $\mathbf{R}^{(qr)} = \boldsymbol{V}\boldsymbol{U}^T$ and $\boldsymbol{t}^{(qr)} = \bar{\boldsymbol{C}}^{(q)} - \bar{\boldsymbol{C}}^{(r)}$.

3.3.2.2 Polynomial Regression

The rigid transformation is sufficient if all sources of intrinsic distortion have been fully compensated. In practice, there are always residual error and a more flexible regression model could further minimize the error without overfitting. In this section, we focus on d-degree polynomial transformation $F^{(qr)}(\cdot)$ to map $\boldsymbol{C}_i^{(q)}$ to $\boldsymbol{C}_i^{(r)}$ for $i = 1, 2, \ldots, n$. We can parameterize the polynomial fitting problem by treating $F^{(qr)}$ as a matrix multiplication with the extrinsic matrix $\boldsymbol{P}_q^{-1}$, again treating C_r as the world frame, after a feature mapping function $h_d(\cdot)$. The overall cost function to be minimized is as follows:

$$J_{PR}(\boldsymbol{P}_q^{-1}) = \sum_{i=1}^{n} \left\| \boldsymbol{C}_i^{(r)} - \boldsymbol{P}_q^{-1} h_d(\boldsymbol{C}_i^{(q)}) \right\|^2 \tag{3.13}$$

$h_d(\cdot)$ expands an input 3D point into the products of all cross terms with up to d coordinates. For example, in the case $d = 2$, $h_2(\cdot)$ is as follows:

$$h_2\left([x \;\; y \;\; z \;\; 1]^T\right) = \left[x^2 \;\; y^2 \;\; z^2 \;\; xy \;\; xz \;\; yz \;\; x \;\; y \;\; z \;\; 1\right]^T \tag{3.14}$$

The corresponding extrinsic matrix $\boldsymbol{P}_q^{-1}$ would be a 4×10 matrix. This matrix has 30 degrees of freedom after removing redundancy based on the use of homogeneous

coordinates. While a regression function with a higher degree can fit the calibration data better, it might have problems generalizing to unseen data, especially outside the vicinity of the sphere trajectory. This problem can be addressed by cross-validation and we will evaluate and compare regression functions of different degrees in Section 3.4.

3.3.2.3 Manifold Alignment

Even without overfitting, non-rigid transformations can produce non-Euclidean artifacts that can significantly degrade the quality of the 3D reconstruction. As such, it is important to preserve as much as possible the metric relationship within the data. Manifold Alignment [8], unlike rigid body transformation and polynomial regression, can align correspondences across datasets, while preserving metric structures within each individual dataset. For camera calibration, its flexibility can potentially model the alignment better than rigid transformation, while preserving Euclidean relationship better than polynomial regression. In this dissertation, we adapt the feature-level alignment in [8] for our camera calibration problem. Given two valid 3D trajectories at camera C_r and C_q, the mapping functions $(\boldsymbol{F}^{(r)}, \boldsymbol{F}^{(q)})$ can register the points in the manifold space by minimizing the following cost function:

$$\begin{aligned} J_{MA}(\boldsymbol{F}^{(r)}, \boldsymbol{F}^{(q)}) = \mu \sum_{i=1}^{n} \|\boldsymbol{F}^{(r)}\boldsymbol{C}_i^{(r)} - \boldsymbol{F}^{(q)}\boldsymbol{C}_i^{(q)}\|^2 + \\ \sum_{i=1}^{n}\sum_{j=1}^{n} W_r^{i,j} \|\boldsymbol{F}^{(r)}\boldsymbol{C}_i^{(r)} - \boldsymbol{F}^{(r)}\boldsymbol{C}_j^{(r)}\|^2 + \\ \sum_{i=1}^{n}\sum_{j=1}^{n} W_q^{i,j} \|\boldsymbol{F}^{(q)}\boldsymbol{C}_i^{(q)} - \boldsymbol{F}^{(q)}\boldsymbol{C}_j^{(q)}\|^2 \end{aligned} \quad (3.15)$$

The first term of Equation (3.15) is the alignment cost between the two trajectories. The second and the third terms attempt to preserve the local metric relationship by incorporating the similarity measurements $W_r^{i,j}$ and $W_q^{i,j}$. Specifically, $W_r^{i,j} = \exp(-\|\boldsymbol{C}_i^{(r)} - \boldsymbol{C}_j^{(r)}\|^2)$ and $W_q^{i,j} = \exp(-\|\boldsymbol{C}_i^{(q)} - \boldsymbol{C}_j^{(q)}\|^2)$. μ is an empirical parameter to balance the two parts of the cost function.

To map the manifold alignment representation to our extrinsic matrix representation, it is easy to see that $\boldsymbol{P}_q^{-1} = \left(\boldsymbol{F}^{(r)}\right)^{-1} \boldsymbol{F}^{(q)}$ with an identity feature mapping. To ensure both $\boldsymbol{F}^{(r)}$ and $\boldsymbol{F}^{(q)}$ are invertible, the formulation in [8] also incorporates a regularization constraint to enforce a constant volume after the alignment. Unlike rigid transformation or polynomial regression, m invocations of pairwise manifold alignment with the reference camera will produce m *different* transformations at the reference camera. In order to produce just one transformation at the reference frame, we modify the cost function (3.15) so that the same transformation is used to simultaneously minimize the error with respect to every other camera.

3.3.3 Simultaneous Optimization

In the final stage, we jointly refine all the extrinsic parameters estimated from the previous steps to produce the simultaneously optimal extrinsic parameters. Our simultaneous optimization algorithm is based on Bundle Adjustment (BA) [113], which has been widely used in many 3D reconstruction applications. The goal of bundle adjustment is to simultaneously adjust the camera parameters and 3D points to minimize the overall projection error between the observed and expected locations of the 3D points. In the original formulation of [113], BA was carried out based on

the estimated 3D points and their 2D projections as follows:

$$\min_{\boldsymbol{P}_j, \boldsymbol{X}_i} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} d(f(\boldsymbol{P}_j, \boldsymbol{X}_i), \tilde{x}_{ij})^2 \tag{3.16}$$

where m and n are the total number of cameras and 3D scene points respectively. The function f denotes the relation that maps 3D point $\boldsymbol{X}_i$ in world coordinate to 2D image pixel $\boldsymbol{x}_{ij}$ by the corresponding projection matrix $\boldsymbol{P}_j$. The variable $v_{ij} \in \{0, 1\}$ indicates whether the point is visible by camera j. The function d denotes a distance function on the camera plane.

For our problem, we are interested in minimizing distance errors in 3D space instead of in 2D, and the obtained optimal extrinsic parameters will be used for our real-time 3D rendering. We assume that the intrinsic parameters of all camera are known. The input of for this stage are the m sequences of n 3D sphere center locations from the m RGB-D cameras: $\{\boldsymbol{C}_1^{(j)}, \boldsymbol{C}_2^{(j)} ..., \boldsymbol{C}_n^{(j)}\}$ for $j = 1, 2, \ldots, m$. The goal is to find the n "true" 3D points $\{\boldsymbol{C}_1, \boldsymbol{C}_2 ..., \boldsymbol{C}_n\}$ and the optimal extrinsic matrices $\boldsymbol{P}_j$ for $j = 1, 2, \ldots, m$ that transform these 3D points to the m observed sphere center sequences. Our pseudo bundle adjustment equation can be written as follows:

$$\min_{\boldsymbol{P}_j, \boldsymbol{C}_{i=1,\ldots,n}} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} \parallel h^{\dagger}(\boldsymbol{P}_j \boldsymbol{C}_i) - \boldsymbol{C}_i^{(j)} \parallel^2 \tag{3.17}$$

Different from the classical BA, our formulation uses the 3D Euclidean distance in the local camera coordinate system. The minimization problem (3.17) is non-linear and the standard approach is to use the Levenberg-Marquardt (LM) algorithm [114,115], which is an iterative procedure to find a local minimum of a cost function. At the t^{th} iteration step, our adapted LM procedure first computes an estimate of $[\boldsymbol{C}_i]_t$ for

$i = 1, 2, \ldots, n$ based on averaging the projections of the data points onto the world frame using the estimated $[\boldsymbol{P}_j]_{t-1}$:

$$[\boldsymbol{C}_i]_t = \frac{1}{\sum_{j=1}^{m} v_{ij}} \sum_{j=1}^{m} v_{ij} [\boldsymbol{P}_j]_{t-1}^{\dagger} h\left(\boldsymbol{C}_i^{(j)}\right) \tag{3.18}$$

Then, LM updates the estimates of the extrinsic matrices as follows:

$$[\boldsymbol{P}_j]_t = [\boldsymbol{P}_j]_{t-1} + \Delta_t^{(j)} \tag{3.19}$$

where $\Delta_t^{(j)}$ is determined based on a combination of steepest-descent and Gaussian-Newton methods in minimizing the cost function in (3.17) but fixing $\boldsymbol{C}_i = [\boldsymbol{C}_i]_t$ for $i = 1, 2, \ldots, n$. The iteration continues until the reduction in the cost function becomes negligible. The details of the LM procedure can be found in [115].

3.4 Experiments

The proposed algorithm is agnostic about the type of depth-sensing technologies, may that be stereo, structured-light or time-of-flight RGB-D cameras. For concreteness, we have chosen to use Microsoft Kinect v.1 structured-light cameras to capture all color and depth images in the experiments. They are inexpensive, which is an important consideration to build a large camera network. In addition, recent study has shown that structured-light cameras provide a good price and performance tradeoff among different types of depth sensors [116].

The setup of our camera network is shown in Figure 4.3. The camera network consists of 5 Kinect camera sparsely placed in an indoor room of 45.36m^2. A client-server architecture is built for parallel computing and data collection from the cameras.

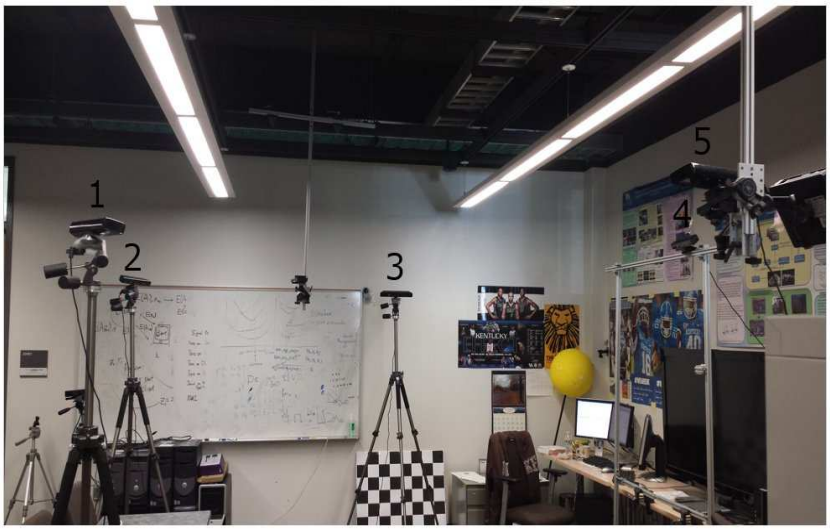


Figure 3.5: Overview of our camera network setup.

Each camera is connected to a separate client computer, which is a Wintel machine with Intel Core 2 Quad Q9650 processor, 8GB of RAM, running Windows 7. The server is Wintel machine with Intel Core i7-5820k processor and GeForce GTX-1080 GPU, 32.0GB of RAM, running Windows 10. The local network is a 100BASE-TX ethernet. In practice, our proposed system does not require extra setup effort or additional equipment. We have tested a multi-camera network using Wi-Fi and commodity hardware in our laboratory, various classrooms and auditorium in our university. During initial calibration stage, each client sends the identified sphere centers and timestamp information to the server. For online 3D rendering, each client sends aligned color and depth images with 640×480 resolution at 30 fps to the server. As such, static and dynamic objects are captured and reconstructed in real-time. To ensure the server receives the accurate corresponding frames sent by each camera, we set up a local NTP time server to synchronize all computers. The time server is equipped with a GPS Board, which provides a precise PPS (pulse per second) signal

for time synchronization. After synchronizing with the local time server, system time for capturing each frame among all computers are within 4 ms offset.

The performance of the proposed system described in Section 3.3 is systematically measured. A yellow sphere with known radius $\bar{R}$ is used as the calibration object as described in Section 3.3.1. We have tested spheres of different radii ranging from 127mm to 203.2mm. While there is no definitive mathematical relationship between the size of the ball and the algorithm, a larger ball is visible to more cameras and in general requires fewer video frames for calibration. As such, all the results in Section 3.3.1 are based on using the sphere with radius $\bar{R} = 203.2$mm. The three approaches of camera view transformation described in Section 3.3.2, including rigid transformation, polynomial regression, and manifold alignment, were tested. For polynomial regression, we have tested linear feature mapping feature and two variants of the quadratic feature mapping: a simplified version (3.14) without the cross terms and (3.14) itself. These three forms of regressions are denoted as Regression I with 12-DOF, Regression II with 21-DOF, Regression III with 30-DOF, respectively. Each of these methods is combined with the simultaneous optimization step as described in Section 3.3.3. To compare these methods with the state-of-the-art, we include the scheme by Herrera et al. [6] based on their publicly available software. Both quantitative and qualitative results of calibrations were measured as described below.

3.4.1 Quantitative Evaluation

For the initial calibration, about 1000 RGB-D images are captured for sphere detection by each camera. The whole calibration process is fully automatic in our

Table 3.1: Execution performance on our RGB-D camera network calibration

Task	Processing time
Sphere center detection	44ms (per frame)
Pairwise calibration	60ms
Global optimization	2.2s

constructed client-server architecture. The speed performance of each task during the calibration stage is shown in Table 3.1. The total execution time for our calibration is 47 seconds. The short execution time enables rapid reconfiguration of the camera network for any target applications. For real-time 3D capture after the calibration, the pairwise calibration step and simultaneous optimization are no longer needed. Instead, the color and depth data are locally compressed using motion-JPEG and transmitted to the server for rendering. The bandwidth requirement for 5 camera is measured to be 157.4 Mbits per second on average and the rendering speed of the point clouds from all cameras at the server is approximately 20 frames per second.

Next, we evaluate our sphere fitting algorithm. The goal of the sphere fitting algorithm is to estimate the location of the sphere center $\boldsymbol{C}(t)$ at frame t based on the observed 3D depth points $\boldsymbol{X}_{d_i}(t)$ for $i = 1, \ldots, D_p(t)$ identified on the sphere surface. While it is difficult to establish groundtruth for the unobservable sphere centers, we know the groundtruth radius of the sphere to be $\bar{R} = 203.2$ mm. As such, we can calculate the average deviation from the groundtruth radius of the radii estimated based on the identified sphere centers:

$$\sigma = \sqrt{\frac{1}{\sum_{t=1}^{T} D_p(t)} \sum_{t=1}^{T} \sum_{i=1}^{D_p(t)} \left(\bar{R} - \|\boldsymbol{X}_{d_i}(t) - \boldsymbol{C}(t)\|\right)^2} \tag{3.20}$$

300 sequential frames are tested to evaluate the sphere fitting accuracy. The esti-

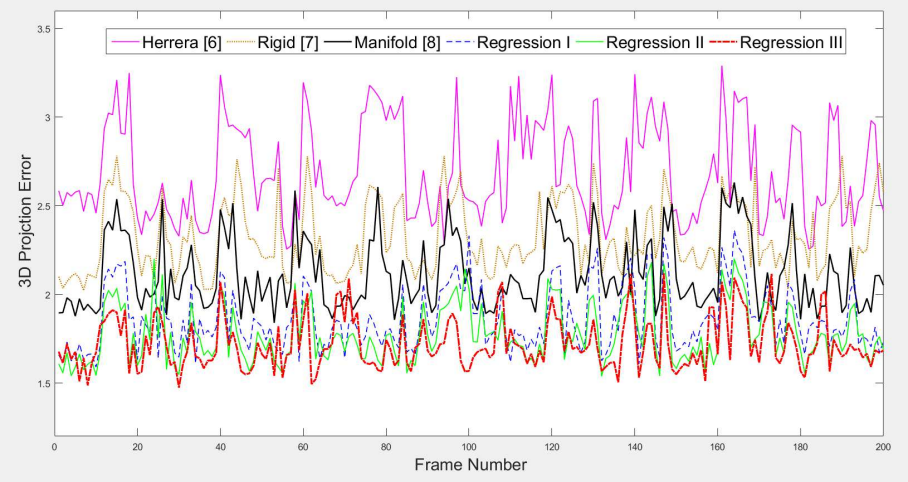


Figure 3.6: The 3D projection error of camera 1 for each frame.

mate is unbiased with standard deviation σ equalled to 6.42 mm or 3.15 % of the groundtruth radius.

After the initial pairwise calibration, the acquired initial extrinsics are then optimized by our pseudo bundle adjustment algorithm. To test whether the estimated extrinsics can extrapolate unseen data, we apply them on a separate testing set of RGB-D images, which has the 200 detected sphere centers for each camera, to validate the correctness of the calibration. The back-projection error for each camera is compared across all the schemes. To calculate back-projection error, we use the pre-computed extrinsic matrices to project the 3D sphere center locations in local coordinates to the global frame, take the average of the projections from all the cameras, back-project it onto each local coordinate system and calculate the root mean square error (RMSE). To show that there is no bias in each method, we show the 3D projection error of camera 1 for each frame in Figure 3.6. As shown in the figure, different curves representing different schemes seldom cross over each other. This shows that the relative performance among the five schemes stay constant regardless of the

Table 3.2: 3-d projection errors on validation data

Local Coordinate	Herrera [6]	Rigid [7]	Manifold [8]	Regression I	Regression II	Regression III
Camera 1	2.74 ± 0.23	2.40 ± 0.2	2.24 ± 0.22	1.98 ± 0.19	1.87 ± 0.17	1.80 ± 0.15
Camera 2	2.73 ± 0.22	2.36 ± 0.21	2.01 ± 0.23	2.01 ± 0.15	1.94 ± 0.16	1.88 ± 0.18
Camera 3	4.94 ± 0.54	4.56 ± 0.42	2.29 ± 0.22	2.12 ± 0.2	1.90 ± 0.16	1.85 ± 0.2
Camera 4	2.86 ± 0.22	2.29 ± 0.18	1.56 ± 0.12	1.44 ± 0.11	1.40 ± 0.1	1.49 ± 0.12
Camera 5	1.86 ± 0.17	2.33 ± 0.2	2.27 ± 0.17	2.05 ± 0.19	1.88 ± 0.17	1.84 ± 0.17
Average	3.03	2.79	2.07	1.92	1.80	1.77

Table 3.3: p-value hypothesis testing

Local Coordinate	R. III - Herrera [6]	R. III - Rigid [7]	R. III - Manifold [8]	R. III - R. I	R. III - R. II
Camera 1	0.0001	0.0001	0.0001	0.0905	0.9996
Camera 2	0.0001	0.0001	0.9998	0.9952	0.9999
Camera 3	0.0001	0.0001	0.0001	0.1302	0.9999
Camera 4	0.0001	0.0001	0.0001	0.1517	0.9989
Camera 5	0.0001	0.0001	0.0001	0.8743	0.9999

location of the sphere. As the plots for other cameras are similar, only the results for camera 1 are shown in the figure. Table 3.2 shows the mean and standard deviation of the 3D projection errors of the entire trajectory for each camera-method combination. Table 3.3 and 3.4 show the p-values and t-values when comparing Regression III with each the other methods, with the null hypothesis that the two methods produce similar results and the alternative hypothesis that Regression III produces better results. The sample size is 40. Based on the recommendation by Fisher [117], there are very strong evidence ($p < 0.001$) among the majority of the cameras against the null hypothesis when comparing Regression III with Herrera [6], Rigid [7], and Manifold [8]. On the other hand, there are no evidence ($p > 0.1$) against the null hypothesis when comparing Regression III with the other two regression techniques.

For the visual alignments, the plots of the sphere movement points in the global frames are in Figure 3.7. One can see that there are significant misalignment error

Table 3.4: t-value hypothesis testing

Local Coordinate	R. III - Herrera [6]	R. III - Rigid [7]	R. III - Manifold [8]	R. III - R. I	R. III - R. II
Camera 1	-17.84	-15.12	-8.56	-1.34	3.32
Camera 2	-15.96	-8.31	3.49	2.59	5.74
Camera 3	-33.34	-33.09	-5.18	-1.13	4.58
Camera 4	-18.56	-16.18	-7.14	-1.03	3.08
Camera 5	-17.37	-14.41	-5.62	1.15	5.75

among trajectories from different cameras in Herrera's scheme and the Rigid scheme. The Manifold scheme produces a skewed global frame but appears to produce reasonable alignment. All regression schemes produce similar alignment results, with minor improvements as the degree of freedom increases. We should caution that while the testing data are different from the training data, they are all captured in the same center area where there is a significant overlap among the fields of view of different cameras. To extrapolate the coverage into areas with little overlap, we evaluate in the next section the visual quality of captured 3D environment including static background and moving foreground when the system performs real-time 3D rendering.

3.4.2 Qualitative Evaluation

In this section, we evaluate the results of our real-time 3D rendering in indoor environment by RGB-D camera network. Figure 3.8a and 3.8b show the coverage area from each camera and the merged camera view respectively. One can see that the center area has a higher density of point cloud than the surroundings. To compare alignment accuracy among different camera view transformations, we first consider the reconstruction of the foreground objects near the center of the captured area,

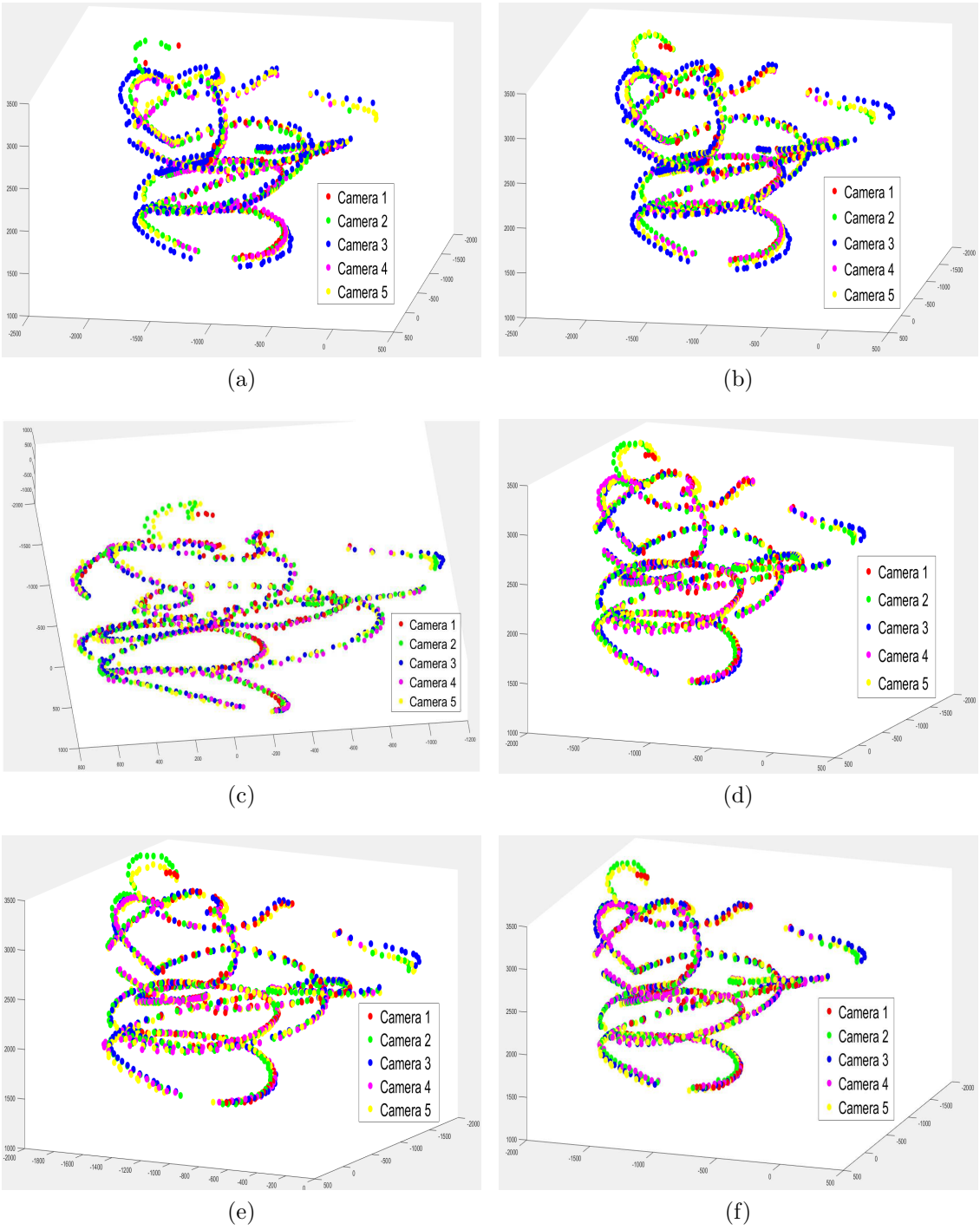
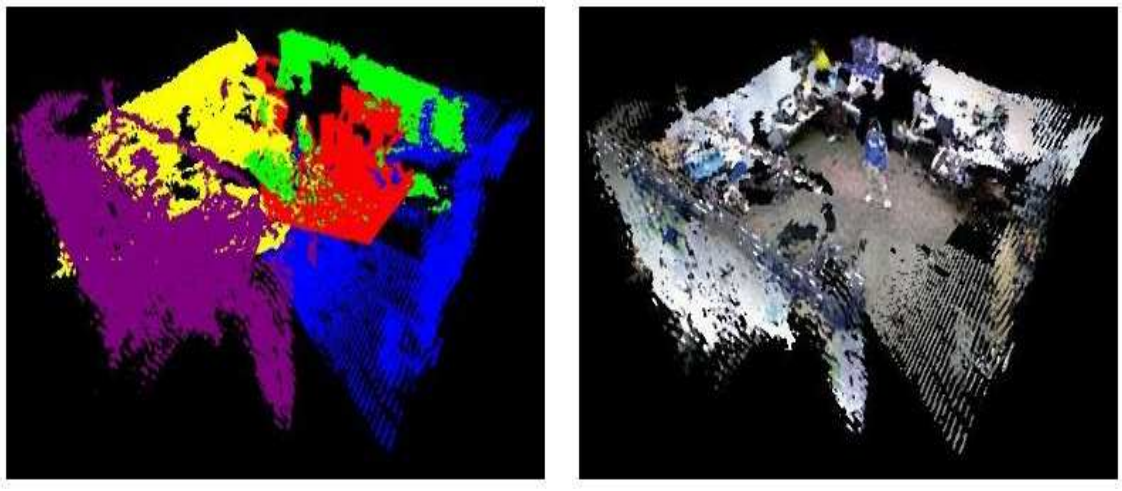


Figure 3.7: Sphere movement points alignment: (a) Herrera [6]; (b) Rigid [7]; (c) Manifold [8]; (d) Regression I; (e) Regression II; (f) Regression III.



(a) (b)

Figure 3.8: Real-time camera view merging in indoor room using Regression I. Field of view for each camera is rendered in a different color in the left image.

which include a stationary mannequin and a walking person. Two randomly-selected virtual viewpoints of the mannequin and one viewpoint of the person are shown in Figure 3.9. Similar to the numerical results in Section 3.4.1, using polynomial regression produces better alignment on the model's face and feet than the other methods. Next, we evaluate the reconstruction quality of the entire indoor scene for each view transformation method. In Figure 3.10, Herrera's scheme has significant alignment problem when extending beyond the center area. Rigid and Manifold schemes produce similar reconstruction results. All regression schemes have better alignments (less holes in the reconstructions) but Regression II and III start to introduce non-linear distortion in the background wall. The reason is the data from the training set are overfitted by higher degree parameters and cross terms, which can no longer preserve the geometry of the far-distance objects outside the calibration area. Overall, Regression I produce the best reconstruction results in our experiments with the

Figure 3.9: Comparison of 3D point cloud alignments zoom-in on the specific targets. From left to the right column: (1) Herrera [6]; (2) Rigid [7]; (3) Manifold [8]; (4) Regression I; (5) Regression II; (6) Regression III.

minimal amount of misalignment and geometrical distortion.

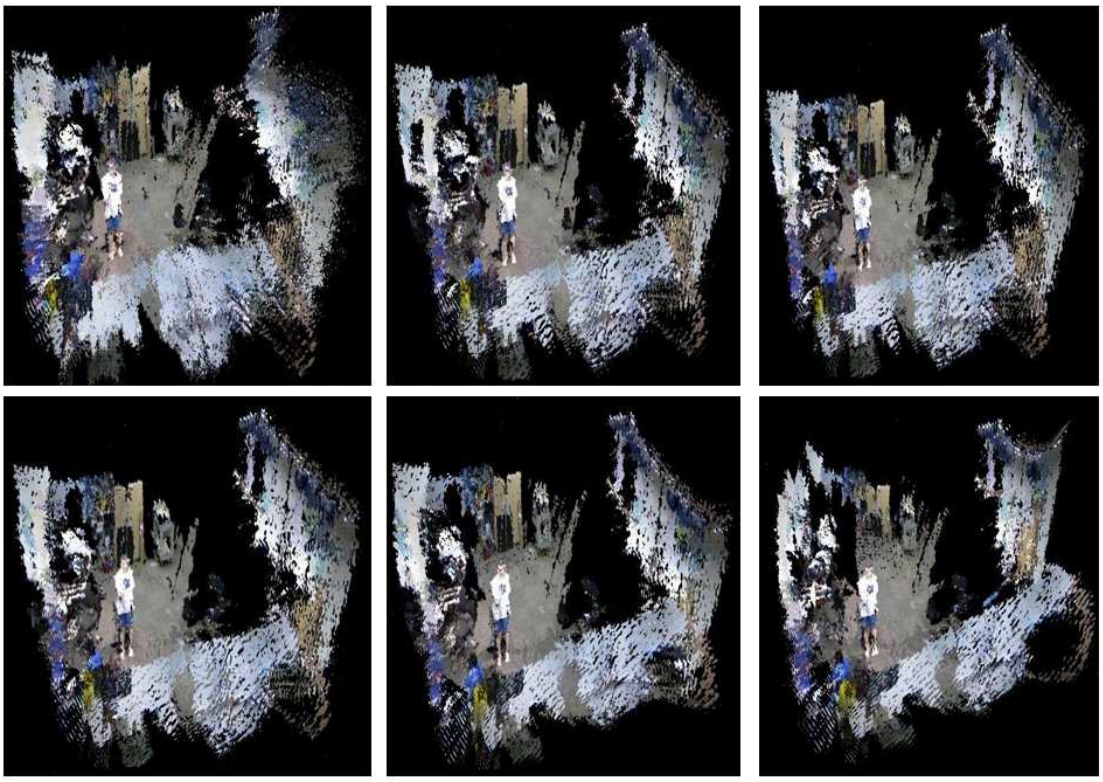


Figure 3.10: Comparison of 3D point cloud alignments on the indoor environment. From the top to the bottom (row-major order): (a) Herrera [6]; (b) Rigid [7]; (c) Manifold [8]; (d) Regression I; (e) Regression II; (f) Regression III.

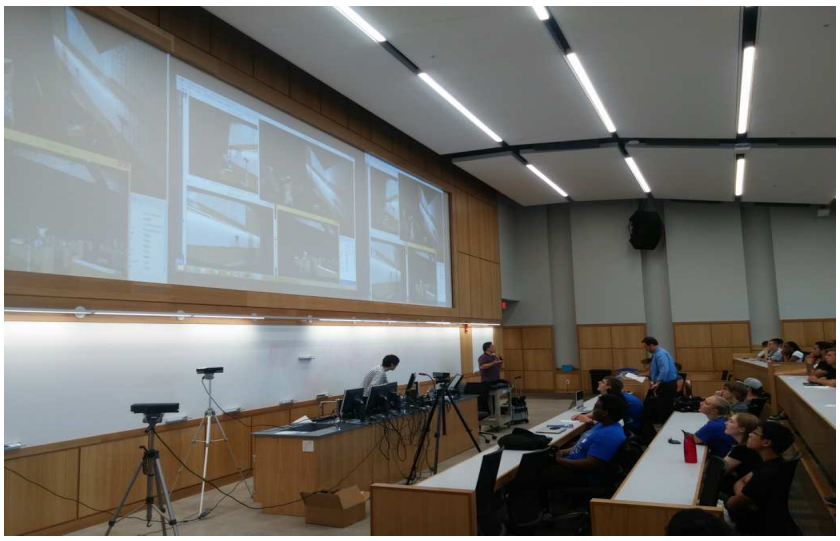


Figure 3.11: Real-time demonstration of our camera network system in the classroom.

Chapter 4 Real-time Physical Scene Rendering System

In this chapter, we describe the proposed real-time physical scene rendering system with multiple RGB-D cameras. The system can render in real-time dynamic 3D scenes on a virtual arbitrarily-shaped curved mirror. Moreover, we explore the possibility of using dynamically reconstructed scenes to enhance 3D interactions between people who are geographically separated.

4.1 Curved Mirror Simulation

To simulate a virtual curved mirror, the simple setup of having a video camera on top of a monitor and showing the output of the camera on the monitor is clearly insufficient - the viewpoint is fixed for a camera while the mirror depends on the position of the viewer. In addition, for a curved mirror surface, light rays emanating from a scene point may be reflected through multiple paths towards the viewpoint. Thus, the main challenge of simulating the curved mirror is to render reflective content correctly on the display depending on the viewers perspective. In order to simulate a large curved mirror surface that can cope with wide displacement of a viewer, a camera display system must be able to capture the 3D environment while rendering the new view based on the viewer's position. Furthermore, it must be able to accomplish all these tasks in real-time and with high fidelity, otherwise the virtual mirror system loses the instant visual feedback required to provide the realism of a mirror.

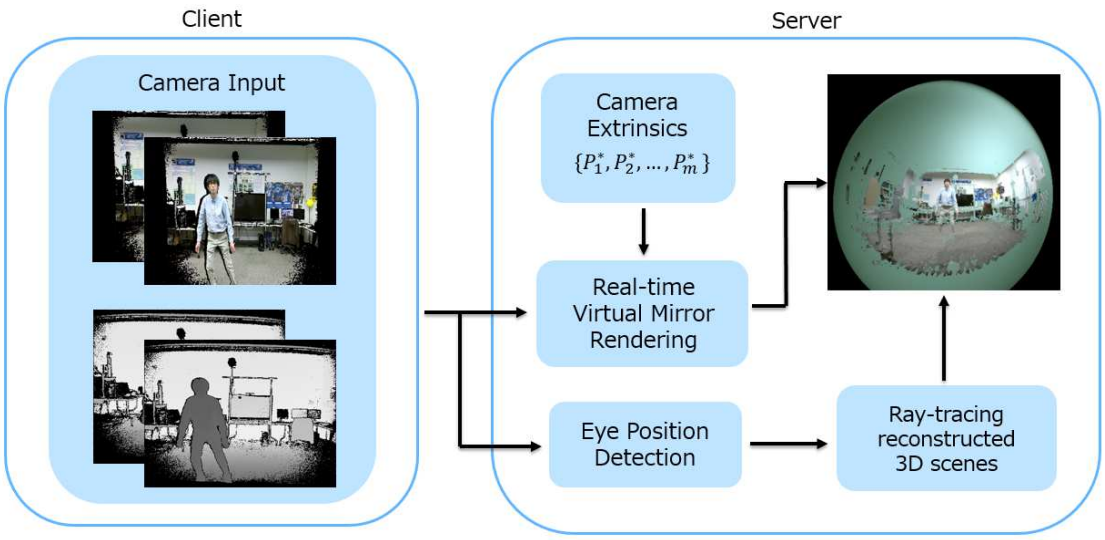


Figure 4.1: Overview of our curved mirror rendering framework.

4.1.1 Overview

The major functional building blocks of our virtual mirror rendering system are shown in Figure 4.1. The system is based on the framework in Chapter 3, a server-client architecture for calibrating multiple cameras, capturing and rendering physical scenes in real-time. Our multiple RGB-D camera calibration software collects point clouds from disparate cameras can be registered into an unified coordinate system. To provide scalability, each RGB-D camera is connected to a computer served as a client to collect depth and color information and detect correspondences for calibration. As such, much of the computational tasks can be executed on client side. Next, the server receives the data from the clients and then rendering 3D environments and reflective scene on virtual curved mirror. Each client establishes time synchronization across the camera network using Network Time Protocol (NTP) [107]. Details of rendering reflective scene on virtual curved mirror are described in the following section.

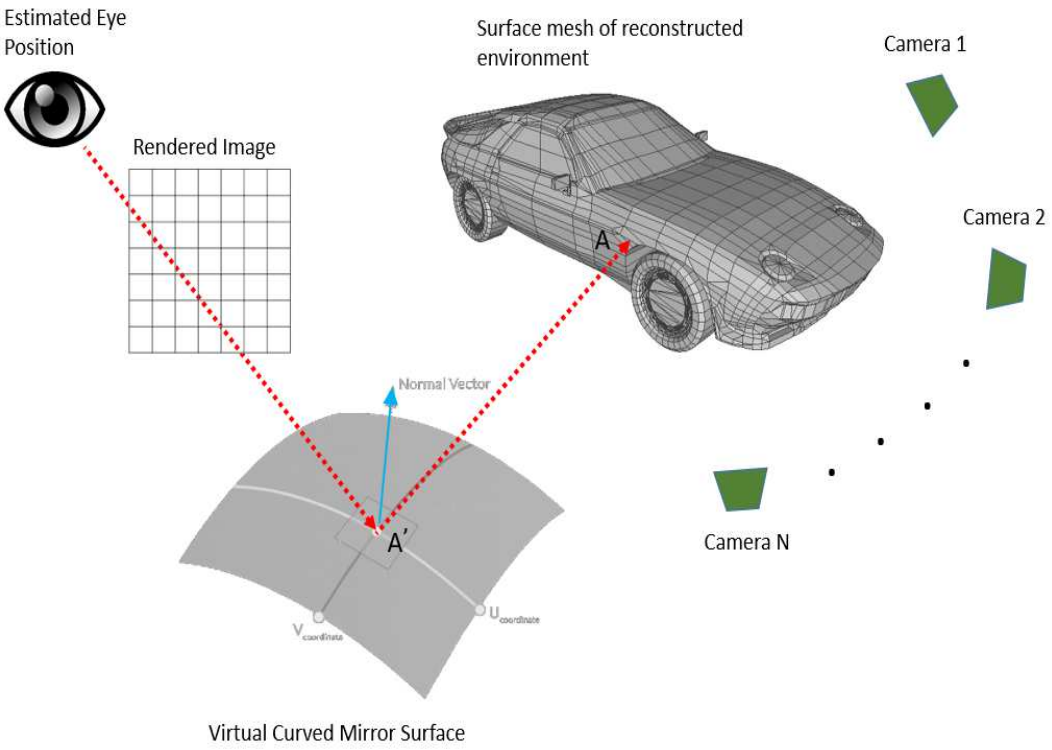


Figure 4.2: Scene reflection on virtual curved mirror

4.1.2 Reflective Scene on Virtual Curved Mirror

The process of generating reflective scene on a virtual mirror is shown in Figure 4.2. RGB-D information of the environment and 3D position of the viewpoint are necessary to accomplish this process in a virtual mirror system. The mirror image is based on reflecting a virtual light ray from the reconstructed 3D scene to our eye. For example, the scene point A is reflected to the spot A' on the virtual curved mirror surface in Figure 4.2.

The method to calibrate RGB-D camera network for 3D scene capture has been described in Chapter 3. After aligning all the cameras to the same coordinate, the system proceeds to the rendering phase for virtual mirror rendering. To render reflective scenes on a virtual curved mirror, 3D point clouds obtained from the RGB-D

cameras are triangulated into surface meshes for ray tracing from eye position. To support real-time rendering, we create surface meshes for each camera separately and merge them together at the server to avoid expensive mesh refinement processes. Given a pair of aligned RGB-D image $\boldsymbol{I}_{c_j}$ and $\boldsymbol{I}_{d_j}$, each pixel and any two of neighboring pixels in $\boldsymbol{I}_{d_j}$ can form 3D triangles. Specifically, each local region consists of four pixels $\boldsymbol{u}_1 = (x, y)$, $\boldsymbol{u}_2 = (x+1, y)$, $\boldsymbol{u}_3 = (x, y+1)$, $\boldsymbol{u}_4 = (x+1, y+1)$ in the depth image $\boldsymbol{I}_{d_j}$ and can generate two triangles $T_1 = (E_{\boldsymbol{u}_1\boldsymbol{u}_2}, E_{\boldsymbol{u}_2\boldsymbol{u}_3}, E_{\boldsymbol{u}_1\boldsymbol{u}_3})$ and $T_2 = (E_{\boldsymbol{u}_2\boldsymbol{u}_3}, E_{\boldsymbol{u}_2\boldsymbol{u}_4}, E_{\boldsymbol{u}_3\boldsymbol{u}_4})$. Here x and y are denoted as the column and row indices in depth image. E represents triangle's edge in 3D space. If one of the edges in the triangle is longer than an empirically defined threshold E_t, that triangle will be discarded. The threshold is determined by the depth of the triangle. If the triangle is far from the reference camera, higher threshold is defined for checking the triangle's eligibility. To colorize triangle meshes, we compute the barycentric point to determine what percentage for each vertex's RGB value contributing to the surface in the triangle.

Since the reconstructed 3D environments may have many missing regions caused by the occlusions between foreground and background, we pre-capture static background from each camera to provide more complete triangulated surfaces during ray tracing process. To further improve the quality of a reconstructed scene, the 3D scene scanning algorithm can be adopted in [25], a volumetric reconstruction method called truncated signed distance function or TSDF that can fuse captured point clouds into a single surface. The TSDF value stored at each voxel is based on the signed distance towards the closest 3D scene point. The sign of this distance value is based on whether

the voxel is in front or behind the implicit surface as defined by the local surface normal. The distance value is truncated as only the voxel values that are sufficiently closed to the surface are recorded. Given a TSDF structure, we can generate the surface by first raycasting from the optical center of the virtual camera and traverse through the voxel structure. A visible surface point is identified when the ray passes through a zero-crossing region in which the distance values change from positive to negative. The camera pose of a moving RGB-D camera is estimated sequentially by aligning the captured 3D point cloud at the current frame with the predictive surface points obtained from the TSDF structure followed by an Euclidean transformation corresponding to the last estimated camera pose. The alignment is achieved by estimating the rotation and translation based on the iterative closest point algorithm (ICP), which minimizes the combined point-plane energy for all existing point-pair between the current frame and the predicted frame. The resulting transformation is then applied to the TSDF structure so that it aligns with the current frame. After the alignment, we can then proceed to update the TSDF structure using the current frame data. By repeating the previous steps, the 3D surfaces can be accurately generated.

After forming surfaces from captured point clouds, the eye detector in our system tracks the midpoint between the two eyes of the viewer in the color image and calculates the 3D position based on the depth image. Virtual light rays are traced from the viewpoint to each rendering position on the curve surface and then reflected into the physical space. The algorithm then searches all stored triangular meshes and identifies the one intersecting with each reflected ray as the reflected scene point to

Figure 4.3: Camera network setup for virtual mirror rendering.

be rendered. As shown in Figure 4.2, the reflective scenes are rendered on a virtual curved mirror in our system. To obtain precise rendering results, the ray tracing algorithm on quadratic surface is used for our virtual curved mirror rendering [118].

4.1.3 Experiments

4.1.3.1 Environmental Setup

In our experiments, Microsoft Kinect v.2 cameras are used to capture color and depth images. The camera network consists of 4 cameras sparsely placed in a medium sized space 5.2m $\times$ 5.4m where background and dynamic foreground objects can be captured. Figure 4.3 is our camera network setup. The chosen reference camera tracks viewer's eye and estimates its 3D position for ray tracing reconstructed scenes. A client-server architecture is built for data collection from the cameras. Each camera is connected to a separate computer and send aligned and compressed RGB-D image data with 960×540 resolution to a server during the calibration stage and for online

Figure 4.4: Eye tracking by IntroFace [9]

3D environment and virtual mirror rendering. The total bandwidth for each frame is about 1.02MB, with 0.96MB for 3D data and 0.06MB for color data. Therefore, the total data bandwidth requirement is about 4.1MB per frame or 980Mbps at 30fps. With the prevalence of 40/100 Gigabit Ethernet or even with the latest 802.11ad wireless standard, our architecture can scale up to more cameras without having any issue on data transmission. To ensure the server receives the right corresponding frames sent by each camera, we set up a local NTP time server to synchronize all computers. We installed a GPS Board at the time server, which can output a accurate PPS (pulse per second) signal. After synchronizing with the local time server, system time for capturing each frame among all computers are less than 4ms clock skew, which is negligible for typical human movements.

Our virtual mirror system is implemented by C/C++. In addition, OpenCV library is used for 3D image processing and OpenGL library is used for 3D scene rendering. To render reflective dynamic scenes, we use CUDA to accelerate the ray tracing algorithm. A viewer's eye position is tracked by using software IntraFace [9]

as Shown in Figure 4.4. The hardware setting for the server is Intel Core (TM) i7-5820k CPU at 3.30 GHz and 32.0GB of RAM. The GPU is GeForce GTX-1080.

4.1.3.2 Calibration Results

To calibrate our camera network, we capture about 250 RGB-D images from each client and perform pairwise camera calibration. Then, bundle adjustment with 30 iterations is used to optimize the extrinsic parameters of each camera by minimizing the total 3D projection errors. To evaluate the alignment errors, root mean square error (RMSE) is used to calculate detected sphere center between the reference camera and all other cameras in the reference coordinate:

$$E_p = \sqrt{\frac{1}{n}\sum_{i=1}^{n}[(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2 + (\hat{z}_i - z_i)^2]} \tag{4.1}$$

where $(\hat{x}_i, \hat{y}_i, \hat{z}_i)$ is the observed sphere center point, (x_i, y_i, z_i) is the estimated sphere center point, n is the number of sphere center points in the reference camera coordinate. Since the estimated extrinsic parameters will be used to transform 3D points from local camera coordinate to the reference coordinate for online 3D rendering, we apply them on another three sets of detected sphere centers to check the accuracy of our camera calibration. The error analysis result in Table 4.1 showed that the average of RMSE for each camera pair was under 2.325cm resulting in less than 0.6% relative error over the entire environment.

4.1.3.3 Reflective Scenes Rendering Results

Figure 4.5 is our reconstructed 3D environment. The curved, green light shiny surface is the virtual mirror surface. Each virtual light ray originated from the viewer's

Table 4.1: Alignment error analysis by RMSE

Validation data	RMSE (cm)	# of frames
Dataset 1	2.325	238
Dataset 2	2.2	245
Dataset 3	2.23	232

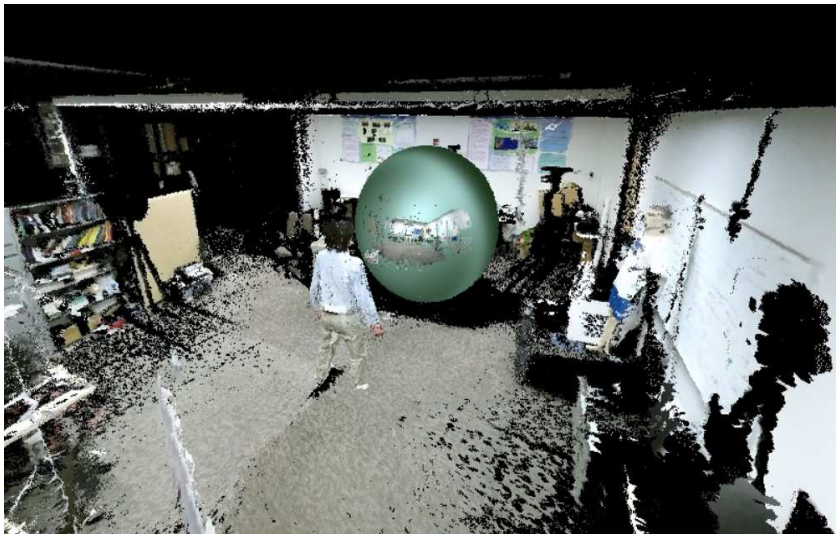
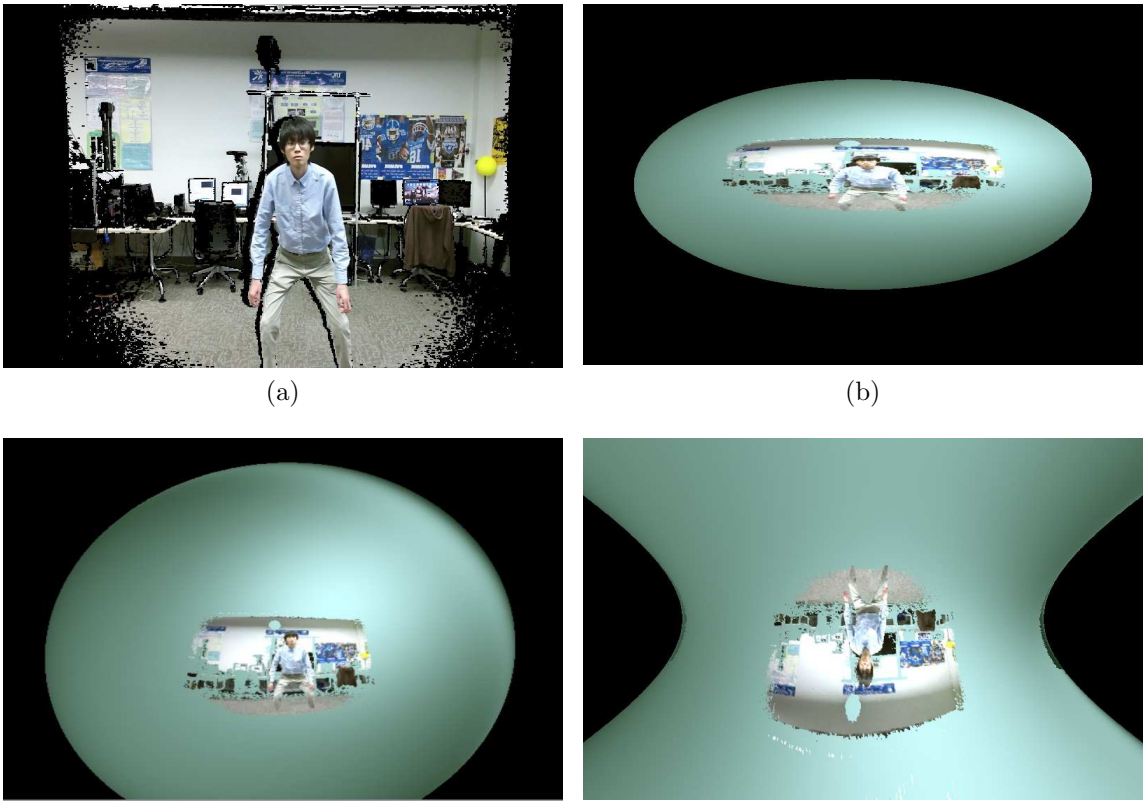


Figure 4.5: Reconstructed 3D environment with virtual mirror.

eye position and reflected by the virtual curved surface traces out the reconstructed scene through a 960x540 image plane. Our CUDA optimized code captures and renders at, on average, 6.7 frames per second. Our experimental results using different types of curved mirror surfaces are shown in Figure 4.6, 4.7 and 4.8. To compare the range of rendering scenes between single camera and multiple cameras, first, the results by using only one Kinect camera are shown in Figure 4.6. The top row of Figure 4.6a is the RGB image captured from the reference camera and Figure 4.6b, 4.6c and 4.6d are the corresponding reflective scenes. The reflective scenes are very limited due to the insufficient captured 3D information.

By setting up multiple cameras around the environment, we can acquire a much



(a) (b)

(c) (d)

Figure 4.6: Reflective scenes on different virtual curved surface rendered by only 1 camera: (a) Input RGB image; (c) Ellipsoidal mirror; (c) Sphere mirror; (d) Sandglass mirror.

wider view on the mirror surfaces. The results by using multiple Kinect cameras with different viewpoint positions are shown in Figure 4.7 and 4.8. The top rows are input RGB images and the rest of the rows are reflective scenes. The results showed that the reflective scenes changed based on the viewer's eye position detected by the eye tracker. Convex mirror surfaces are capable of capturing a large environment in a small rendering surface. While the multiple camera setup enlarges the field of view, there are still some uncovered ranges that cannot be readily captured, for example, ceiling and very far distance scenes. For the concave type of mirror shown in the last

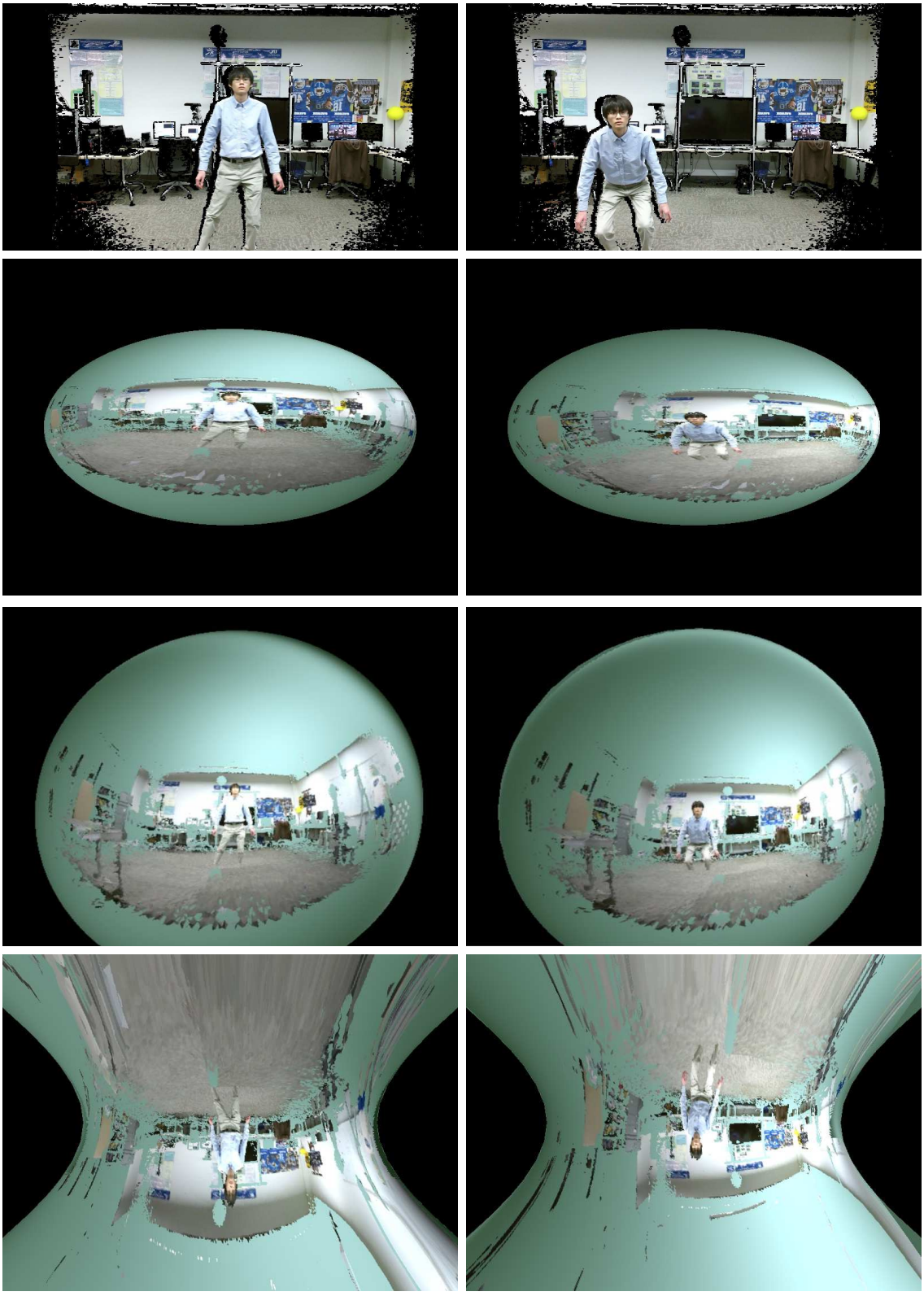


Figure 4.7: Reflective scenes on different virtual curved surface rendered by 4 cameras. From top to bottom: (1) Input RGB image; (2) Ellipsoidal mirror; (3) Sphere mirror; (4) Sandglass mirror.

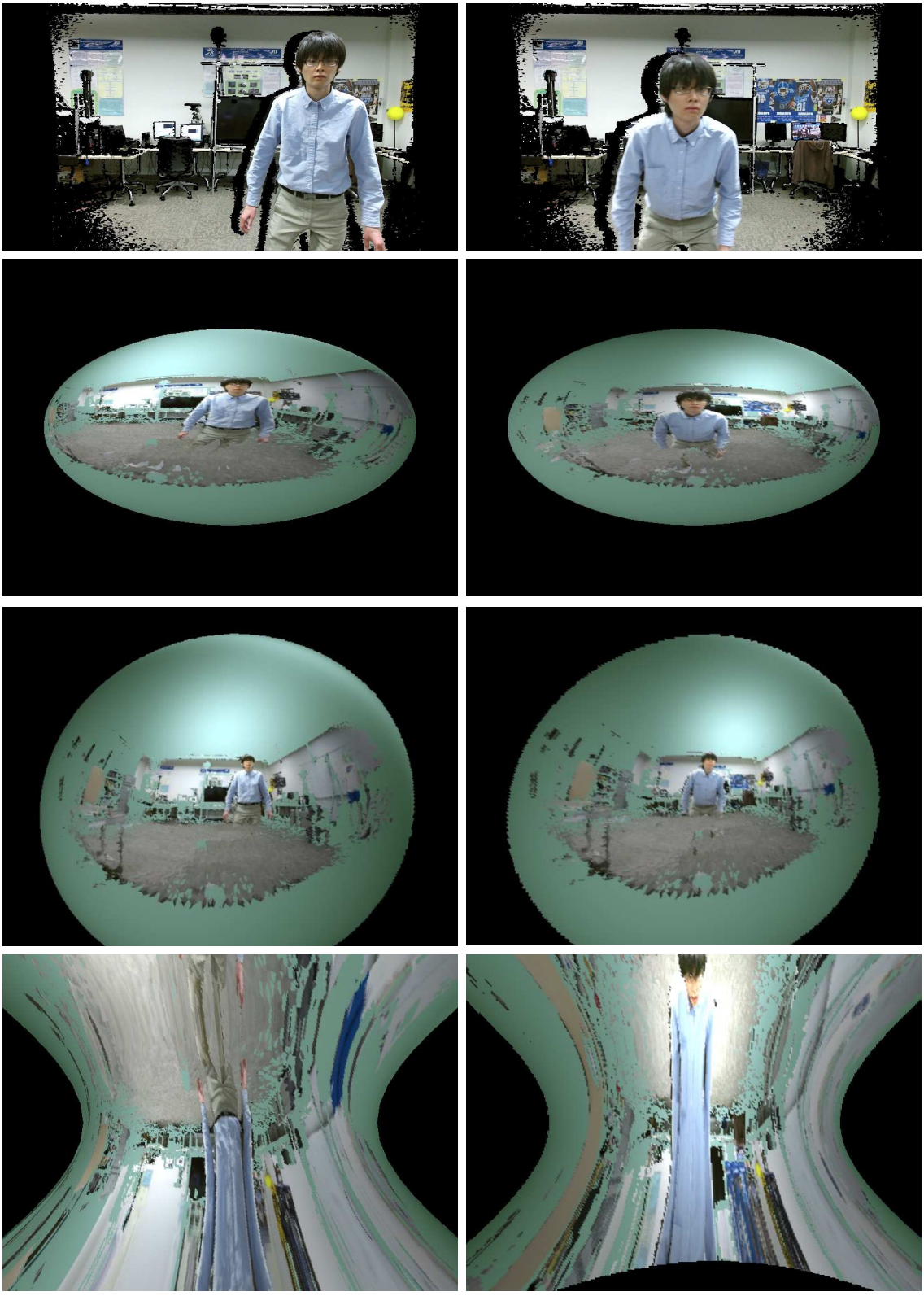


Figure 4.8: Reflective scenes on different virtual curved surface rendered by 4 cameras. From top to bottom: (1) Input RGB image; (2) Ellipsoidal mirror; (3) Sphere mirror; (4) Sandglass mirror.

row of Figure 4.7 and 4.8, the captured objects in near distance from the camera were enlarged while the captured scenes in far distance were flipped vertically.

4.2 Application: 3D Telepresence

In this section, an application based on remote 3D interaction is implemented in our RGB-D camera network. Our real-time scene acquisition and reconstruction system enables people to communicate in a 3D environment. The availability to acquire and render wide-viewed real world with dynamic perspectives to end users provides realistic interactive experiences. The system is scalable to bring multiple users from different places into a large reconstructed environment. A series of experiments are performed to verify the proposed system and demonstrate the potential benefits to existing telepresence systems with better motivating and engaging effects.

4.2.1 Overview

Our proposed framework for 3D telepresence is shown in Figure 4.9. A camera network is first calibrated and then used to provide users with dynamically reconstructed environments based on our proposed scheme in Chapter 3. The scalability of a camera network is provided by connecting each camera to a computer served as a client to capture, process and transmit RGB-D data. The server is responsible for calculating extrinsics between cameras and rendering 3D scenes. The dynamic target objects from a remote site are merged into the reconstructed environment in real-time. Details of our proposed framework are described in the following section.

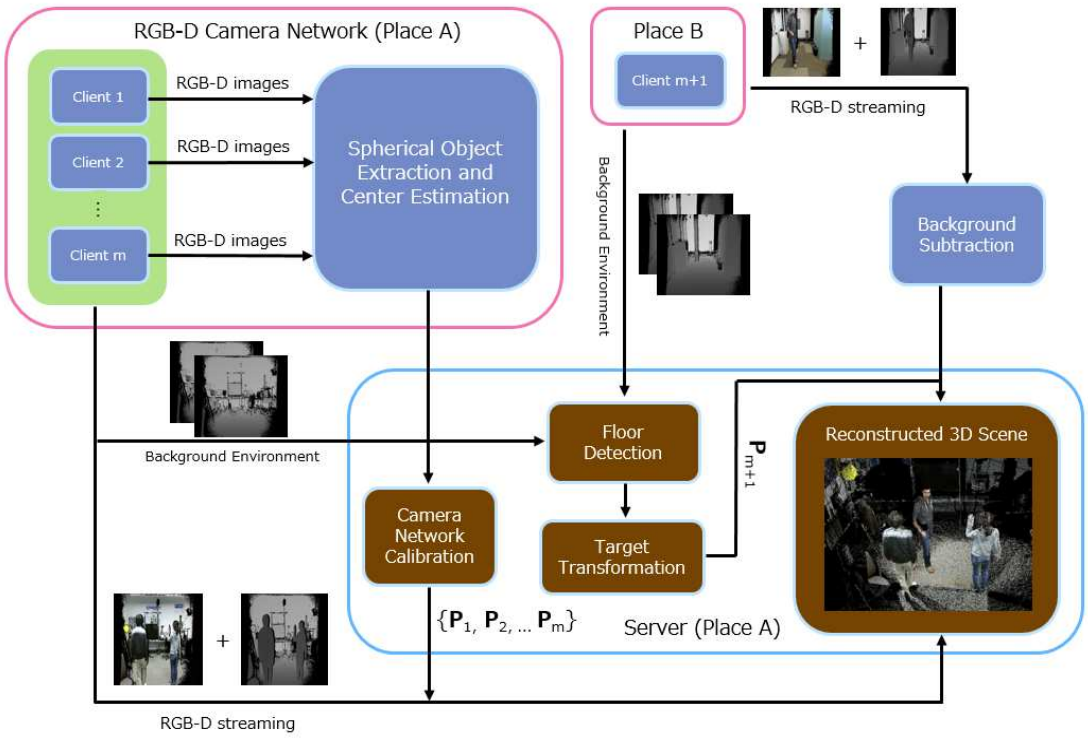


Figure 4.9: Overview of our 3D telepresence framework.

4.2.2 Scene Reconstruction and Remote Target Fusion

We describe our proposed system that can merge target objects from remote places into a dynamically reconstructed environment. The camera network calibration scheme in Chapter 3 is extended to further expand the field of view while maintaining as few cameras as possible. Previously we assume there is a significant overlap in the same center area among the fields of view of different cameras. Pairwise calibration can be carried out smoothly to find the view transformation from each camera to an arbitrarily-chosen reference camera. However, non-overlapping issues occur when cameras point to non-central region as illustrated in Figure 4.10. Here we use the same mathematical notation to denote camera. For each camera C_j in the camera

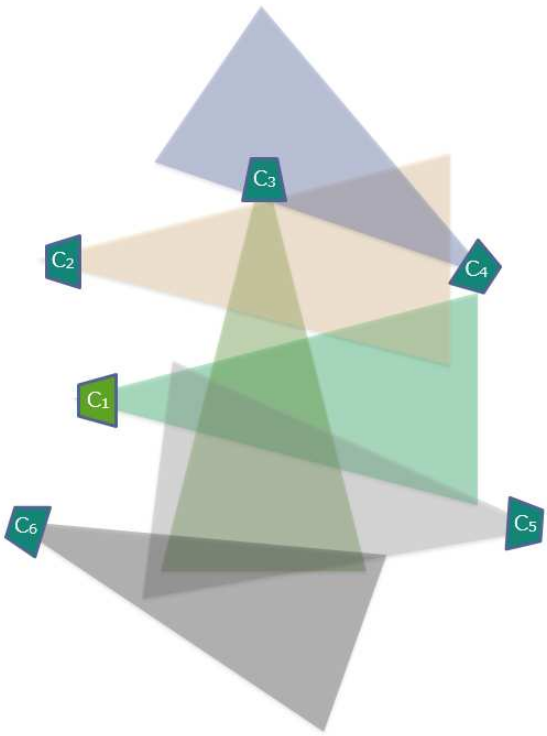


Figure 4.10: The camera network with non-overlapping between reference camera C_1 and camera C_4, C_6.

network ($j = 1, \ldots, m$), if there is no overlapping with reference camera C_1, it will search other cameras that share field of view in the same regions. The closest camera in physical distance with valid corresponding pairs to C_j is selected and then used to estimate indirect view transformation between C_1 and C_j. After initial calibration, we optimize camera extrinsics using Equation 3.17 for our camera network.

After camera network calibration, the system starts estimating view transformation between the reference camera and the camera at a remote location. Our remote target fusion relies on depth information of planar floor to make the floor for both places align in parallel. Here we assume floor occupies most of the lower portion of the captured images. To segment out the floor, first, we calculate the surface normal of each 3D point based on neighboring 3D points. More specifically, for each pixel

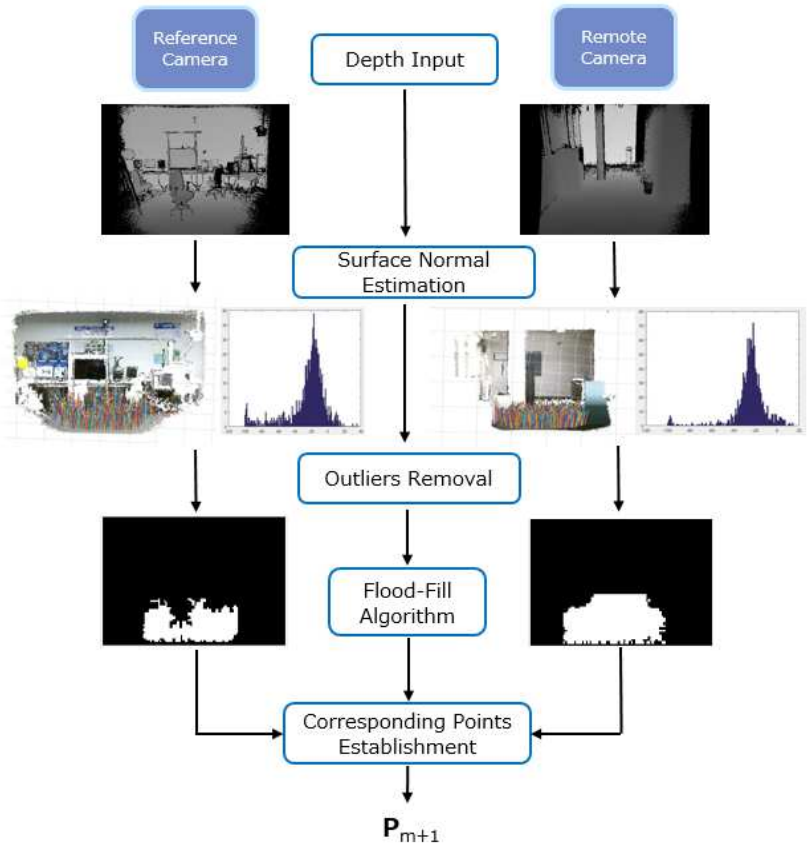


Figure 4.11: Flow chart of planar floor alignment

and its neighboring pixels $\boldsymbol{u}_1 = (x, y)$, $\boldsymbol{u}_2 = (x+1, y)$ and $\boldsymbol{u}_3 = (x, y+1)$ in the depth image $\boldsymbol{I}_{d_j}$, we can calculate the surface normal $\boldsymbol{N}_s$ by taking the cross product of the vectors $\boldsymbol{L}_1 = \boldsymbol{V}(\boldsymbol{u}_2) - \boldsymbol{V}(\boldsymbol{u}_1)$ and $\boldsymbol{L}_2 = \boldsymbol{V}(\boldsymbol{u}_3) - \boldsymbol{V}(\boldsymbol{u}_1)$, where u and v are the column and row indices in $\boldsymbol{I}_{d_j}$. $\boldsymbol{V}(\cdot)$ is the 3D point obtained by back-projecting 2D image point to 3D space in Equation 3.7.

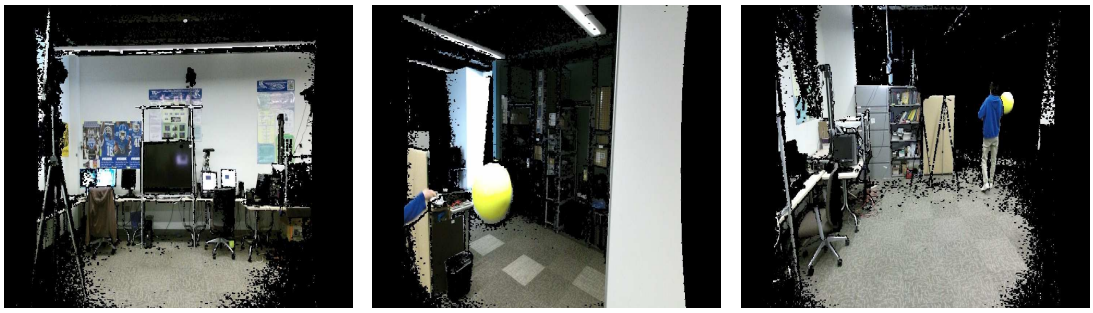
Next, we classify the surface normals into several groups based on their directions. Since we know planar floor dominates the lower part of the image, a group that contains most surface normals must belong to the floor. The initial segment result is converted into the binary image, which indicates the detected floor as white region. Then, the flood-fill algorithm is applied to determine the largest connected blob as

aligning correspondence [119]. Therefore, the corresponding point pairs are built to align the floors between the reference camera and the remote camera. We downsample the number of corresponding pairs by taking the first pixel of each $n \times n$ block in the image. The transformation P_{m+1} can be estimated by minimizing the distance between the corresponding pairs [112]. The workflow of the planar floor alignment algorithm is shown in Figure 4.11. After estimating the transformation for the floor alignment, a remote target is captured and brought into the reconstructed scene by RGB-D camera network. At the remote site, the static background mask based on depth information is used to extract the foreground target from the captured scene. The RGB-D data of the target is sent over to the server and then rendered to accomplish dynamic scene reconstruction for remote 3D interaction.

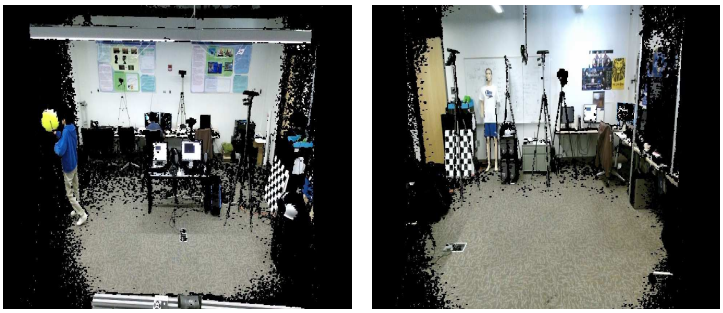
4.2.3 Experiments

4.2.3.1 Environmental Setup

In our 3D telepresence experiments, we use Microsoft Kinect V2 cameras to capture RGB-D images. The camera network that consists of 5 RGB-D cameras sparsely placed in a 10.2m $\times$ 6.4m space for physical environment reconstruction is shown in Figure 4.12. Another camera placed at a remote location captures the foreground target objects. The server is responsible for collecting RGB-D data from all the cameras and rendering the 3D environments. The total data bandwidth requirement for 6 cameras transmitting compressed RGB-D data to the server is 6.1MB per frame or 1,464 Mbps at 30fps. Same as our virtual mirror system, the server receives the



(a) (b) (c)



(d) (e)

Figure 4.12: The sparse camera network in the indoor environment. The camera in (b) faces the non-central region.

data in real-time with the popular 40/100 Gigabit Ethernet, and a local NTP time server is set up to synchronize all computers for camera network calibration. To simultaneously provide the results to end-to-end users, we use screen sharing software for viewing the dynamically reconstructed 3D scenes. Our proposed system is implemented by C/C++. OpenCV library is used for image processing and view transformation of 3D point clouds. For visualization, we use OpenGL library to render 3D scenes. To achieve real-time performance, the hardware setting for the server is Intel Core (TM) i7-5820k CPU at 3.30 GHz and 32GB of RAM with powerful GPU GTX-1080 to accelerate rendering speed. For each client, the hardware setting is Intel Core (TM) i7-4770s CPU at 3.1 GHz and 16GB of RAM.

4.2.3.2 Quantitative Evaluation

We evaluate the results of our camera network calibration quantitatively in this section. First, 5 RGB-D cameras in different poses are sparsely placed to cover the entire indoor environment. We simulated the non-overlapping situation illustrated in Figure 4.10. The camera pairs between each camera and the reference camera may not simultaneously observe ball movement in all the captured frames. During the initial camera calibration, we capture the 500 RGB-D images with different ball positions throughout the whole environment. Our neighboring camera searching method is used to find the initial extrinsics between the camera pairs. Then, the pseudo bundle adjustment is employed to optimize the extrinsics for each camera in the last calibration stage. Similar to the alignment testing experiments in Section 4.1.3.2, we apply the estimated extrinsics on another 5 RGB-D image sequences of sphere movement to validate the alignment accuracy. The root mean square error (RMSE) for the camera pairs had average of 3.32cm. As a result, the relative error over the entire captured environment was 0.58%. For remote target fusion, we randomly selected three different places to align their floors with the reconstructed environment. A plane fitting algorithm using RANSAC is applied to search a plane that best fits the given 3D points and the surface normal of the plane can be estimated. The results showed that the angle difference of the surface normals between each floor and the reference floor was less than 0.023 degrees. Thus, we can know the floors from the different places are successfully aligned at the same plane.

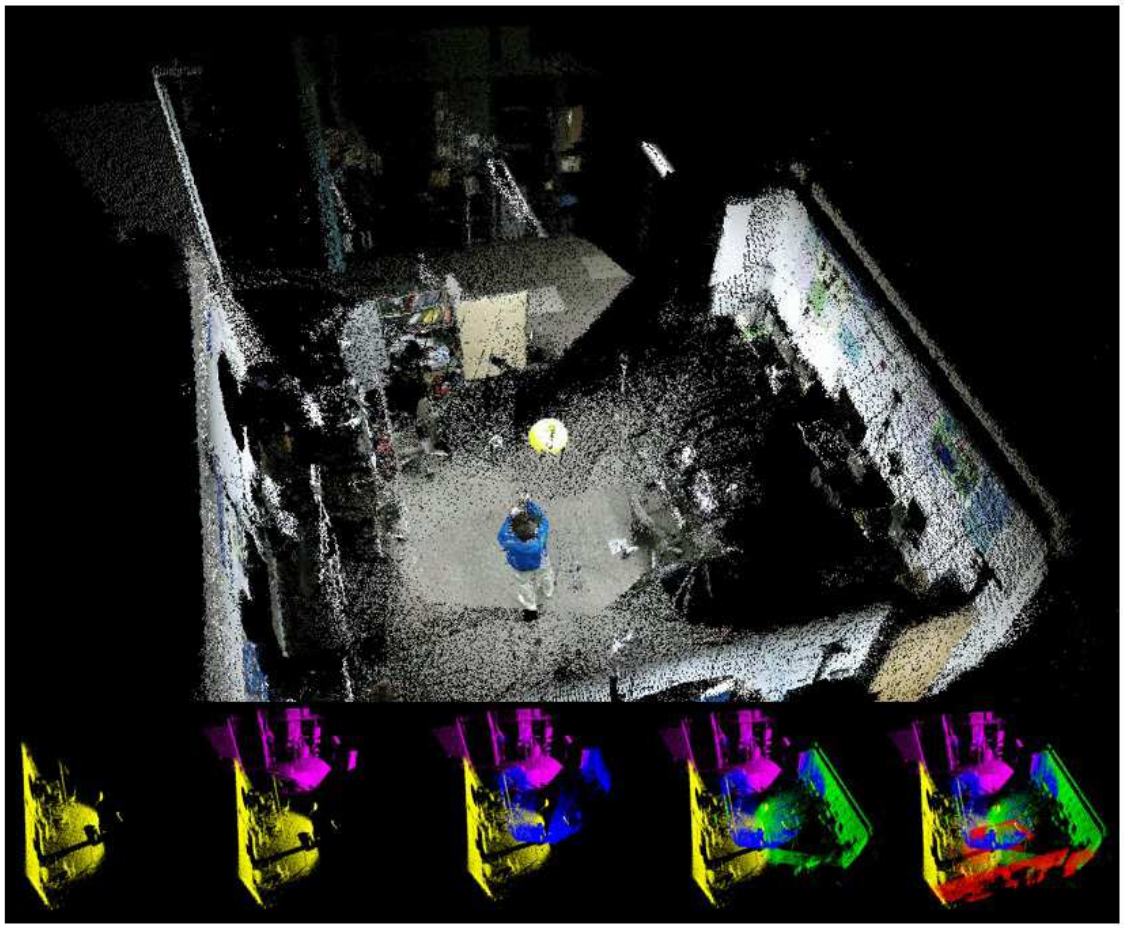
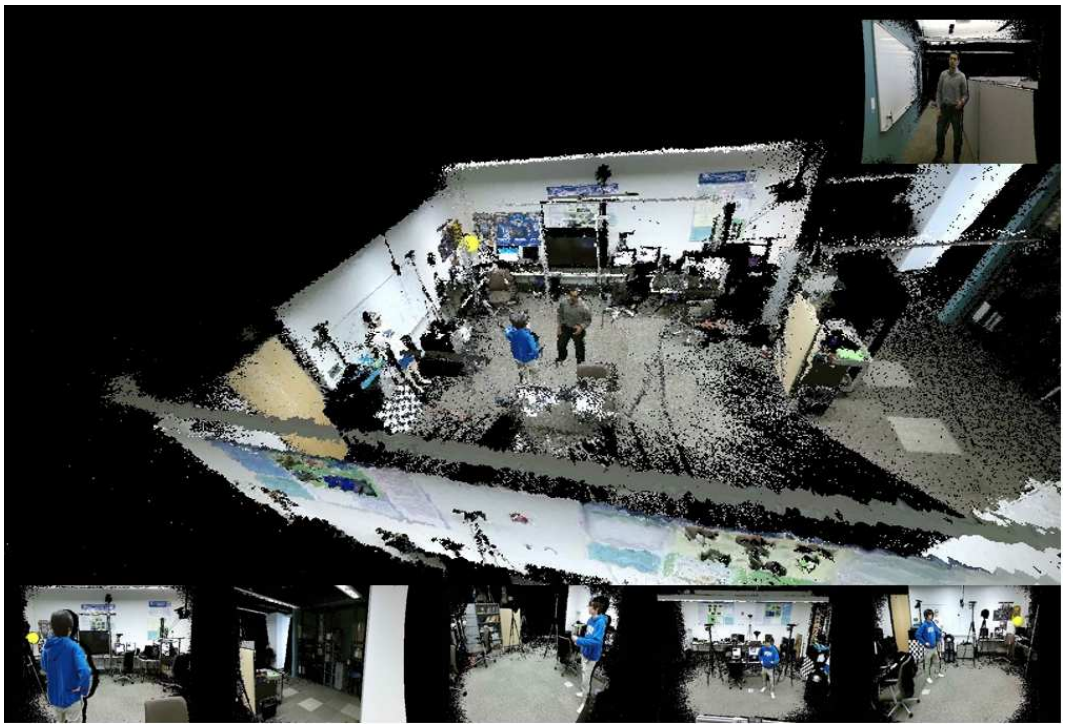


Figure 4.13: The reconstructed 3D environment by our Kinect v.2 camera network. Each camera view rendered by different color is shown in the bottom of the figure.

4.2.3.3 Qualitative Evaluation

We evaluate the quality of our reconstructed 3D scenes in this section. First, the top-down view of the reconstructed environment in 3D point clouds is shown in Figure 4.13. The field of view for each camera encoded by different color is in the bottom of the figure. From a visual standpoint, there is no misaligned issue in the overlapping regions - the person and the sphere ball are well-aligned. By merging the multiple camera views using our calibration scheme, the whole environment can be accurately generated in real-time. Note that background scenes may be occluded by



(a)

(b)

Figure 4.14: The results of 3D interactions for the geographically separated users (a)(b): The remote camera view is shown in the upper right of the figure and each camera view in the camera network is shown in the bottom of the figure.

foreground objects in images. Therefore, to further improve the rendering quality, we pre-capture the static background to fill the missing regions caused by the foreground objects and perform surface meshing to generate the 3D surfaces. The results of our reconstructed 3D environments after fusion with the remote target are shown in Figure 4.14. The camera view at the remote site is in the upper right of the image. The remote target is naturally merged into the reconstructed environment. Our system is scalable to bring multiple users from multiple places for remote 3D interactions.

Chapter 5 Conclusions and Future Work

In this dissertation, I have presented a real-time physical scene capture and rendering system with an efficiently calibrated RGB-D camera network. A fast and robust method for RGB-D camera network calibration using a spherical object has been proposed to address the issue of finding correspondences for a network of sparsely placed RGB-D cameras. The proposed 3D sphere center estimation scheme has been shown to produce a robust estimate of the sphere center trajectory. Compared with planar calibration objects, our solution is more reliable and requires less overlap between camera views. Using the observed sphere center trajectories at different camera views, we have tested four types of extrinsic camera calibrations including rigid transformation, manifold alignment, linear transformation and quadratic transformation, followed by a customized global refinement based on bundle adjustment. Our results have shown that linear transformation produced the best results by providing good alignment in the central view overlapping region and preserving geometric features in the peripheral area that has limited camera coverage. The proposed scheme has been implemented using a client-server architecture that enables rapid calibration and real-time 3D scene rendering.

To simulate a mirror that can reflect physical scenes, I have presented a novel system for rendering virtual arbitrarily-shaped curved mirror with multiple RGB-D cameras by using raytracing. The system can collect real 3D information provided by each camera and generate precise reflective scenes on virtual mirror surfaces in a large environment. The real-time implementation in our system can generate dy-

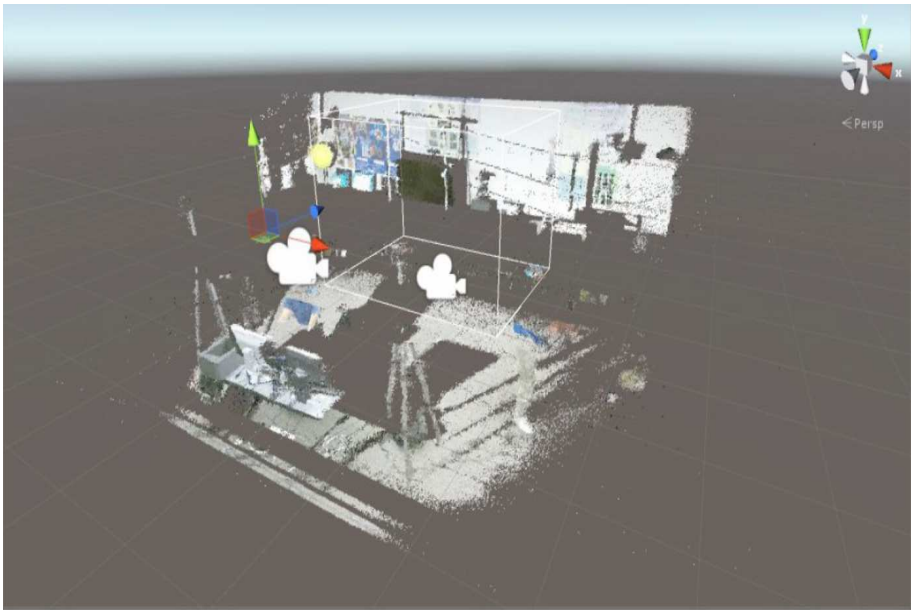


Figure 5.1: Reconstructed 3D physical environment applied to VR device.

namic scenes for users based on their viewpoints. The experimental results have demonstrated accurate calibration of up to 4 RGB-D cameras and real-time rendering of reflected scenes on different virtual curved mirrors. As an application of providing realistic interactive experience to geographically separated users, the 3D telepresence system based on our RGB-D camera network has been designed. The experimental results have shown that the remote user was naturally and accurately fused into a dynamically reconstructed environment for remote 3D interaction. With the ability to transmit and process large volumes of RGB-D data in real-time, the system is scalable to reconstruct a large-scale 3D scene and bring multiple users from remote places into the same environment.

For future work, I plan to extend our camera netowrk system to cover an area as large as an entire floor of the building, and to develop more data-efficient representations so as to scale the network to tens and hundreds of cameras. For physical

scene rendering, I am currently working on a number of extensions of the current system. First, the rendering quality could be improved through better surface meshing processes over the aggregated triangular meshes on dynamic foreground objects. Second, higher efficiency could be achieved through better searching process of where the reflected rays landed on the 3D scenes. Third, multiple reflections in the case of concave mirror need to be considered for more accurate rendering. Fourth, multiple viewers can be supported through the use of individual VR goggle displays as shown in Figure 5.1. By integrating Virtual Reality with a unique wide-area Kinect camera network, our system has potential to be a breakthrough in the world of augmented virtuality as mixed reality.

Bibliography

[1] J. Brekelmans and J. d. Mooij. Kinect lets you see yourself in vr. https://blogs.msdn.microsoft.com/kinectforwindows/2016/03/07/kinect-lets-you-see-yourself-in-vr-game/, 2016.

[2] Holoportation. virtual 3d teleportation in real-time. https://www.microsoft.com/en-us/research/project/holoportation-3/, 2014.

[3] J. Shen, P.-C. Su, S.-C. S. Cheung, and J. Zhao. Virtual mirror rendering with stationary rgb-d cameras and stored 3-d background. *IEEE Transactions on Image Processing*, pages 3433–3448, 2013.

[4] N. Uzuegbunam, W.H. Wong, S. Cheung, and L. Ruble. Mebook: Kinect-based self-modeling intervention for children with autism. In *Multimedia and Expo (ICME), 2014 IEEE International Conference on*, 2015.

[5] A. Kapoor. The return of the wizaed. http://www.huffingtonpost.com/yazmany-arboleda/a-candid-conversation-wit_1_b_791463.html, 2011.

[6] C. Herrera, J. Kannala, and J. Heikkil. Joint depth and color camera calibration with distortion correction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, pages 2058–2064, 2012.

[7] J. Shen, W. Xu, Y. Luo, P.-C. Su, and S.-C.S. Cheung. Extrinsic calibration for wide-baseline rgb-d camera network. In *Multimedia Signal Processing (MMSP), 2014 IEEE 16th International Workshop on*, pages 1–6, 2014.

[8] C. Wang and S. Mahadevan. A general framework for manifold alignment. In *AAAI Fall Symposium: Manifold Learning and Its Applications*, 2009.

[9] F. D. l. Torre, W.-S. Chu, X. Xiong, F. Vicente, X. Ding, and J. F. Cohn. Intraface. In *IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, 2015.

[10] Niantic. Pokemon go. http://www.pokemongo.com/, 2016.

[11] Statistics. Pokemon go. http://expandedramblings.com/index.php/pokemon-go-statistics/, 2016.

[12] HTC. Vive. https://www.htcvive.com/us/, 2016.

[13] Oculus. Rift. https://www.oculus.com/, 2016.

[14] IDC. Vr and ar headsets to hit 80 million by 2021. http://www.anews.com/us/p/77388111/, 2016.

[15] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6), 2007.

[16] K. Konolige and M. Agrawal. Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, pages 1066–1077, 2008.

[17] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, pages 559–568. ACM, 2011.

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

[19] F. Kahlesz, C. Lilge, and R. Klein. Easy-to-use calibration of multiple-camera setups. In *Workshop on Camera Calibration Methods for Computer Vision Systems (CCMVS2007)*, 2007.

[20] J. Puwein, R. Ziegler, J. Vogel, and M. Pollefeys. Robust multi-view camera calibration for wide-baseline camera networks. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 321–328. IEEE, 2011.

[21] T. Kuo, Z. Ni, S. Sunderrajan, and B.S. Manjunath. Calibrating a wide-area camera network with non-overlapping views using mobile devices. In *ACM Transactions on Sensor Networks (TOSN)*, volume 10, 2014.

[22] J. Zhu, L. Wang, R. Yang, J. Davis, and Z. Pan. Reliability fusion of time-of-flight depth and stereo geometry for high quality depth maps. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 33(7):1400–1414, 2011.

[23] H. Shim, R. Adelsberger, J. D. Kim, S.-M. Rhee, T. Rhee, J.-Y. Sim, M. Gross, and C. Kim. Time-of-flight sensor and color camera calibration for multi-view acquisition. *The Visual Computer*, pages 1139–1151, 2012.

[24] M. Hansard, G. Evangelidis, Q. Pelorson, and R. Horaud. Cross-calibration of time-of-flight and colour cameras. *Computer Vision and Image Understanding*, 134:105–115, 2015.

[25] P.-C. Su, J. Shen, and S.-C. S. Cheung. A robust rgb-d slam system for 3d environment with planar surfaces. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 275–279, 2013.

[26] A. T. Tran, K. Harada, et al. Depth-aided tracking multiple objects under occlusion. *Journal of Signal and Information Processing*, page 299, 2013.

[27] R. Macknojia, A. Chavez-Aragon, P. Payeur, and R. Laganiere. Calibration of a network of kinect sensors for robotic inspection over a large workspace. In *Robot Vision (WORV), 2013 IEEE Workshop on*, pages 184–190, 2013.

[28] R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 588–595, 2014.

[29] S. Hauswiesner, M. Straka, and G. Reitmayr. Virtual try-on through image-based rendering. *IEEE transactions on visualization and computer graphics*, pages 1552–1565, 2013.

[30] M. Yuan, I.R. Khan, F. Farbiz, S. Yao, A. Niswar, and M.-H Foo. A mixed reality virtual clothes try-on system. *Multimedia, IEEE Transactions on*, pages 1958–1968, 2013.

[31] T. Blum, V. Kleeberger, C. Bichlmeier, and N. Navab. mirracle: An augmented reality magic mirror system for anatomy education. In *Virtual Reality Short Papers and Posters (VRW), 2012 IEEE*, pages 115–116, 2012.

[32] J. Mercier-Ganady, F. Lotte, E. Loup-escande, M. Marchal, and A. Lecuyer. The mind-mirror: See your brain in action in your head using eeg and augmented reality. In *Virtual Reality (VR), 2014 iEEE*, pages 33–38, 2014.

[33] M.-Z. Poh, D.l McDuff, and R. Picard. A medical mirror for non-contact health monitoring. In *ACM SIGGRAPH 2011 Emerging Technologies*, pages 2:1–2:1, 2011.

[34] O. Erazo, J.A. Pino, R. Pino, and C. Fernandez. Magic mirror for neurorehabilitation of people with upper limb dysfunction using kinect. In *System Sciences (HICSS), 2014 47th Hawaii International Conference on*, pages 2607–2615, 2014.

[35] W. Andy Li and H. Fu. Augmented reflection of reality. In *ACM SIGGRAPH 2012 Emerging Technologies*, pages 3:1–3:1, 2012.

[36] J. Shen, S.C.S. Cheung, and J. Zhao. Virtual mirror by fusing multiple rgb-d cameras. In *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*. IEEE, 2012.

[37] H. Afzal, D. Aouada, D. Foni, B. Mirbach, and B. Ottersten. Rgb-d multi-view system calibration for full 3d scene reconstruction. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 2459–2464, 2014.

[38] M. Ruan and D. Huber. Calibration of 3d sensors using a spherical target. In *3D Vision (3DV), 2014 2nd International Conference on*, pages 187–193, 2014.

[39] W. Kumara, S.-H. Yen, H.-H. Hsu, T. K. Shih, W.-C. Chang, and E. Togootogtokh. Real-time 3d human objects rendering based on multiple camera details. *Multimedia Tools and Applications*, 76(9):11687–11713, 2017.

[40] Z. Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1330–1334, 2000.

[41] N. Ahmed and I. N. Junejo. A system for 3d video acquisition and spatio-temporally coherent 3d animation reconstruction using multiple rgb-d cameras. *Proceedings of IJSIP*, pages 113–128, 2013.

[42] W. Lemkens, P. Kaur, K. Buys, P. Slaets, T. Tuytelaars, and J. D. Schutter. Multi rgb-d camera setup for generating large 3d point clouds. In *IROS*, pages 1092–1099, 2013.

[43] X. Chen and J. Davis. Wide area camera calibration using virtual calibration objects. In *Proceedings of CVPR*, pages 520–527, 2000.

[44] T. Svoboda, D. Martinec, and T. Pajdla. A convenient multicamera self-calibration for virtual environments. *Presence: Teleoper. Virtual Environ.*, pages 407–422, 2005.

[45] D. S. Alexiadis, D. Zarpalas, and P. Daras. Real-time, full 3-d reconstruction of moving foreground objects from multiple consumer depth cameras. *Multimedia, IEEE Transactions on*, pages 339–358, 2013.

[46] A. Irschara, C. Zach, and H. Bischof. Towards wiki-based dense city modeling. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.

[47] C. Kuster, T. Popa, C. Zach, C. Gotsman, M. H. Gross, P. Eisert, J. Hornegger, and K. Polthier. Freecam: A hybrid camera system for interactive free-viewpoint video. In *VMV*, pages 17–24, 2011.

[48] R. K. Kumar, A. Ilie, J.-M. Frahm, and M. Pollefeys. Simple calibration of non-overlapping cameras with a mirror. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7, 2008.

[49] J. H. Kim and B. K. Koo. Convenient calibration method for unsynchronized camera networks using an inaccurate small reference object. In *Opt. Express 20*, pages 25292–25310, 2012.

[50] J. Park, H. Kim, Y. W. Tai, M.S. Brown, and I. Kweon. High quality depth map upsampling for 3d-tof cameras. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1623–1630. IEEE, 2011.

[51] C. Herrera, J. Kannala, and J. Heikkil. Accurate and practical calibration of a depth and color camera pair. *Proc. 14th Int. Conf. Comput. Anal. Images Patterns*, pages 437–445, 2011.

[52] W. Liu, Y. Fan, Z. Zhong, and T. Lei. A new method for calibrating depth and color camera pair based on kinect. In *Audio, Language and Image Processing (ICALIP), 2012 International Conference on*, pages 212–217, 2012.

[53] Y. Kim, D. Chan, C. Theobalt, and S. Thrun. Design and calibration of a multi-view tof sensor fusion system. *CVPR Workshop on time-of-light Camera based Computer Vision*, pages 1–7, 2008.

[54] C. Zhang and Z. Zhang. Calibration between depth and color sensors for commodity depth cameras. *Proc. IEEE ICME*, pages 1–6, 2011.

[55] S. N. Sinha, M. Pollefeys, and L. McMillan. Camera network calibration from dynamic silhouettes. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, pages I–195–I–202 Vol.1, 2004.

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

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

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

[59] T. D. A. Prasad, K. Hartmann, W. Wolfgang, S. E. Ghobadi, and A. Sluiter. First steps in enhancing 3d vision technique using 2d/3d sensors. In *11. Computer Vision Winter Workshop 2006*, 2006.

[60] OpenNI. Open natural interaction. `http://structure.io/openni/`, 2011.

[61] D. S. Ly, C. Demonceaux, P. Vasseur, and C. Pegard. Extrinsic calibration of heterogeneous cameras by line images. *Machine Vision and Applications*, pages 1601–1614, 2014.

[62] M. Bruckner, F. Bajramovic, and J. Denzler. Intrinsic and extrinsic active self-calibration of multi-camera systems. *Mach. Vis. Appl.*, pages 389–403, 2014.

[63] E. Fernandez-Moral, J. Gonzalez-Jimenez, P. Rives, and V. Arevalo. Extrinsic calibration of a set of range cameras in 5 seconds without pattern. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 429–435, 2014.

[64] R. S. Yang, Y. H. Chan, R. Gong, M. Nguyen, A. G. Strozzi, P. Delmas, G. Gimel'farb, and R. Ababou. Multi-kinect scene reconstruction: Calibration and depth inconsistencies. In *Image and Vision Computing New Zealand (IVCNZ), 2013 28th International Conference of*, pages 47–52. IEEE, 2013.

[65] A. N. Staranowicz, G. R. Brown, F. Morbidi, and G.-L. Mariottini. Practical and accurate calibration of rgb-d cameras using spheres. *Comput. Vis. Image Underst.*, pages 102–114, 2015.

[66] A. N. Staranowicz, C. Ray, and G.-L. Mariottini. Easy-to-use, general, and accurate multi-kinect calibration and its application to gait monitoring for fall prediction. In *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*, pages 4994–4998. IEEE, 2015.

[67] P. Eisert, P. Fechteler, and J. Rurainsky. 3-D Tracking of Shoes for Virtual Mirror Applications. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2008. CVPR 2008.

[68] A. Jimeno-Morenilla, J. L. Sánchez-Romero, and F. Salas-Pérez. Augmented and virtual reality techniques for footwear. *Computers in Industry*, 2013.

[69] Y.-I. Yang, C.-K. Yang, X.-L. Liao, and C.-H. Chu. Virtual try-on of footwear in augmented reality using rgb-d cameras. In *Industrial Engineering and Engineering Management (IEEM), 2015 IEEE International Conference on*. IEEE, 2015.

[70] A. Hilsmann and P. Eisert. Realistic cloth augmentation in single view video. In *Proc. of Vision, Modeling, and Visualization Workshop*, 2009.

[71] L. Wang, R. Villamil, S. Samarasekera, and R. Kumar. Magic mirror: A virtual handbag shopping system. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 19–24. IEEE, 2012.

[72] D. Saakes, H.-S. Yeo, S.-T. Noh, G. Han, and W. Woo. Mirror mirror: An on-body t-shirt design system. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2016.

[73] J. Fu, Y. Liu, J. Jia, Y. Ma, F. Meng, and H. Huang. A virtual personal fashion consultant: Learning from the personal preference of fashion. In *AAAI*, pages 5087–5088, 2017.

[74] S. Hauswiesner, M. Straka, and G. Reitmayr. Free viewpoint virtual try-on with commodity depth cameras. In *Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry*, pages 23–30. ACM, 2011.

[75] M. Straka, S. Hauswiesner, M. Rüther, and H. Bischof. A free-viewpoint virtual mirror with marker-less user interaction. In *Scandinavian Conference on Image Analysis*, pages 635–645, 2011.

[76] S. Baker and S. K. Nayar. A theory of single-viewpoint catadioptric image formation. *International Journal of Computer Vision*, 1999.

[77] L. Szirmay-Kalos, B. Aszdi, I. Laznyi, and M. Premecz. Approximate ray-tracing on the gpu with distance impostors. In *Computer Graphics Forum (Proceedings of Eurographics 2005)*, 2005.

[78] F. Bravo-Valenzuela and M. Torres-Torriti. Comparison of panoramic stereoscopic sensors based on hyperboloidal mirrors. In *2009 6th Latin American Robotics Symposium (LARS 2009)*, pages 1–8, 2009.

[79] A. Moroni, S. Cunha, J. Ramos, and J. Manzolli. Sonifying robotic trajectories with a spherical omnidirectional vision system in the aural environment. In *NICS Reports*, 2009.

[80] A. Akin, O. Cogal, K. Seyid, H. Afshari, A. Schmid, and Y. Leblebici. Hemispherical multiple camera system for high resolution omni-directional light field imaging. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 3(2):137–144, 2013.

[81] B. E. Madeira and L. Velho. Virtual table – teleporter: Image processing and rendering for horizontal stereoscopic display. In *2012 14th Symposium on Virtual and Augmented Reality*, pages 1–9, 2012.

[82] R. Swaminathan, M. D. Grossberg, and S. K. Nayar. Non-single viewpoint catadioptric cameras: Geometry and analysis. *International Journal of Computer Vision*, pages 211–229, 2006.

[83] W. Wang and Z. Cao. A cylindrical virtual space based catadioptric real-time panorama imaging system. In *Robotics, Automation and Mechatronics, 2006 IEEE Conference on*, pages 1–5. IEEE, 2006.

[84] W.-J. Lee, Y. Shin, J. Lee, J. Kim, J. Nah, S. Jung, S. Lee, H. Park, and T. Han. Sgrt: a mobile gpu architecture for real-time ray tracing. In *Proceedings of the 5th high-performance graphics conference*, pages 109–119. ACM, 2013.

[85] Y. Gotanda, M. Kawase, and M. Kakimoto. Real-time rendering of physically based optical effects in theory and practice. In *ACM SIGGRAPH 2015 Courses*, SIGGRAPH '15, pages 23:1–23:14. ACM, 2015.

[86] H. Joo, S. Kwon, S. Lee, E. Eisemann, and S. Lee. Efficient ray tracing through aspheric lenses and imperfect bokeh synthesis. *Computer Graphics Forum*, pages 99–105, 2016.

[87] P. Andrade, T. Sabino, and E. Clua. Towards a heuristic based real time hybrid rendering a strategy to improve real time rendering quality using heuristics and ray tracing. In *Computer Vision Theory and Applications (VISAPP), 2014 International Conference on*, pages 12–21. IEEE, 2014.

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

[89] M. Knecht, C. Traxler, C. Winklhofer, and M. Wimmer. Reflective and refractive objects for mixed reality. *IEEE transactions on visualization and computer graphics*, pages 576–582, 2013.

[90] J. E. F. Lindoso, L. S. Figueiredo, V. Teichrieb, R. A. Roberto, and R. F. dos Anjos Filho. Integrated pipeline for natural interaction with photorealistic rendering. In *Virtual and Augmented Reality (SVR), 2013 XV Symposium on*, pages 125–133. IEEE, 2013.

[91] A. L. Miguel, A. C. Nogueira, and N. Goncalves. Real-time 3-d visualization of accurate specular reflections in curved mirrors a gpu implementation. In *2014 International Conference on Computer Graphics Theory and Applications (GRAPP)*, pages 1–8, 2014.

[92] X. Lu, J. Shen, S. Perugini, and J. Yang. An immersive telepresence system using rgb-d sensors and head mounted display. In *2015 IEEE International Symposium on Multimedia (ISM)*, pages 453–458. IEEE, 2015.

[93] W. Wu, A. Arefin, Z. Huang, P. Agarwal, S. Shi, R. Rivas, and K. Nahrstedt. "i'm the jedi!" - a case study of user experience in 3d tele-immersive gaming. In *2010 IEEE International Symposium on Multimedia*, pages 220–227, Dec 2010.

[94] O.-H. Kwon, S.-Y. Koo, Y.-G. Kim, and D.-S. Kwon. Telepresence robot system for english tutoring. In *Advanced Robotics and its Social Impacts (ARSO), 2010 IEEE Workshop on*, pages 152–155. IEEE, 2010.

[95] A. Maimone and H. Fuchs. Encumbrance-free telepresence system with real-time 3d capture and display using commodity depth cameras. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 137–146. IEEE, 2011.

[96] T. Blum, V. Kleeberger, C. Bichlmeier, and N. Navab. mirracle: An augmented reality magic mirror system for anatomy education. In *Virtual Reality Short Papers and Posters (VRW), 2012 IEEE*, pages 115–116. IEEE, 2012.

[97] C. Kamphuis, E. Barsom, M. Schijven, and N. Christoph. Augmented reality in medical education? *Perspectives on medical education*, pages 300–311, 2014.

[98] W. Matusik and H. Pfister. 3d tv: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. *ACM Transactions on Graphics (TOG)*, pages 814–824, 2004.

[99] J. Edelmann, P. Gerjets, P. Mock, A. Schilling, and W. Strasser. Face2facea system for multi-touch collaboration with telepresence. In *Emerging Signal Processing Applications (ESPA), 2012 IEEE International Conference on*, pages 159–162. IEEE, 2012.

[100] M. Gross, S. Würmlin, M. Naef, E. Lamboray, C. Spagno, A. Kunz, E. Koller-Meier, T. Svoboda, L. Van Gool, S. Lang, et al. blue-c: a spatially immersive display and 3d video portal for telepresence. In *ACM Transactions on Graphics (TOG)*, pages 819–827. ACM, 2003.

[101] G. Kurillo, R. Bajcsy, K. Nahrsted, and O. Kreylos. Immersive 3d environment for remote collaboration and training of physical activities. In *Virtual Reality Conference, 2008. VR'08. IEEE*, pages 269–270. IEEE, 2008.

[102] R. Vasudevan, G. Kurillo, E. Lobaton, T. Bernardin, O. Kreylos, R. Bajcsy, and K. Nahrstedt. High-quality visualization for geographically distributed 3-d teleimmersive applications. *IEEE Transactions on Multimedia*, pages 573–584, 2011.

[103] D. S. Alexiadis, D. Zarpalas, and P. Daras. Real-time, realistic full-body 3d reconstruction and texture mapping from multiple kinects. In *IVMSP Workshop, 2013 IEEE 11th*, pages 1–4. IEEE, 2013.

[104] P.-A. Blanche, A. Bablumian, R. Voorakaranam, C. Christenson, W. Lin, T. Gu, D. Flores, P. Wang, W.-Y. Hsieh, M. Kathaperumal, et al. Holographic three-dimensional telepresence using large-area photorefractive polymer. *Nature*, pages 80–83, 2010.

[105] T. Balogh and P. T. Kovács. Real-time 3d light field transmission. In *SPIE Photonics Europe*, pages 772406–772406. International Society for Optics and Photonics, 2010.

[106] S. Orts-Escolano, C. Rhemann, S. Fanello, W. Chang, A. Kowdle, Y. Degtyarev, D. Kim, P. L. Davidson, S. Khamis, M. Dou, et al. Holoportation: Virtual 3d teleportation in real-time. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 2016.

[107] NTP. The network time protocol. `http://www.ntp.org/`, 2014.

[108] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[109] J. Smisek, M. Jancosek, and T. Pajdla. 3d with kinect. In *2011 IEEE International Conference on Computer Vision Workshops. Barcelona: Institute of Electrical and Electron-ics Engineers— nc*, pages 1154–1160, 2011.

[110] N. Burrus. Kinect calibration. `http://nicolas.burrus.name/index.php/Research/KinectCalibration`. Accessed February 5, 2017.

[111] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, pages 381–395, 1981.

[112] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 698–700, 1987.

[113] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Vision Algorithms: Theory and Practice*, 1999.

[114] K. LEVENBERG. A method for the solution of certain non-linear problems in least squares. In *Quart. Appl. Math*, pages 164–168, 1944.

[115] D. MARQUARDT. An algorithm for the least-squares estimation of nonlinear parameters. In *SIAM J. Appl. Math*, pages 431–441, 1963.

[116] H. Sarbolandi, D. Lefloch, and A. Kolb. Kinect range sensing: Structured-light versus time-of-flight kinect. *Computer Vision and Image Understanding*, pages 1–20, 2015.

[117] R. A. Fisher. Statistical methods and scientific inference. 1956.

[118] C. Sigg, T. Weyrich, M. Botsch, and M. Gross. Gpu-based ray-casting of quadratic surfaces. In *Proceedings of the 3rd Eurographics / IEEE VGTC Conference on Point-Based Graphics*, pages 59–65. Eurographics Association, 2006.

[119] J. Dunlap. Queue-linear flood fill: A fast flood fill algorithm. `http://www.codeproject.com/KB/GDI-plus/queuelinearfloodfill.aspx`, 2006.

Vita

Po-Chang Su

Education:

- Ph.D., Electrical and Computer Engineering, University of Kentucky, USA, 2017
- M.S., Electrical and Computer Engineering, University of Kentucky, USA, 2013
- B.S., Electrical Engineering, Yuan Ze University, Taiwan, 2007

Publications:

Journal Articles

1. **P.-C. Su**, J. Shen, W. Xu, S.-C. Cheung and Y. Luo, "A Fast and Robust Extrinsics Calibration for RGB-D Camera Network", **Sensors, 2017, (under review)**.
2. J. Shen, **P.-C. Su**, S.-C. Cheung, and J. Zhao, "Virtual Mirror Rendering With Stationary RGB-D Cameras and Stored 3-D Background" **IEEE Transactions on Image Processing vol. 22, issue 9, pp. 1-16, (TIP 2013)**.

Conferences Proceeding Papers

1. **P.-C. Su**, W. Xu, J. Shen and S.-C. Cheung, "Real-Time Rendering of Physical Scene On Virtual Curved Mirror with RGB-D Camera Network", **IEEE International Workshop on Hot Topics in 3D Multimedia (ICMEW), 2017 (oral)**.
2. **P.-C. Su**, J. Shen, and M. U. Rafique, "RGB-D Camera Network Calibration and Streaming for 3D Telepresence in Large Environment", **IEEE International Conference on Multimedia Big Data (BigMM), 2017 (oral)**.
3. W. Xu, **P.-C. Su**, and S.-C. Cheung, "Human pose estimation using two RGB-D sensors", **IEEE International Conference on Image Processing (ICIP), 2016**.
4. J. Shen, W. Xu, Y. Luo, **P.-C. Su**, and S.-C. Cheung, "Extrinsic Calibration for Wide-baseline RGB-D Camera Networks", **IEEE International Workshop on Multimedia Signal Processing (MMSP), 2014**.

5. **P.-C. Su**, J. Shen, and S.-C. Cheung, "A Robust RGB-D SLAM System for 3D Environment with Planar Surfaces", **IEEE International Conference on Image Processing (ICIP), 2013**.

Professional Activities

- Reviewers for
  - IEEE Transactions on Multimedia (TMM) 2017
  - IEEE International Conference on Image Processing (ICIP) 2014-2016
  - IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2015-2017
  - IEEE International Conference on Multimedia and Expo (ICME) 2016
  - IEEE International Symposium on Circuits and Systems (ISCAS) 2017