2014

Computational Multimedia for Video Self Modeling

Ju Shen
University of Kentucky, jushen.tom@gmail.com

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.

Ju Shen, Student

Dr. Judy Goldsmith, Major Professor

Dr. Miroslaw Truszczynski, Director of Graduate Studies

COMPUTATIONAL MULTIMEDIA FOR VIDEO SELF MODELING

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

Ju Shen

Lexington, Kentucky

Co-Directors: Dr. Judy Goldsmith, Professor of Computer Science

and: Dr. Sen-ching Samson Cheung, Professor of Electrical and

Computer Engineering

Lexington, Kentucky

2014

ABSTRACT OF DISSERTATION

COMPUTATIONAL MULTIMEDIA FOR VIDEO SELF MODELING

Video self modeling (VSM) is a behavioral intervention technique in which a learner models a target behavior by watching a video of oneself. This is the idea behind the psychological theory of self-efficacy – you can learn or model to perform certain tasks because you see yourself doing it, which provides the most ideal form of behavior modeling. The effectiveness of VSM has been demonstrated for many different types of disabilities and behavioral problems ranging from stuttering, inappropriate social behaviors, autism, selective mutism to sports training. However, there is an inherent difficulty associated with the production of VSM material. Prolonged and persistent video recording is required to capture the rare, if not existed at all, snippets that can be used to string together in forming novel video sequences of the target skill. To solve this problem, in this dissertation, we use computational multimedia techniques to facilitate the creation of synthetic visual content for self-modeling that can be used by a learner and his/her therapist with a minimum amount of training data. There are three major technical contributions in my research. First, I developed an Adaptive Video Re-sampling algorithm to synthesize realistic lip-synchronized video with minimal motion jitter. Second, to denoise and complete the depth map captured by structure-light sensing systems, I introduced a layer based probabilistic model to account for various types of uncertainties in the depth measurement. Third, I developed a simple and robust bundle-adjustment based framework for calibrating a network of multiple wide baseline RGB and depth cameras.

KEYWORDS: Video Self Modeling, Computational Multimedia, RGB-D System, Depth Sensors, Virtual Reality

Author's signature: Ju Shen

Date: July 9, 2014

COMPUTATIONAL MULTIMEDIA FOR VIDEO SELF MODELING

By

Ju Shen

Co-Directors of Dissertation: Judy Goldsmith and Sen-ching Samson Cheung

Director of Graduate Studies: Miroslaw (Mirek) Truszczynski

Date: July 9, 2014

ACKNOWLEDGMENTS

I would never have been able to finish my dissertation without the guidance of my committee members, help from friends, and support from my family.

I am heartily thankful to Dr. Sen-Ching Samson Cheung for the encouragement, guidance and strong support in my research. I also would like to express my sincere gratitude to Dr. Judy Goldsmith for that she has given me many advices and support during the last five years. Finally, I give many thanks to my committee members and outside examiner: Dr. Ruigang Yang, Dr. Nathan Jacobs, Dr. Lisa Ruble and Dr. Kenneth Kubota, for the time they spent on my dissertation review. I appreciate all of the suggestions and comments.

My deepest gratitude goes to the people who have had such a significant impact on my life. My parents encouraged me to enter a Ph.D. program in the United States. They have always given me unconditional support and love.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, one can learn just about anything by watching a video on the web, on television or from the thousands of DVD/Blue-ray titles available from different sources. Watching a video to learn or model a target positive behavior is in fact a well-studied technique in behavior therapy called Video Modeling (VM) interventions. They are widely used in rehabilitation and education of patients recovering from surgery [4] and cancer [5] as well as job and safety training for hospital staffs [6] and office workers [7]. VM is also effective in a school setting to teach children and young adolescents various skills including social interactions, communication, self-monitoring and emotional regulation [8].

Rather than watching others, some researchers have argued that we can learn even more effectively by watching our own positive behaviors. Such form of self modeling is classically done with a mirror and one of the most prominent examples is the use of the "mirror box" in treating phantom limb pain among amputees [9]. A mirror box is a simple box with a mirror inside it, separating the box into two vertical chambers. The top part of the chamber behind the mirror is covered and there is an opening in the front part of each chamber. When a person puts both of his/her arms into the openings, the mirror image creates an illusion of the opposite arm. It

is this illusion that creates visual feedback to the brain that alleviates the phantom pain caused by the amputated arm. While this form of mirror visual feedback is still not fully understood, it has already helped a countless number of patients and has significant implications in our understanding of the elasticity of the adult human brains ability to rewiring itself [9]. Seeing or visualizing oneself accomplishing the target behavior provides the most ideal form of behavior modeling. Though still in its early development, effectiveness of VSM has been demonstrated for many different types of disabilities and behaviorial problems ranging from stuttering, inappropriate social behaviors, autism, selective mutism to sports training. A summary of this research can be found in [10].

There are two forms of VSM: positive self-review and feedforward [11]. In positive self-review, the portions of the recorded video showing poorly executed routines are removed leaving only the positive target behaviors. The resulting video will be reviewed to enhance fluency of the skills that have already been acquired by the learner but not yet perfected. On the other hand, the feed-forward VSM focuses on teaching new skills to a learner by showing novel skills that have never been observed but still within the reach of the learner. Evidence shows that the feed-forward approach delivers a more dramatic learning effect than the positive self-review approach [10]. One explanation of these findings comes from the psychological theory of self-efficacy - "I know I can because I have done it before" [12]. There is an inherent difficulty associated with the production of VSM material. Prolonged and persistent video recording is required to capture the rare, if existed at all, snippets that can be used to string together in forming novel video sequences of the target skill. An example

of feedforward VSM can be found in [10] – the author records more than six hours of video from a child who can only speak one or two-word utterances to produce a two-minute clip of the same child saying one full sentence.

This problem can be potentially solved by using computational multimedia techniques. From computer generated imagery to speech synthesis, there exist a myriad of multimedia tools that can synthesize realistic video content. The goal is to adapt such tools for the development of feedforward VSM systems that can be used by a learner and his/her therapist in creating VSM contents with minimum amount of training data. For such system to be useful in practice, the synthesis process must be automatic and real-time so that rapid feedback can be provided. The synthetic content should be perceptually indistinguishable from real video footage. Such systems can have a significant impact in reducing the time and effort to achieve the target learning objectives. The goal of the research is to develop new technologies to facilitate the creation of synthetic visual contents for self-modeling, and combine the synthetic contents with 3D sensing to render a mirror-like feedback on novel display devices.

1.1 Problem Statement

In this dissertation, I study two problems by using computation multimedia for video self modeling. The first problem is on how to synthesize novel audio-visual contents for video self modeling. The second is on how to render self-modeled feed-forward video, in real-time, through an immersive display system so as to reduce the time gap between the emergence of the target behavior and the availability of the self-modeled

video. In order to provide a context for these research problems, we have chosen specific applications and focuses technological development for these applications. For the first problem, we have chosen to design a feedforward VSM content production system called SpeakToMe for patients suffering from vocal hyperfunction. For the second problem, we have developed a virtual mirror rendering system using a network of commodity structured-light RGB-D cameras that would be particularly beneficial to children with autism spectrum disorder (ASD). While we have extensively collaborated with domain experts in this area, this dissertation focuses on the development of the technology and defers the confirmation of the clinica values of these technology for future research.

1.1.1 VSM for Voice Disorder

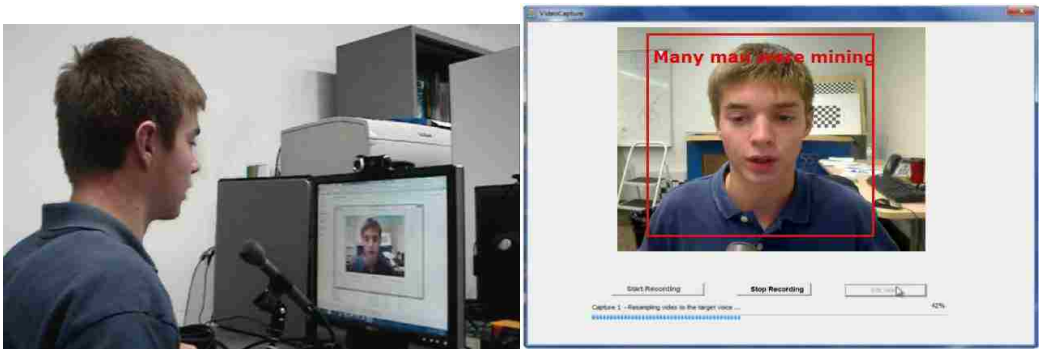
We designed a novel feedforward VSM content production system for patients suffering from a specific type of vocal disorders called vocal hyperfunction. Vocal hyperfunction refers to the use of excessive muscle force and physical effort in the production of voice, and usually requires a prolonged period of speech therapy. Long-standing untreated voice disorders can detrimentally affect an individual psychosocially and academically and can be a source of substantial economic cost to society in terms of higher health care costs.

The proposed video self modeling approach can be used for voice therapy in individuals with vocal fold nodules, functional dysphonia. It can also be used in speech therapy for post-surgical management of individuals with vocal fold polyps and vocal fold cysts. However, before subjecting the approach of video self modeling to

empirical testing with traditional voice therapy approaches, it is critical to test the robustness/accuracy of the image processing algorithm. The goal of our system is to reduce the amount of time on therapy using the principle of video self modeling. Our system records a video of a patient at the clinic reciting a known script, and synthesizes a new video with a "healthier" voice for self-modeling. To achieve lip-synchronization, the original video needs to be re-sampled to match the pace of the replacing voice. Significant up-sampling or down-sampling creates unevenness in motion or motion jitter, making the resulting video unnatural. Furthermore, consistent segmentation across speakers remains a very challenging problem. Our system uses a data-driven approach in selecting a replacement speech best resembled that of the patient and apply adaptive re-sampling to preserve the nation motion in the synthesized video.

Figure 1.1a shows the setup of the system. It captures the raw video of the patient through a web camera situated on top of a desktop computer. Figure 1.1b shows the user interface. A red square is shown in the middle of the screen to provide a visual cue to anchor the head position. In order to capture a proper eye gaze, the left-to-right scrolling script is shown near the camera.

Figure 1.2 shows the audiovisual analysis and the VSM content generation process. After the raw video is captured, the audio track is extracted. The audio is segmented to extract time markers corresponding to the phone boundaries. The system then generates a replacement speech using either perceptually similar pre-recorded healthy voice or text-to-speech synthesis. The merits of both methods will be studied in the experimental section. Time markers for phone boundaries in the replacement speech



(a) (b)

Figure 1.1: (a) System Setup; (b) User Interface

will also be identified using the same segmentation module. Both sets of markers are needed to align the video track with the replacement speech in order to minimize motion jitter and provide lip-synchronization. Frame interpolation is then applied to re-sample the video track which is then combined with the new speech track. Our audio-visual synchronization process is described in Section 3.3 while the generation of the replacement speech and video re-sampling are presented in Section 3.4.

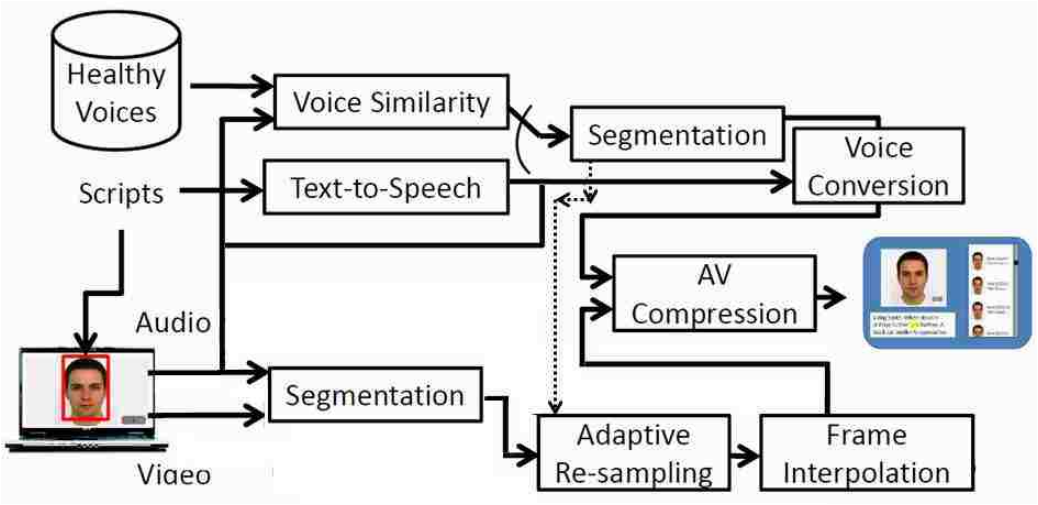


Figure 1.2: VSM Contenet Generation

1.1.2 VSM for Children with ASD

We are all familiar with the use of a mirror in fashion, cosmetics and plastic surgery. The use of a real mirror is often cumbersome because the user needs to physically change her appearance which is not always possible. The use of image or video manipulation software allows a much greater flexibility but lacks the real-time feedback of a mirror. The marriage between a mirror-like display and computer generated graphics presents the holy grail in such applications, and prototype systems have already been developed for these purposes [13][14][15].

The connection between mirror images and our cognitive functions, however, goes far beyond mere aesthetics. Perhaps the most well-known example is the use of the "mirror box" in helping patients suffering from chronic neurological disorders such as phantom pain, hemiparesis from stroke, and other complex regional pain syndrome [9]. A mirror box is a simple top-open box with an opening on the side through which the patient can put her arm inside. There is a mirror in the middle facing the inserted arm and the mirror image creates an illusion of the presence of the opposite arm. It is this illusion that creates a visual feedback to the brain that alleviates the phantom pain caused by the amputated arm. While this form of mirror visual feedback is still not fully understood, it has already helped countless number of patients and has significant implication in our understanding of the elasticity of adult human brain in rewiring itself [16].

Equally mysterious is the special affinity towards self images among children on the autistic spectrum. Autism affects 1 out of 110 children and it is one of the fastest

growing development disorders in the United States [17]. Autistic children have significant delay in development and typically have poor eye gaze when interacting with others. Recent fMRI studies have confirmed comparatively low activity in the brains of autistic children when viewing pictures of human faces even of their family members, while maintaining high level of activity when viewing pictures of themselves [18]. While a full neurological explanation of self recognition is still under investigation, there is little doubt that the omnipresence of mirrors and other reflective surfaces in the environments as well as the instant visual feedback from the mirror image are important contributing factors. Both of these studies in neurology strongly suggest that self images can very useful in the rehabilitation and education of people suffering from various types of neurological disorder. In fact, one of the prominent behavioral therapies, the so-called Video Self Modeling or VSM, use multimedia editing tools to splice unrelated footage together in creating videos of the autistic child herself seemingly accomplishing tasks that are beyond her immediate capability and use them as teaching material [18]. There is little doubt that a "programmable" mirror-like display can provide the flexibility in creating visual contents and the instant feedback to the patients that go far beyond the rigid video editing tools available today.

Simulating a mirror is not simple. The naive 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 image depends on the position of the viewer. Thus, one challenge of simulating the mirror is to render different content on the display depending on the viewer's perspective. To simulate a large mirror 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 viewpoints, render new view based on the position of a virtual mirror and possibly add new visual content that are compatible with the scene geometry. Furthermore, it must be able to accomplish all these tasks in real-time and with extremely high fidelity otherwise it loses the instant visual feedback required to provide the realism of a mirror.

For this reason, a virtual mirror rendering system is developed using a network of commodity structured-light RGB-D cameras. The depth information provided by the RGB-D cameras can be used to track the viewpoint and render the scene dynamically based on the viewer's prospectives. To speed up the overall performance, a scalable client-and-server architecture is used with the 3-D point cloud processing, the viewpoint estimate, and the mirror image rendering are all done on the client side. The mirror image and the viewpoint estimate are then sent to the server for final mirror view synthesis and viewpoint refinement. Figure 5.1 shows the hardware setting.

Figure 1.3: System Setting: three Kinects are mounted around the display for 3D data catpuring.

1.2 Contributions of Dissertation

The research work presented in this dissertation addresses the challenges of Video Self Modeling and develops novel multimedia processing algorithms for improving video synthesis and 2D/3D rendering that can be applied for many potential VSM applications. Specifically,

1. To achieve automatic speech replacement on the captured videos of patients with voice disorder, we have introduced a novel audiovisual algorithm that combines audio segmentation with lip-state detection [19]. It allows us to ac-

curate identify corresponding time markers in the audio and video tracks. Different from traditional lip or speech recognition methods that use model fitting together with probabilistic algorithm [20][21], such as Hidden Markov Model (HMM) based approach [22], to recognize specific phonics, our method only focus on the alignment of the original speech signal and the replacement speech signal by using both audio and visual information. This can avoid additional training steps in our algorithm. Furthermore, to generate realistic re-sampled video, we have introduced a novel adaptive sampling strategy that minimizes the amount of motion jitter and preserves the spatial sharpness [19]. Our experimental results have shown that natural human voice selected through speaker similarity provides the subjectively best results.

2. Realistic simulation of a mirror is challenging as it requires accurate viewpoint tracking and rendering, wide-angle viewing of the environment, as well as real-time performance to provide immediate visual feedback. Different from previous work, our system provides a realistic mirror visual effect by incorporating three key features: viewpoint dependent content, wide field of view, and 3D based rendering [23]. Existing work are limited in either one or some of these features, which is further argued in section 2.2. The system is achieved by fusing a number of RGB-D cameras rather than relying on a single or stereo pair of cameras. Multiple RGB-D cameras, however, are still not enough to provide the wide field of view of a freely moving human viewer. As such, we combine the dynamically-rendered view provided by the stationary RGB-D camera network

with a wide-area 3D environmental model of the background which is scanned off-line using a movable Kinect camera [2]. The environmental model not only provides a wider view, but also fills in missing background details due to occlusion. The depth information provided by the RGB-D cameras can be used to track the viewpoint and render the scene from different prospectives.

3. Missing and erroneous depth measurements are common problems with structured-light cameras, which can significantly degrade the performance of any subsequent vision processing. As such, we developed a novel stochastic framework that combines multiple RGB-D system noise models to robustly determine the depth layer label and uses depth layer in steering the completion process to produce well-defined depth edges [24]. The key to our model is the use of depth layers to account for the differences between foreground objects and background scene, the missing depth value phenomenon, and the correlation between color and depth channels. The depth layer labeling is solved as a maximum a-posteriori estimation problem, and a Markov Random Field attuned to the uncertainty in measurements is used to spatially smooth the labeling process. Guided by the layers, we can easily separate foreground and background, which can be used as the input for the mirror rendering. Our depth correction and completion algorithm outperforms other techniques in the literature [1][3].

4. To provide real-time performance, we avoid the computational intensive surface meshing process and base our design on a 3D point cloud which is faster in terms of both view acquisition and rendering. Our proposed algorithm admits

a parallel implementation across multiple machines in a scalable client-and-server architecture, where much of the 3D processing can be carried out on the client side. Specifically, each client machine can trace a light ray from each 3D scene point to the viewpoint, and render a partial mirror image which is then aggregated at the server.

5. To facilitate 3D construction and virtual view rendering from multiple color and depth cameras, I introduced a simple and robust framework for calibrating a network of cross-modal cameras with wide baselines. Rather than using the standard checkerboard, we use a sphere as a calibration object to identify the correspondences across different views. The procedure can be summarized as follows. First, we propose an effective sphere-fitting algorithm to identify the sphere centers in the RGB and depth images respectively. Second, the extrinsics are automatically obtained based on the corresponding sphere centers across different views. Two separate scenarios are considered in our framework: RGB and depth calibration and depth-and-depth calibration.

1.3 Dissertation Organization

This dissertation is organized as follows: After introducing the basic concept of Video Self Modeling and its potential applications described in Chapter 1.1, I review previous methods for solving different technical problems that are involved in our VSM systems in Chapter 2. In Chapter 3, I describe the details of one feedforward VSM content production system for patients suffering from voice disorders, mainly explor-

ing the solution for video resampling and lip synchronization. In Chapter 4, the implementation details of another VSM system, magic mirror, are provided. In particular, I introduce a novel depth denoising and completion algorithm that can be applied to any structure-light based depth sensors and potentially improve many subsequent vision processing. As an extension of mirror views for VSM, users can view self from any arbitrary 3D view. To provide such flexibility, multiple cameras are often used to capture the 3D data. To achieve this, a fundamental step is to multiple camera calibration. In chapter 5, I describe our calibration method for multiple cross-modal camera network to obtain globally aligned 3D point clouds. Finally, I conclude the dissertation and discuss future work in Chapter 6.

Chapter 2

Related work

In this related work section, I categorize it into multiple sub-sections according to different problems.

2.1 Lip Synchronization for SpeakToMe Device

Our goal of synthesizing new talking head video bears resemblance with the large body of work in facial animation using either real video footage [25] or avatars [26, 27]. The key difference is that we have exploited the requirements from the domain application in developing a fully automated real-time system. For example, we only need to re-sample the video sequence to achieve lip synchronization rather than a complete re-rendering of a new sequence as in [25]. Also it is unimportant for us to preserve emotion as in [26, 27]. On the other hand, there are more stringent audio requirements that we need to overcome in synthesizing a new speech track with a healthy voice that bears strong resemblance to the patient.

The problem of lip synchronization has been extensively studied in literature, which can be grouped into three different categories according to the data source: audio-based, video-based, and joint audiovisual processing. For pure audio-based techniques, Mermelstein's algorithm [28] is an influential rule-based syllable segmen-

tation approach. It locates syllable boundaries by computing the convex hull of the intensity envelope between 500Hz and 4kHz. A modification to Mermelstein's convex hull algorithm based on periodicity and normalized energy was developed in [29] for syllable nuclei detection. In [30], Howitt incorporated Neural Network into an energy-based vowel detector using Mermelstein's algorithm. In [31], a multi-pass automatic speech segmentation algorithm was proposed, which involves a broad segmentation by intensity dips in the filtered speech, followed by further adjustments for syllable nuclei. Despite the sophistication of audio segmentation techniques, consistent segmentation across speakers remains a very challenging problem.

As for video based techniques, a lip-motion recognition method using Hidden Markov Model (HMM) is proposed in [22]. Lip contour boundary was extracted based on the contrast intensity of the image. Similarly, in [32] and [33], a lip segmentation algorithm was developed by contour detection and model fitting. However, these methods require a priori knowledge about lip structure, which makes it difficult to achieve full automation. To avoid the training steps, a geometric deformable model was proposed in [34]. They used spatial fuzzy clustering to create a probability map about the lip image. Then, the lip position was obtained by maximizing the join probability of the lip region and non-lip region. Nevertheless, this method requires heavy computation due to the complex probability model, which makes it hard to achieve realtime or close-to-realtime performance. In [35], the authors provided an efficient implementation through field-programmable gate array (FPGA). In their system, a naive Bayes classifier was used to extract lips features. But this method may be error-prone if the lip color is close to that of the skin. In [36], an improved

active contour model was developed to extract the lip shapes by iteratively minimizing the proposed energy functions. Similar to our approach, it is based on the snake algorithm. But ours is simpler since for our problem, we do not need to track the lip motion over time.

Lip-motion and audio analysis are often combined to enhance the accuracy of speech recognition. There is a body of research termed *audio-visual speech recognition* (AVSR) which incorporates lip-motion as additional features in building the speech recognition engines [37, 38, 39, 40, 20, 21]. In this research, both visual and audio information are also combined in the production of the output video. The difference is that we use them for the alignment of the original speech signal and the replacement speech signal, rather than recognition of specific phones. As such, our focus is on the accuracy in identifying the same set of markers across speeches from different individuals. Once the alignment is identified, they are used to re-sample the video to establish lip-synchronization with the replacement speech.

2.2 Virtual Mirror

Virtual mirror systems are not new concept in computer vision. Several research groups have developed virtual mirror prototypes. Though they differ in some aspects, most of them implement simple appearance modification with a limited viewpoint [41, 42, 43]. Darrell et al. described a virtual mirror interface that reacted to people by applying different graphical effects on their faces [41]. Similarly, in [42], Kitanovski and Izquierdo proposed a virtual facial modification program by user-driven 3D-aware image warping. The same authors also presented a system with virtual mirror

experience in [42]. However, neither of them considered the viewpoint's influence on rendering a virtual mirror. The authors of [42] focused on face alteration of a monocular view of a webcam, while our system is designed to use multiple RGB-D cameras to capture the entire environment. Francois and Kang designed a hand-held mirror simulation device in [44]. Although they considered the viewpoint change during the mirror image transformation, their system used a simplified model by assuming the 3D world as a plane parallel to the mirror/imaging.

Virtual mirrors have also been used in some very specific applications. In [14], a real-time system was proposed for the visualization of customized sports shoes. Hilsmann et al. proposed a dress-fitting system where users can virtually dress themselves in different clothes and see how they look from the virtual mirror [45]. Similar systems have also been developed to generate virtual mirror images for fashion [15, 13, 46]. Since the target objects in these systems are already known, they usually have a pre-computed model from either an existing 3D model or collection of large training data. Therefore, the on-line computation can focus on rendering the correct texture on the generated image. In addition, the view point change is usually neglected in these systems due to its particular commercial intention. A recent paper by Straka et al. described a system that allowed the user to view himself or herself anywhere within a 360° field of view [47]. Multiple RGB cameras were mounted around the user to capture the data for 3D reconstruction. However, in their proposed system, only the person was rendered in the mirror without any background, which negated the essential characteristic of a mirror where the viewer could observe different scenes as the viewpoint changed.

One closely related subject of the virtual mirror is view-dependent texture mapping (VDTM) [48, 49, 50]. Though our system shares many similar designs as in VDTM, it differs from the traditional VDTM in the following aspects. First, the geometry of the object is usually known and represented as a pre-computed 3D mesh. In our system, we have a dynamic scene so the geometry needs to be estimated on-the-fly from the depth cameras. The incomplete description of the 3D world and the lack of object segmentation make generating a complete 3D mesh representation computationally difficult. Second, texture information in VDTM systems is usually recorded as planar image and image warping is a popular tool to map these textures into the 3D shapes. On the other hand, the network of image and depth camera pairs in our system is able to obtain texture information at different viewing angles for every 3D point collected in the system. This provides a more realistic 2D rendering of the scene. Finally, the viewpoint of our virtual mirror system is part of the 3D world captured and rendered by our system. Physical laws allow us to ignore certain aspects of rendering – for example, we cannot see the mirror when we are facing away from the mirror. No such restriction exists for VDTM.

2.3 Depth Image Denoising and Completion

A key component of our virtual mirror system is the use of commodity structured-light depth cameras such as the Microsoft Kinect cameras. Depth images obtained by such devices often have distorted and missing depth values. Most of the works in depth image enhancement can be grouped into two categories: super-resolution [51] [52] [53] [3] and image in-painting [54] [55]. A common theme is to rely on information obtained

from the companion color images to predict missing depth information. The use of color information for depth enhancement is based on the assumption that certain correlation exists between depth continuity and color image consistency [56]. While providing useful cue for interpolation, this assumption does not always hold as color edges and depth edges do not necessarily coincide with each other. In [51], Garro et al. presented an interpolation scheme for depth super-resolution. A high resolution RGB camera is used to guide the up-sampling process on the depth image. To interpolate the missing depth pixel, the scheme uses neighboring depth pixels mapped into the same color segment as the target pixel. This method relies strongly on the extrinsic alignment between the color and depth image. A similar segmentation based method can also be found in [52] where a non-local means filtering based approach is used to regularize depth maps and maintain fine detail and structure. In [54], Wang et al. proposed a stereoscopic in-painting algorithm to jointly complete missing texture and depth by using two pairs of RGB and depth cameras. Regions occluded by foreground are completed by minimizing an energy function. The system is cumbersome as an additional pair of color and depth cameras are needed. A recent depth fusion paper proposed by Zhang et al [57] aimed at capturing full frame depth by adaptively adjusting the contribution from photometric stereo and completed the depth in a edge-preserving manner. Three additional LEDs are used as assistance to fulfill the task.

Various probabilistic frameworks are often used in modeling depth measurements, fusing depth and color information, and predicting missing values. In [3], Diebel et al. demonstrated the use of Markov Random Field in the super-resolution of depth

data using high-resolution color data. However, their work provide little insight in modeling the sources of error in the depth sensor. Similarly, in [53], a low resolution depth image is iteratively refined through the use of a high resolution color image. Bilateral filter is applied to a cost function based on depth probabilities. A final high resolution image is produced by a winner-takes-all approach on the cost function. These approaches work well for the super-resolution problem where missing depth pixels are uniformly distributed. Depth images obtained by structured-light sensors often have large contiguous regions of missing depth measurements which cannot be handled by such approaches.

2.4 Cross-Modal Camera Calibration

Another important component and essential step of our virtual mirror system is the cross-modal camera calibration. As the depth or color sensors often have low resolution and limited field of view, the acquired 3D objects often suffer from nonintuitive, self-occluding hulls rather than full 3D shapes. Multiple cameras are often used to resolve these limitations. As such, it is critical to obtain the calibration parameters of the mounted cameras. Usually the calibration of a camera system consists of multiple connected cameras, which need to be registered globally by using the estimated extrinsic parameters to produce a unified data-stream. To achieve this task, a wide variety of different approaches have been proposed.

2.4.1 Color Camera Calibration

Traditional color camera calibration method is that the intrinsic and extrinsic parameters of a camera are calculated by mathematical transformations after the images are processed and calculated under a certain camera model, such as the pinhole camera model. Then an linear or nonlinear optimization of the calculation results is done using the maximum likelihood criterion. A reference target object is often used for the image capturing to identify correspondences across multiple views. It takes the advantage of a given calibration target that is known in shape and size. One of most widely used calibration technique is the Zhang's method [58], which uses a planar checkerboard pattern. The extracted checkerboard corners provide suitable constraints for the color images, followed by solving an optimization problem to estimate the camera parameters. Similar techniques can also be found in [59][60][61]. Other calibration objects have also been explored including texture plane [62] or circular features [63]. Among color camera calibration methods, three target types are often used: corners, checkerboard corners, and circular dots. The advantages and limitations between corner targets and circular targets can be found in [64]. There are also several target-free methods for camera calibration, in particular for view alignment for multiple cameras. In [65], the authors use sensor ego-motion and tracking moving texture plane to estimate the baseline between multiple cameras. Similarly, in [66], the topology of a large camera network is inferred by analyzing the motion of multiple moving objects. However, for these methods, they often have some limited assumptions. For example, the tracked moving object is arbitrarily assumed flat as

a plane, which is not always true and cannot be applied to a generic scenario.

2.4.2 Depth Camera Calibration

For depth cameras, different calibration methods have been developed based on the sensor's feature and imaging mechanism. For a time-of-flight (ToF) camera, it can simultaneously generate a depth image and an intensity image from the same viewpoint, which makes the calibration procedure simpler as the color discontinuity can be easily identified from the intensity image. However, for structure light depth sensors, texture patterns are not visible to the captured images. Special objects with depth variations need to be used to calibrate multiple depth sensors. For example, a planar calibration pattern with holes is used in [52] and [67]. However, there is significant noise around the holes in the depth images, which lead to erroneous correspondences across different depth views. A complicated multi-spline model is proposed in [62] to minimize the noises on the depth image. An improved version of [67] also incorporates a depth distortion and denoising step to improve the accuracy of calibration [68]. But each noise model has its own limitations and may not be able to fully remove noise in the small overlapping area between two depth sensors that are far apart. In contrast, our method does not rely on any depth noise model and is suitable to be used for any type of depth cameras. Targetless extrinsic calibration has also been studied in [69] [70]. They mount range sensors and color cameras on a moving platform and use the environment for calibration. However, these approaches require an initial estimate of the sensor pose.

2.4.3 Cross Modal Calibration

The task of estimating the geometric relationships in a multi-view, multi-modal camera network is called *cross-calibration*. A typical example is an RGB-D system, which provides multiple views from color and depth cameras. In the paper [71][72], the authors proposed a sensor-fusion framework by integrating depth and color sensors. It introduces an empirical calibration method which builds a look-up table with 4 dimensions, mapping observed intensity and 3D positions reported by the sensor to ground truth distance. However, the proposed methods require manually selecting correspondences points, and are based on a weak perspective camera model. In contrast, our method is automatic. Another group of cross-calibration methods [73][74][68] use plane constraints to detect the correspondences from the depth image without using features. However, these methods do not handle the noise very well as the correspondences are extracted directly from the depth image pixels. Furthermore, due to the limited field of view of the cameras and the resolution requirement on the captured target object, these methods do not work very well for wide baseline scenario. The same issues also exist in [46][75]. To the best of our knowledge, none of the approaches above have considered global optimization on multiple color and depth cameras simultaneously.

Chapter 3

Audiovisual Self Modeling

In this chapter, we describe a novel feedforward VSM content production system for patients suffering from voice disorders, which appear to be the most common communication disorder across the lifespan, are disabling and compromise quality of life.

3.1 VSM for Voice Disorder

Considering that 3-9% of the population has some type of voice disorder at any given point in time [76], is a significant medical problem. According to the National Institute on Deafness and Other Communication 7.5 million people in the United States have trouble using their voices. Voice disorders can have significant personal as well as societal impact. Voice disorders are a source of substantial functional loss for individuals, and a source of substantial economic cost to society in terms of higher health care costs.

Voice therapy is often the primary choice of treatment for voice disorder called vocal hyperfunction. Vocal hyperfunction is one type of voice disorders that is defined as the use of excessive muscle force and physical effort in the production of voice [77]. The traditional model of voice therapy typically involves participation in one or two

40-45 minute sessions per week over the course of eight weeks with the speech language pathologist to facilitate behavior change in production of voice. High dropout rates of 16% to 65% [78, 79, 80] coupled with reduced long-term success rates of 51% to 68% [81], suggest the need for development of new approaches for delivery of voice therapy to improve treatment success [82][83].

Access to voice therapy services for management of voice disorders in rural areas and developing countries is particularly lacking, due to difficulties in recruiting and retaining speech language pathologists and the expenses of time and travel for the required voice therapy program necessary to re-mediate the voice disorder. The use of VSM for voice therapy is a novel application where the pathologist can use the proposed system to create videos of a patient speaking with an improved voice. The patients can either take these videos after their initial visit to the clinic or access them through internet, and continue their behavioral modeling in their home. This new form of treatment has the potential of reducing the length of the treatment program and the number of therapy sessions, thereby reducing health disparities in rural populations.

3.2 System Overview

As vocal hyperfunction usually requires a prolonged period of speech therapy. Long-standing untreated voice disorders can detrimentally affect an individual psychosocially and academically and can be a source of substantial economic cost to society in terms of higher health care costs. The goal of our system is to reduce the amount of time on therapy using the principle of video self modeling. Our system records a

video of a patient at the clinic reciting a known script, and synthesizes a new video with a "healthier" voice for self-modeling. The purpose of this new video is to encourage patients to continue practising of the skills learned at the clinic in order to accelerate the behavioral changes. The proposed video self modeling approach can be used for voice therapy in individuals with vocal fold nodules, functional dysphonia. It can also be used in speech therapy for post-surgical management of individuals with vocal fold polyps and vocal fold cysts. However, before subjecting the approach of video self modeling to empirical testing with traditional voice therapy approaches, it is critical to test the robustness/accuracy of the image processing algorithm. Our system uses a data-driven approach in selecting a replacement speech best resembling that of the patient. A novel joint audio-visual algorithm is developed to synchronize the old speech with the replacement speech. The synchronization process produces a set of time markers which are then used to re-sample the video to achieve perfect lip synchronization. To minimize the amount of motion jitter introduced during the re-sampling process, we introduce a novel adaptive sampling strategy to preserve the motion energy of the original video. Extensive objective and subjective testing have been conducted to demonstrate the effectiveness of the proposed system.

3.3 Audiovisual Analysis

The goal of the audiovisual analysis is to identify a set of time markers that partition the speech signal at the phone boundaries. Two sets of markers are identified based on the variation of the loudness of the speech signal and the lip openness detected in the video signal. They are then combined to obtain a robust alignment between the

original and the replacement speech tracks using dynamic programming. The details are described in the following subsections.

3.3.1 Audio Analysis

Our audio segmentation module uses a derivative-based approach which consists of three steps: signal envelop calculation, differentiation, and minima identification. An example of this three-step process is shown in Figure 3.1. First, the signal envelope is computed by applying a low pass filter on the cepstrum of the smoothed input signal. The envelope is obtained by the following equation:

$$E = exp(F^{-1}(W(F(log(y))))) \tag{3.1}$$

where y is the input audio signal, and W is a low-pass window. F is the Fourier transformation. During the recording phase, we use sliding text to control the display speed for the speaker to read the script. This can roughly partition the speech signal into alternative continuous speech and silence periods. Starting with these rough partitions, we use a sliding window over the envelope to compute the short-term signal variance. A significant increase in the variance indicates the beginning of the speech period and a significant drop represents the end. Second, we compute the derivative of the envelope by convolving it with a Gaussian derivative filter. In the final step, the local minima of the envelope are identified based on the increasing zero-crossings of the derivative signal. Each local minimum is treated as the boundary between two phones.

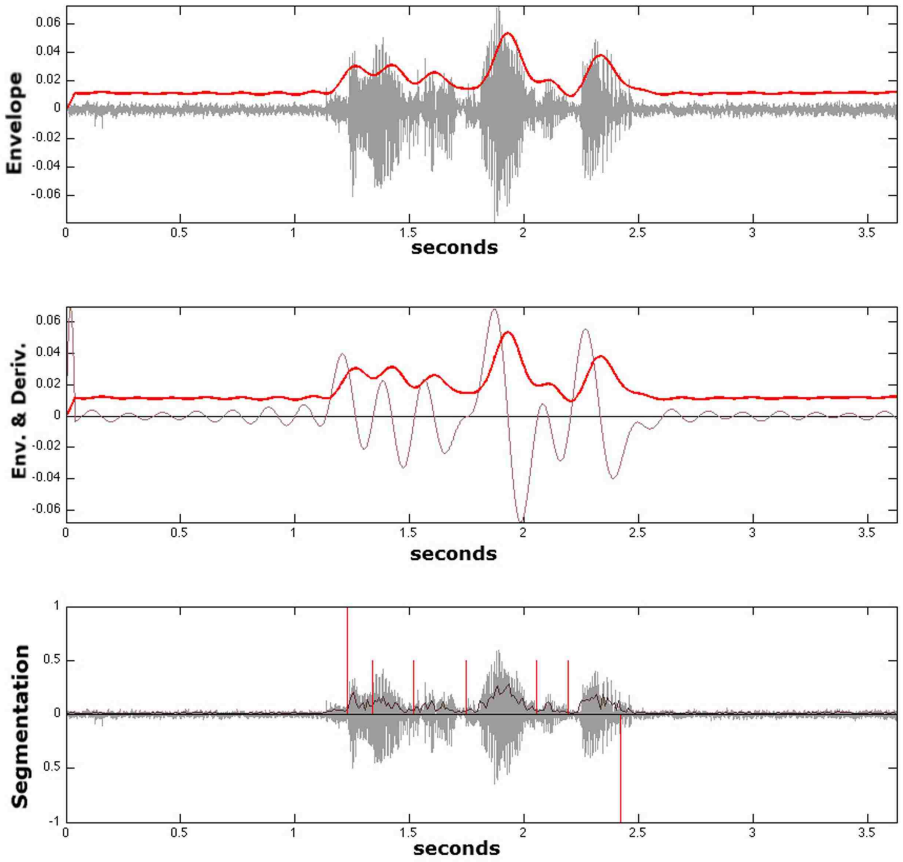


Figure 3.1: Three steps of phone segmentation: envelope calculation (top), differentiation of envelop (middle), minima identification (bottom).

The audio segmentation method does not necessarily recover the true partition of all phones. When the algorithm runs through multiple phones where there is no volume fall-off, it will interpret them as a single phone, resulting in time marker omissions. Such an error is not consistent across different speakers. Discrepancies in time markers between the source and target speech signals can significantly degrade the performance of the subsequent speech alignment. A misalignment due to an omission of a time marker in one signal can propagate to a much longer period before synchronization can be re-established at the end of a continuous speech period. To enhance the correct detection of all time markers, we turn to the video and analyze

the lip movement.

3.3.2 Video Analysis

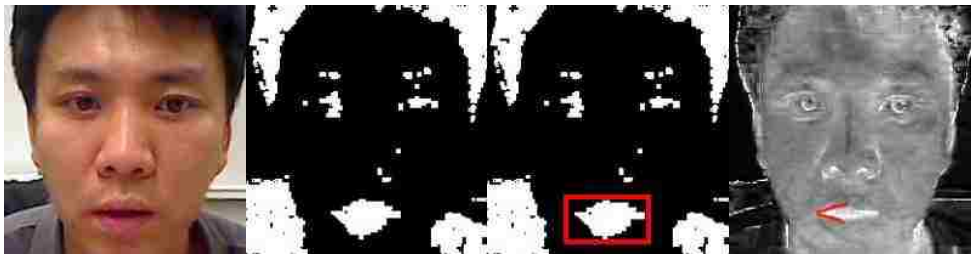
Lip state detection is employed to enhance the accuracy and robustness of phone segmentation. Unlike lip-reading techniques, it is not necessary for us to identify the exact lip shape. Instead, we notice a significant change in lip change from open to closed or vice versa coincide well with time markers of some phones. As such, we employ the following procedure to detect changes in the shape of the lip.

Face Detection

For each frame, the speaker's face is first detected using an Adaboost classifier on Haar-like features [84]. As this classifier is primarily for detecting frontal upright faces, the input image is rotated about the center by a range of small angles and the classifier is applied to each rotated image to ensure proper face detection.

Mouth Detection

In this step, we modify the approach described in [85] to detect the mouth region. The face region is first converted to the HSV space. An edge map is then obtained by applying the Sobel edge filter on the difference between the hue and the luminance channels. Connected component clustering is then applied to the edge map. The mouth blob is determined to be the largest blob straddling the vertical centerline of the face region. Sample results from these steps are illustrated in Figure 3.2(a)-(c).



(a) (b) (c) (d)

Figure 3.2: Processing results on the face image: (a) Original face (b) Sobel Filter on hue-luminance differnce (c) Mouth boundary by blob detection (d) Inside-lip contour

Lip Contour Tracking

After detecting the mouth region, the contours of the lips are extracted in this step. Figure 3.2(d) shows the saturation channel of a face image. As the lip is more saturated in color than the skin tone, we can take advantage of this observation to track the contour of the inside of the lip and determine if the mouth is open or closed. To track the contour of the inside lip, we use the active snake algorithm from [86]. A snake is simply a piece-wise linear contour that is computed based on the optimization of an appropriately chosen objective function. Two snakes, tracking the upper and lower lips, are initiated from the left corner of the mouth which is detected using the feature point detection from [87]. The extension of the snakes from their starting point is guided by the gradient vector field of the hue channel. Specifically, the end-points of the segments of a snake, $\{\mathbf{P_0}, \mathbf{P_1}, \mathbf{P_2}, \ldots, \mathbf{P_n}\} \in \Re^2$ with a fixed starting point $\mathbf{P_0}$, are recursively computed by maximizing the normalized line integral along each line segment:

$$\mathbf{P_i} \triangleq \arg\max_{\mathbf{P} \in S_i} \int_{\overline{\mathbf{P_{i-1}P}}} \frac{\nabla H(\mathbf{r}) \cdot d\mathbf{r}}{|\overline{\mathbf{P_{i-1}P}}|} \tag{3.2}$$

where $H(\cdot)$ is the hue image and $\overline{\mathbf{P_{i-1}P}}$ is the line segment between $\mathbf{P_{i-1}}$ and $\mathbf{P_i}$. S_i defines the search region for the endpoint $\mathbf{P_i}$. In our implementation, the integral in Equation (3.2) is approximated by the corresponding discrete sum, $\nabla H(\mathbf{r})$ by the Sobel edge detector over the hue channel, and $S_i \triangleq \{(x_0 + i\Delta_x, y) : \ y_{i-1} - \Delta_y^1 < y < y_{i-1} + \Delta_y^2)\}$ where $\mathbf{P_i} = (x_i, y_i)$. Δ_x controls the resolution along the x-direction and $\Delta_y^{1,2}$ controls the search range in the y-direction. $\Delta_y^1 < \Delta_y^2$ is used for the upper snake and $\Delta_y^1 > \Delta_y^2$ for the lower snake. These parameters are all empirically determined.

Lip States Classification

In our system, the lip shape is classified into two states: *open* and *closed*. The state is determined by the angle made by the upper and lower snakes. As such, the two snakes are only grown until they reach the centerline of the mouth. Two regression lines are then fit to the upper and lower snakes and the angle ϑ between them is measured. An example of the two regression lines are shown in Figure 3.2(d). A temporal median filter is then used to remove noise in the time series of ϑ. The *closed* state is assigned to time intervals where ϑ is close to zero. Time markers are recorded when the lip state changes from *open* to *closed* or vice versa. The measurement of lip state provides a visual cue for lip synchronization that cannot be provided by the audio segmentation step. However, not every phone involves closing of the lip and the number of lip-state changes is typically far fewer than the actual number of phones. To provide an accurate set of time markers, the time markers from this section must be intelligently combined with those from Section 3.3.1 to obtain the final answers.

3.3.3 Fusion of audio and video time markers for alignment

From Sections 3.3.1 and 3.3.2, we find that results from audio and video segmentations could both be used to perform temporal alignment. Yet neither one produces accurate enough markers for alignment. Audio segmentation may miss markers that separate closely spaced phones, a phenomenon that differs from speaker to speaker. Lip-state segmentation is consistent across speakers, but not every phone can be detected based on changes in lip states. Robust alignment is only possible if we combine both sets of time markers together.

The combination is performed after the time markers for both the original and the replacement speech track have been produced. We first normalize all the time markers by the corresponding duration of the speech tracks so that they are between 0 and 1. Denote the two sets of *lip-state time markers* as $M_1 = \{s_1, s_2, \ldots, s_m\}$ and $M_2 = \{t_1, t_2, \ldots, t_n\}$ with $m \leq n$. The alignment $t_{i(k)}$ in M_2 for each s_k in M_2 is obtained by minimizing the following objective function:

$$S \triangleq \min \sum_{k=1}^{\min(m,n)} |s_k - t_{i(k)}| \tag{3.3}$$

The optimal alignment with the constraint of a monotonic increasing $i(k)$ can be obtained by applying dynamic programming to minimize S. The video alignment produces a coarse but reliable alignment of the two speech tracks. For each pair of corresponding segments (s_{k-1}, s_k) and $(t_{i(k-1)}, t_{i(k)})$, we collect all the audio time markers within and apply exactly the same alignment procedure again to these time markers. This step provides the finer level of alignment of phones between successive video markers.

The fusion of video and audio alignment enhances the matching accuracy and is more robust to handle errors. For example, if there is an absence of audio time markers, it could decrease the overall accuracy of the time marker matching between speech tracks. However, with the fusion of the audio and video, the influence of the wrong or missing time marker can be reduced. As we first use the lip state to divide all the (audio) time marks into several subsets. Only time markers from the corresponding subset are considered for matching. So a wrong or missing audio time maker can only affect the matching accuracy of those time markers from the same subset. In this way, the errors are stopped from propagating down to the whole speech track.

3.4 Video Re-sampling and Synthesis

In this section, we describe the process of generating the replacement speech signal and re-sampling of the original video signal for lip synchronization based on the optimal alignment determined in Section 3.3.

3.4.1 Replacement Speech Generation

To generate the replacement speech track, we have tested two different approaches: the first one is to use a commercially available text-to-speech synthesizer from Cereproc [88] and the second one is to use a speech corpus of healthy voices. The motivation of using the second approach is due to the questionable quality of the synthesized speech from the text-to-speech engine. While the text-to-speech engine offers great flexibility in generating arbitrary scripts and produces reasonably sounding speech,

it still lacks the naturalness in real human speech. Since the scripts used in a typical therapy session are usually fixed, we collect speech clips from a diverse set of individuals with healthy voices reciting the same script used in the therapy session. Then, we identify among all the speakers in the corpus the one who sounds most similar to the patient's voice. To compute speaker similarity, we use a state-of-the-art text-independent speaker identification system called ALIZE [89]. ALIZE represents individual speaker models using Gaussian Mixture Model (GMM) over linear frequency cepstral coefficient features. We use the data collected from a generic speech corpus to construct a 2048 component world GMM model, which is then adapted to individual speaker models in our voice corpus. In the actual deployment, we use the patient's voice as input and find the speaker that produces the maximum likelihood ratio between the respective GMM models among all speakers in the corpus. To make the selected or generated speech signal sound even closer to the patient, we have further experimented with a non-parallel voice conversion process described in [90]. This module modifies the speech based on the vocal tract model constructed using the patients speech. The voice conversion algorithm warps the source speakers spectrum to the target spectrum in time domain using vocal tract model. During the training phase, the warping parameter and the fundamental frequency ratio are computed. During the conversion, the synthetic speech from the text-to-speech engine or the healthy voice speech selected from the corpus is warped using these parameters towards the target spectrum.

3.4.2 Adaptive Video Re-sampling

The objective of the video processing unit is to re-sample the input video track so that it will be lip-synchronized with the replacement speech track. Due to the differences in the word durations between the original and replacement voice tracks, adaptive re-sampling must be applied to achieve lip synchronization. During the segmentation phase in Section 3.3, time markers have already been identified for all segments containing phones. The result is a one-to-one mapping between the segments from the original and from the replacement speech tracks. The goal of the re-sampling scheme will be to re-sample each segment of the original video track to match the length of the corresponding segment in the replacement speech track.

The most straightforward approach is to apply uniform re-sampling for each segment independently. Based on our preliminary study, we notice that while the differences in the duration between corresponding word segments from two speech tracks are relatively small, there are large variations among the corresponding silence segments in between. Significant up-sampling or down-sampling creates unevenness in motion or motion jitter, making the resulting video unnatural. While we maintain a uniform re-sampling for all the word segments, we adopt a different approach for the silence segments to preserve the original motion as much as possible. In the case of down-sampling, we would keep more frames at the portions with higher motion to better preserve the movement. In the case of up-sampling, we would add frames or expand the static portions so that we will not slow down or distort the significant object movements. This procedure is illustrated in Figure 3.3.

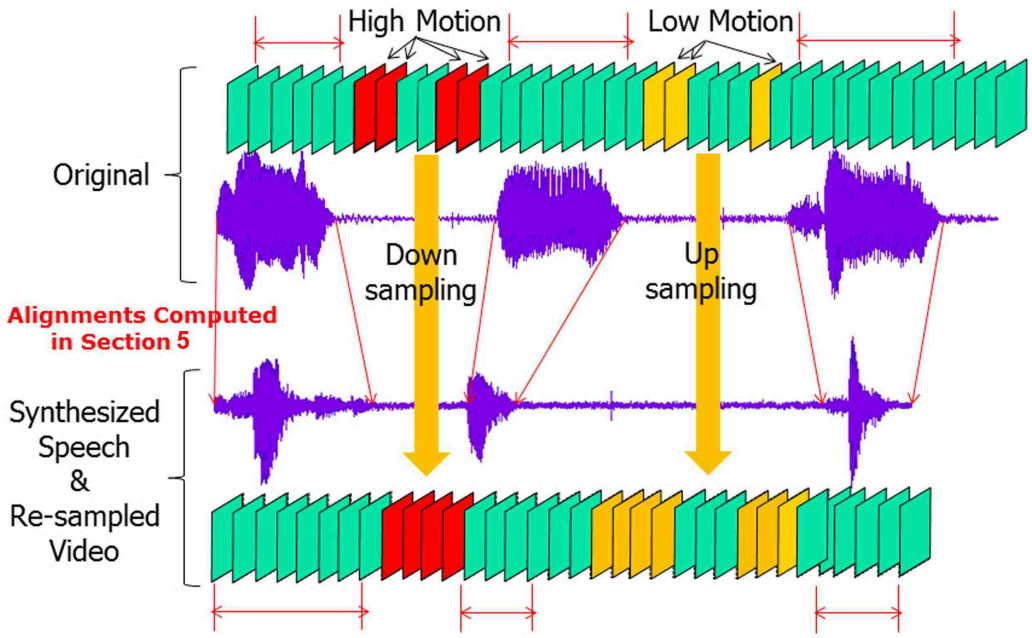


Figure 3.3: Up-sampling and Down-sampling

This results in the proposed adaptive re-sampling algorithm for the silence segments shown in Algorithm 1. The re-sampling is treated as a process of creating the set of output frames with a target number of frames M from the input set of N original frames. Step 1 preserves all the original frames in the case of up-sampling. Motion energy is computed between successive frames in step 2. In step 3 and 4, we identify the pairs of original frames that have the highest or the lowest motion energy, depending on whether the goal is to up-sample or down-sample the sequence. For up-sampling, the new frame will be added to the lowest motion energy to stretch the static region. For down-sampling, the new frame is added to give priority to the portion with the highest energy. The routine INTERPOLATE is used to interpolate a video frame between two different frames. The simplest technique is to use bilinear interpolation which can lead to motion blurriness and ghosting. As such,

we have also tested bidirectional interpolation based on dense optical flow vectors. The forward and backward optical flow vectors are estimated based on the pyramidal Lucas-Kanade algorithm as implemented in the OpenCV library. The flow vectors are then smoothed by a simple median filter. The temporally-scaled forward and backward vectors are then used in identifying pixels on the input frames that can be combined in creating the intermediate frames. For pixels in the intermediate frame that are not mapped by neither a forward or backward vectors, straightforward bilinear interpolation is applied. In step 6 of Algorithm 1, we deliberately remove portions of the sequence where we have already added new frames - this step prevents clustering of added frames in a small number of low/high motion areas. The parameter Δ is empirically determined to be two frames.

Protocol 1 Silence Segment Re-sampling

Input Input frames: $I = \{I_1, I_1, \ldots, I_N\}$
Output Output frames: $J = \{J_1, J_1, \ldots, J_M\}$

1. For up-sampling (i.e. $M \geq N$), insert all frames of I into J.
2. Compute mean-square error between consecutive input frames: $e_i = MSE(I_i, I_{i+1})$ with $I_i, I_{i+1} \in I$.
3. For up-sampling, select the pair (I_i, I_{i+1}) from I with the minimum e_i.
4. For down-sampling ($i.e. M \leq N$), select the pair (I_i, I_{i+1}) from I with the maximum e_i.
5. Create new frame $J = INTERPOLATE(I_i, I_{i+1})$ and add J into J with the time order preserved.
6. Remove $I_{i-\Delta+1}, I_{i-\Delta+2}, \ldots, I_{i+\Delta}$ from I.
7. Repeat previous step 3-6 until $|J| = M$.

3.5 Experiments

To test the performance of our system, we capture video clips from a total of 31 participants. The participants are native English speakers of ages between 18-40. During the recording stage, each participant read the same script commonly used in speech therapy[1], which consists of a series of isolated words and short sentences. The experiment is conducted in a quiet room that only has a researcher and the testing subject involved. To accurately track the speakers face, there is a certain limit in the range of the distance (0.4m to 0.8m) that the subject sits from the camera. The size of the extracted face region ranges from 160×240 to 210×325 in pixels.

We use a Logitech QuickCam Pro 9000 to capture the video at a resolution of 640×480, and an EMU 0404/Electro Voice PL5 combination for the audio recording. The proposed algorithm is implemented in C++ with the OpenCV library that runs on a computer with the hardware setting: Intel Core^{TM} i7-2820QM CPU at 2.3 GHz and 8.0GB of RAM. The video clips are on average 2 minutes and 8 seconds long, and are captured at 30fps (video) and 22.5kHz (audio). For the audio segmentation, it takes about 43 seconds to identify the boundaries between phones. The time cost for the audio-visual matching is about 3 seconds. The frame rate of lip state detection is 17 fps. It takes about 0.244 seconds to synthesize a new video frame from the original sequence.

Out of the 31 participants, 25 of whom are considered to have healthy voice. The speech recordings and the text-to-speech recordings form the candidate dataset for

[1]The script can be found in http://vis.uky.edu/nsf-autism/speaktome

speech replacement. The remaining six participants are voice experts who can imitate the strained voice commonly present in patients with vocal hyerfunction. The speech tracks of their video will be replaced by one of the tracks from the candidate set, followed by voice conversion process. The optimal alignment between the original and the replacement tracks is identified and adaptive video re-sampling is applied to achieve lip-synchronization. In the sequel, we measure the performance of individual components of our proposed system.

Replacement Speech Generation

We first consider the effect of different audio processing steps in producing a speech sample that best resembles the healthy voice of the subject. We use the following log-likelihood ratio measured by ALIZE in gauging the similarity between the candidate speech S and the original speech L:

$$LLR(S|L) = \log\left(\frac{l(S|L)}{l(S|W)}\right) \tag{3.4}$$

where $l(S|L)$ is the likelihood of S based on the adapted GMM model generated using L as the training data and $l(S|W)$ is the likelihood of S based on the world GMM model. The world model is trained based on the entire TIMIT dataset [91]. This dataset contains 6300 utterances from 630 speakers with both male and female from 8 major dialect regions of United States. Table 3.1 shows the log-likelihood ratios of different replacement speech candidates. It is unsurprising to see that the mimicked voice is the one closest to the healthy voice. Among the other candidates, the best human voice is ranked top followed by the text-to-speech version. On the

Table 3.1: Similarity to Healthy Voice

S	LLR
Mimicked Voice	9.03e-2
Best-human	2.26e-2
Best-human + Voice Conversion	0.47e-2
Text-to-speech	1.39e-2
Text-to-speech + Voice Conversion	-0.63e-2

other hand, the application of voice conversion seems to have a detrimental effect. One possible explanation is the proper selection of the warping parameter α and the fundamental frequency ratio r. The parameters computed directly by the software produce voices that are non-human like, most likely due to the non-natural hoarseness in the mimicked voice. We have tuned the parameters in such a way that the voice is more human but it adversely affects the overall similarity to the target voice.

Time Marker Alignment

Once the replacement speech has been identified, the alignment process is performed between the original and the replacement speech. Time markers from the automatic audiovisual analysis are compared against the ground-truth alignment which is manually obtained by listening to the original and replacement speech tracks. To arrive at an appropriate measurement of alignment, consider a pair of speech tracks A and B. Let the ground-truth time markers for A and B be $\{\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_n\}$ and $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$. Note that the number of markers in A and B are identical and the phone (or silence) in A during $[\mathbf{t_{i-1}}, \mathbf{t_i})$ is the same as those in B during $[\mathbf{s_{i-1}}, \mathbf{s_i})$.

After our proposed alignment process, we obtain the correspondences between the automatically-determined time markers from A and B. Denote the correspon-

dences as $\mathbf{a}_i \leftrightarrow \mathbf{b}_i$ for $i = 1, 2, \ldots, m$ where $\mathbf{a}_i$ and $\mathbf{b}_i$ are time markers from A and B. To compare with the ground-truth, we first identify the ground-truth interval $[\mathbf{t}_{j(i)}, \mathbf{t}_{j(i)+1})$ in A that contains $\mathbf{a}_i$ for each i. If the ground-truth is correct, $\mathbf{a}_i$ from A should roughly correspond to $\mathbf{a}'_i$ in B based on linear interpolation:

$$\mathbf{a}'_i = \mathbf{s}_{j(i)} + \frac{(\mathbf{a}_i - \mathbf{t}_{j(i)})(\mathbf{s}_{j(i)+1} - \mathbf{s}_{j(i)})}{\mathbf{t}_{j(i)+1} - \mathbf{t}_{j(i)}}$$

Thus, the absolute error of the correspondence $\mathbf{a}_i \leftrightarrow \mathbf{b}_i$ is $|\mathbf{b}_i - \mathbf{a}'_i|$. Each ground-truth interval may contain zero or more such markers and we want to first measure the matching error over the entire interval. Consider the set of all automatically-determined time markers $M_A(j(i))$ in $[\mathbf{t}_{j(i)}, \mathbf{t}_{j(i)+1})$ of A. We can compute the average relative error:

$$E_A(j(i)) = \begin{cases} 1 \text{ if } M_A(j(i)) \text{ is empty} \\ \frac{1}{|M_A(j(i))|} \sum_{\mathbf{a} \in M_A(j(i))} \frac{|\mathbf{b}-\mathbf{a}'|}{|\mathbf{t}_{j(i)+1}-\mathbf{t}_{j(i)}|} \text{ otherwise} \end{cases} \tag{3.5}$$

where $|M_A(j(i))|$ denotes the number of markers in of $M_A(j(i))$. We use relative error so that we do not bias against long intervals. Note that for significantly skewed alignment, i.e. $\mathbf{b}_i \notin [\mathbf{t}_{j(i)}, \mathbf{t}_{j(i)+1})$, the relative error can be bigger than 1.

This error measurement is not symmetric for the case when $\mathbf{b}_i \notin [\mathbf{t}_{j(i)}, \mathbf{t}_{j(i)+1})$. To derive a symmetric metric, we reverse the role of A and B: if $\mathbf{b}_i$ is in interval $[\mathbf{s}_{k(i)}, \mathbf{s}_{k(i)+1})$ in B, the time marker in A that corresponds to $\mathbf{b}_i$ is

$$\mathbf{b}'_i = \mathbf{t}_{k(i)} + \frac{(\mathbf{b}_i - \mathbf{s}_{k(i)})(\mathbf{t}_{k(i)+1} - \mathbf{t}_{k(i)})}{\mathbf{s}_{k(i)+1} - \mathbf{s}_{k(i)}}$$

The absolute error in this direction would be $|\mathbf{a}_i - \mathbf{b}'_i|$ and the corresponding average relative error within $[\mathbf{s}_{k(i)}, \mathbf{s}_{k(i)+1})$ is analogously defined:

$$E_B(k(i)) = \begin{cases} 1 \text{ if } M_B(k(i)) \text{ is not empty} \\ \frac{1}{|M_B(k(i))|} \sum_{\mathbf{b} \in M_B(k(i))} \frac{|\mathbf{a}-\mathbf{b}'|}{|\mathbf{s}_{k(i)+1}-\mathbf{s}_{k(i)}|} \text{ otherwise} \end{cases} \tag{3.6}$$

Finally, a symmetric average relative error over all the ground-truth intervals can be computed as follows:

$$E = \frac{1}{2n} \sum_{i=1}^{n} (E_A(i) + E_B(i)) \tag{3.7}$$

Using the best human sample as the replacement speech, we measure the average relative error of the alignment for the six strained speech sample. The results are tabulated in Table 3.2. For comparison, the performance of using audio only and video only for alignment are also listed. In the table, the first column $< S_*, H_* >$

Table 3.2: Results of forced choice tests

Video Pair Number	Error (video)	Error (audio)	Error (combined)
$< S_1, H_4 >$	0.7350	0.1854	0.0760
$< S_2, H_{10} >$	0.7413	0.2066	0.0859
$< S_3, H_{10} >$	0.7472	0.2040	0.0692
$< S_4, H_{23} >$	0.7378	0.2125	0.0751
$< S_5, H_{24} >$	0.7408	0.1721	0.0704
$< S_6, H_1 >$	0.7458	0.1714	0.0767

are pairs of strained and healthy voices, in which each healthy voice is identified by ALIZE measurement as one bearing maximum resemblance to the corresponding strained voice. The third column shows the average relative error for alignment using lip-state changes only. The error is very high but consistent across different speakers. The fourth column shows the average relative error for alignment using audio segmentation. The results are better than those using lip-state only but higher variation is observed across different speakers. The last column shows the average relative error of combining both together using our proposed scheme. There is a dramatic reduction in the relative error and the results are accurate across different speakers.

While Table 3.2 shows the average relative error, more information can be obtained by showing the histogram of the relative error $(E_A(i) + E_B(i))/2$ across all ground-truth intervals for all pairs of matched speech tracks. The three histograms corresponding to video only, audio only, and the combined scheme are shown in Figure 3.4a, 3.4b, and 3.4c respectively. The bimodal nature of Figure 3.4a indicates that many phones cannot be detected with the changes of lip-states. The high variance in the relative error shown in Figure 3.4b is characteristics of audio segmentation as the accuracy is highly speaker-dependent. The sharp concentration on low average error shown in Figure 3.4c shows the superior performance of our proposed audiovisual scheme over the other two.

Video Interpolation

Figure 3.5a and 3.5c show two sample frame using bilinear interpolation, while Figure 3.5b and 3.5d the corresponding frame using optical flow interpolation. As expected, optical-flow interpolation produces a much sharper image, especially around high-motion areas such as eyelids and mouth.

Adaptive Re-sampling

We also study the effect of our adaptive re-sampling of silence segments. In Figure 3.6a, we first plot the MSE measurements between successive frames for the original sequence. While keeping all the "word" segments intact, we reduce all the silence segments into one quarter of their original length. Two methods are tested: uniform re-sampling and our proposed adaptive re-sampling. MSE between consecutive frames

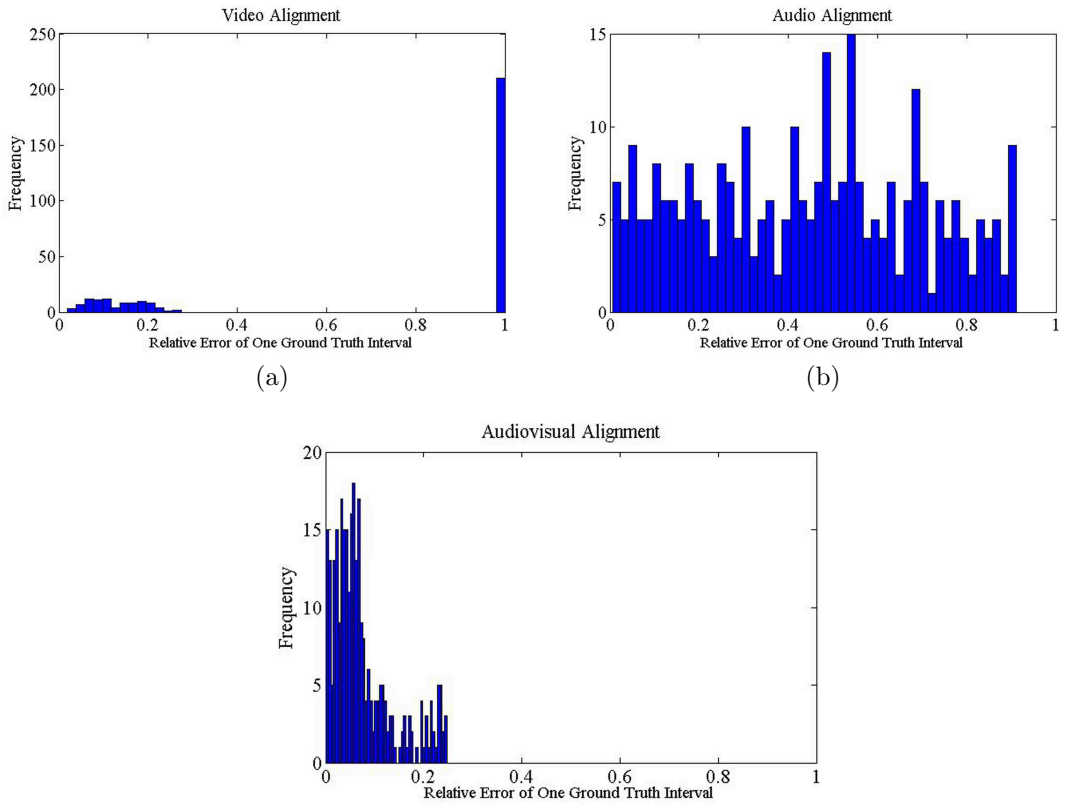
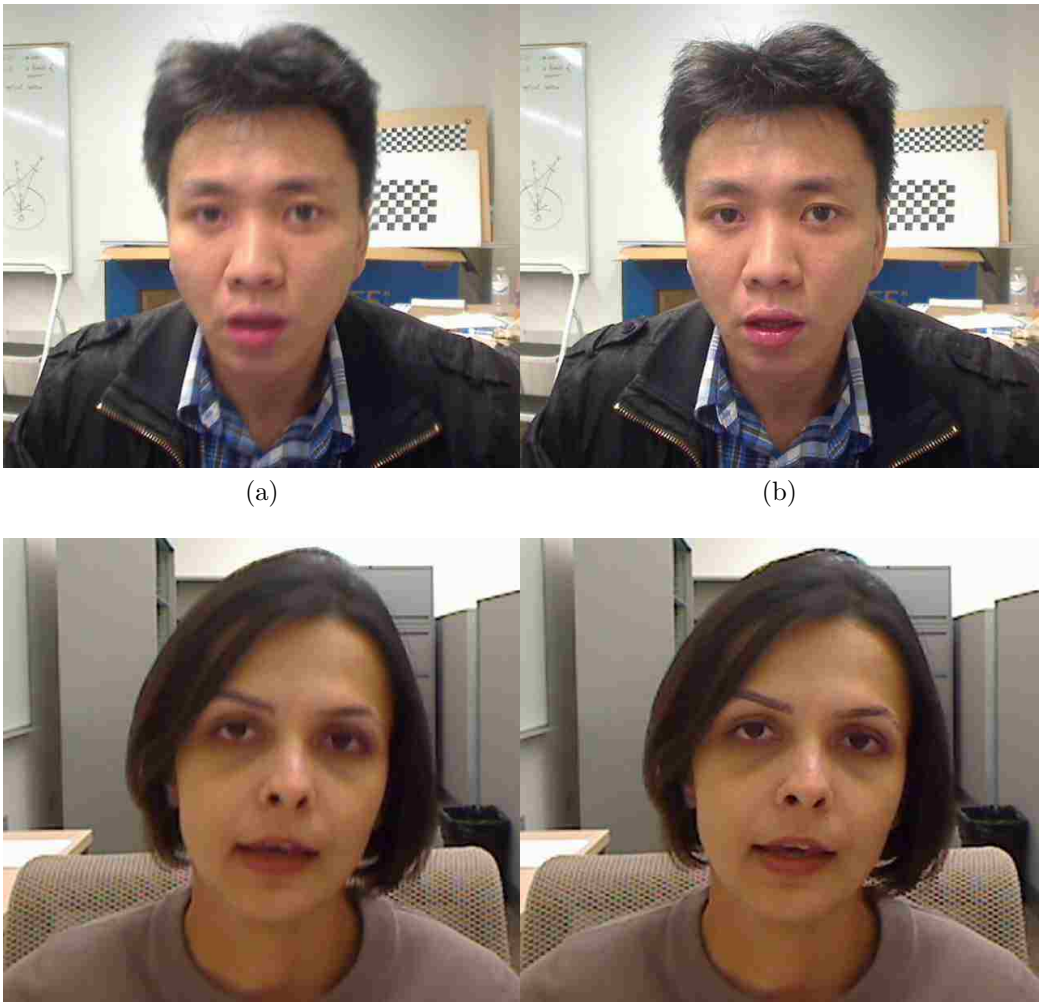


Figure 3.4: Histograms of relative error using (a) video only, (b) audio only, and (c) proposed audiovisual approach

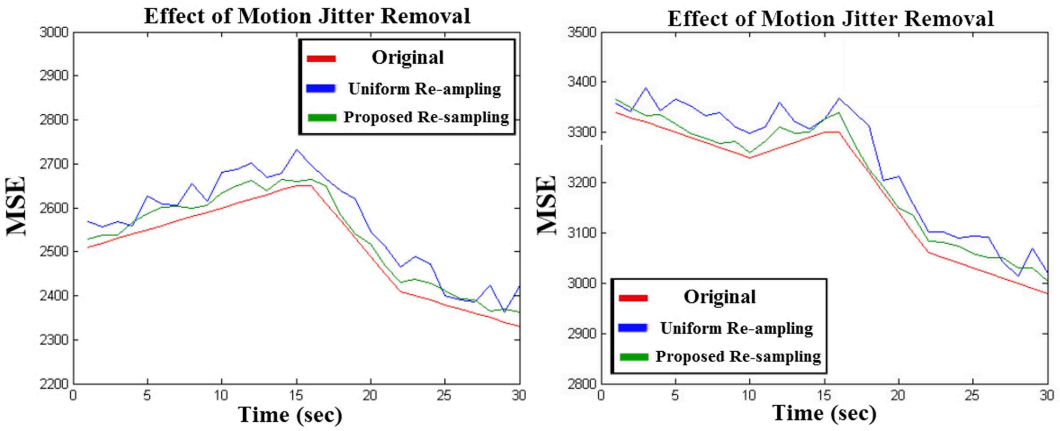
are then measured and the curves re-sampled to be the same time scale as the original curve. As shown in Figure 3.6a, our proposed approach provides a curve that can better preserve the original temporal energy than the uniform re-sampling approach. Figure 3.6b shows a similar trend when we up-sample all silence segments by a factor of four. Figure 3.7 demonstrates the up-sampling case. To synchronize the lip motion with the replacing audio, the original video (on the first and third rows) is prolonged by generating multiple intermediate frames.



(a) (b)

(c) caption (d) caption

Figure 3.5: Interpolation methods comparison: (a),(c) Bilinear Interpolation; (b),(d) Optical Flow Interpolation



(a) (b)

Figure 3.6: MSE curves by Uniform Re-sampling and Adaptive Re-sampling: (a) down-sampling (b) curve-sampling

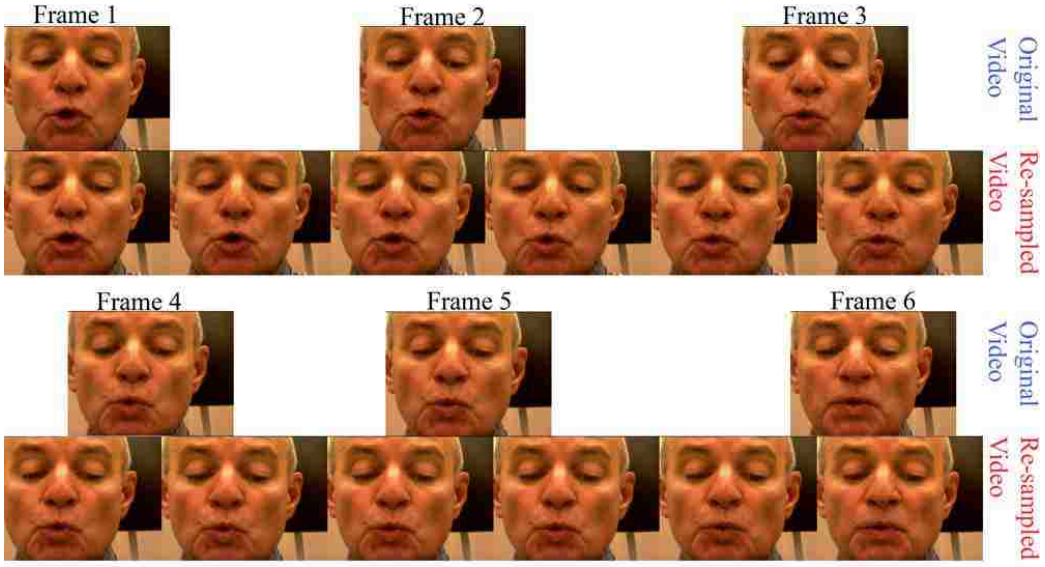


Figure 3.7: Video Sequence Re-sampling Example: the original video (row 1, 3) is prolonged in the synthesized video (row 2, 4).

Chapter 4

Virtual Mirror Self Modeling

In this chapter, we describe another novel feedforward VSM content production system - virtual mirror that facilitates the creation of synthetic visual contents for self-modeling, and combine the synthetic contents with 3D sensing to render a mirror-like feedback on novel display devices. Beyond the obvious applications of virtual mirror in fashion and cosmetics, we hypothesize that our proposed systems would be particularly beneficial to children with autism spectrum disorder (ASD). In contrast to their typical lack of interest in social interactions, many children with ASD appear to be highly interested in their own image in mirrors [18]. Additionally, children with ASD appear to respond well to others imitating their actions [92]. The virtual mirror will allow us to take advantage of both of these factors to enhance the understanding of self/other in children with ASD, resulting in superior social functioning.

Despite many advantages of a virtual mirror, there are a number of serious technical challenges in simulating a mirror. The naive 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 image depends on the position of the viewer. Thus, the main challenge of simulating the mirror is to render different content on the display depending on the viewers perspective. In or-

der to simulate a large mirror surface that can cope with the wide displacement of a viewer, a camera display system must be able to capture the 3-D details of the world, track the moving viewpoint, render the new view based on the position of a virtual mirror, and possibly add new visual content that is compatible with the scene geometry. Furthermore, it must be able to accomplish all these tasks in real-time and with extremely high fidelity, otherwise the virtual mirror system loses the instant visual feedback required to provide the realism of a mirror.

4.1 Overview

We propose a camera-display system that senses the 3-D world via a network of RGB-depth (RGB-D) sensors. Our system uses the depth information to track the viewpoint, and then renders the dynamic scene by tracing the light ray from each scene point to the point of reflection on the virtual mirror, and finally determines the proper position and color values on the display surface at which the virtual reflected ray passes through before reaching the viewpoint. Compared with other RGB-D rendering systems, our proposed system has the following technical contributions:

1. Like many other contemporary RGB-D systems, ours is also based on the low-cost Microsoft Kinect sensors which suffer from missing and erroneous depth measurements around object boundaries and other areas with poor IR reflection or strong interference [93]. We propose a novel graphical-model based on depth denoising and completion algorithm which is steered by a foreground/background separation scheme using color, depth, background model-

ing, depth noise modeling, and spatial constraints.

2. Different from previous work, our system provides a realistic mirror visual effect by incorporating three key features: viewpoint dependent content, wide field of view, and 3D based rendering. Existing work are limited in either one or some of these features, which is further argued in section 2.2. The system is achieved by fusing a number of RGB-D cameras rather than relying on a single or stereo pair of cameras. Multiple RGB-D cameras, however, are still not enough to provide the wide field of view of a freely moving human viewer. As such, we combine the dynamically-rendered view provided by the stationary RGB-D camera network with a wide-area 3D environmental model of the background which is scanned off-line using a movable Kinect camera [2]. The environmental model not only provides a wider view, but also fills in missing background details due to occlusion.

3. To provide real-time performance, we avoid the computational intensive surface meshing process and base our design on a 3D point cloud which is faster in terms of both view acquisition and rendering. Our proposed algorithm admits a parallel implementation across multiple machines in a scalable client-and-server architecture, where much of the 3D processing can be carried out on the client side. Specifically, each client machine can trace a light ray from each 3D scene point to the viewpoint, and render a partial mirror image which is then aggregated at the server.

4.2 Virtual Mirror Modeling and Simulation

As shown in Figure 4.1(a), the physical model of a mirror is very straightforward. The mirror image is based on reflecting a light ray from the scene to our eye, which can be modeled as a pinhole camera. Three components are necessary to reproduce this process in a virtual system:

1. Color and 3D Structure of the environment.

2. 3D coordinate of the viewpoint, or more precisely, the pupil of the eye. We limit our discussions to a single eye. Extension to a full stereoscopic system should be relatively straightforward by duplicating the rendering process for each eye and displaying the results on a stereoscopic display.

3. 3D location and pose of the mirror.

An important observation is that the mirror image is determined only by *the location of the viewpoint but not the pose of the viewer*. When the eye ball rotates from V_1 to V_2 around the same viewpoint in Figure 4.1(a), the scene point A should appear at the same spot A' on the mirror regardless of the viewing pose V_1 and V_2. As the pose of the eye ball is irrelevant to the mirror image, we only need to track the position but not the pose of the viewer.

To simulate the mirror experience, we need a camera-display system that can capture the 3D world, the viewer's position and then render the mirror image on a display surface. Notice that the virtual mirror does not need to coincide with the display. The general relationship between the mirror and the display is illustrated

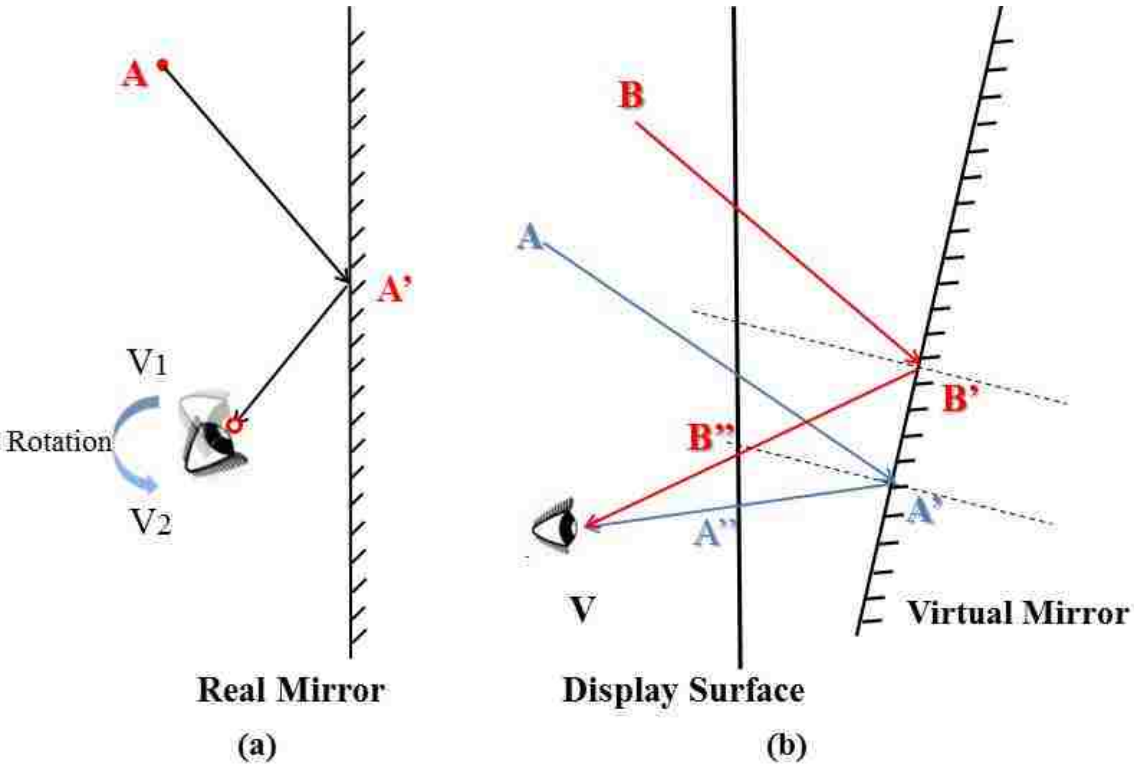


Figure 4.1: Virtual Mirror Modeling and Considerations

in Figure 4.1(b): a light ray from scene point A impinging on the mirror surface at point A' gets reflected along the ray $\overrightarrow{A'V}$ towards the viewpoint V. The visual effect of the virtual mirror is presented by rendering the point A" on the display, which is the intersection between $\overrightarrow{A'V}$ and the display plane. In a similar fashion, B'' is rendered as the mirror reflection of another scene point B with respect to the same virtual mirror.

To capture the 3D world, we have chosen to use the popular Microsoft Kinect RGB-D sensor. Kinect is a structured-light stereo RGB-D system consisting of a projector and two camera sensors. Special light patterns, typically in the infra-red (IR) spectrum, are emitted by the projector and captured by the IR CMOS camera sensor. Depth information of the sensing environment can then be inferred based on

how the light patterns are distorted in the IR images [94]. The RGB camera captures the color information and can be aligned with the IR camera by estimating the extrinsic parameters between them. While the depth information is independent of the viewer's position, color information does depend on the viewpoint unless the scene surface is Lambertian. Even though the Lambertian assumption is not realistic, more accurate sampling of the plenoptic function requires a significantly denser camera array which is not considered in our work.

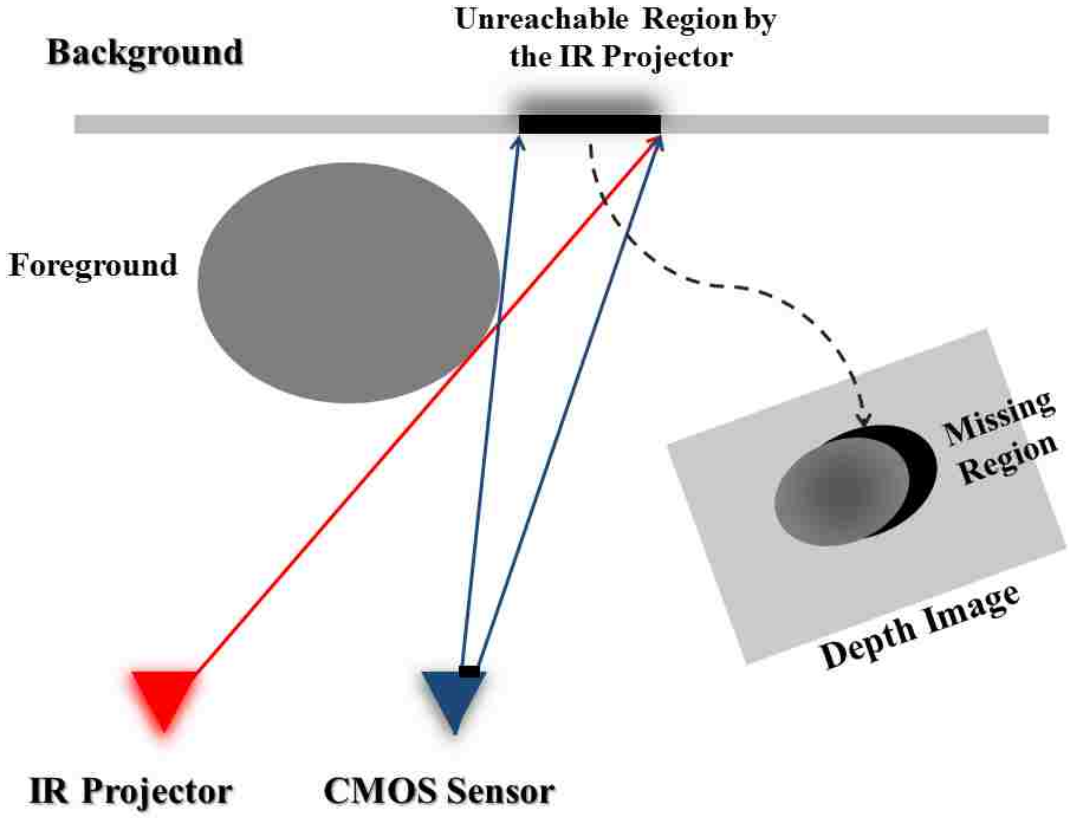


Figure 4.2: Missing depth measurements due to sensor-project disparity

It is well known that depth measurements from Kinect are noisy [93]. Missing and erroneous depth values can be caused by absorption, poor reflection or even shadow reflection of the light patterns. Objects with darker colors, specular surfaces,

or fine-grained surfaces like human hair are prime candidates for poor depth measurements [95]. Surface orientation also plays a role in depth measurements – as the surface normal deviates from the principal axis of the IR camera, the accuracy of the depth measurement declines and becomes unreliable near depth discontinuities [2]. Random noise in depth images may also be a result of inadequate calibration, sensor noise, and round-off error during normalization [93]. If multiple Kinects are used and their scanning ranges overlap, interference of the IR patterns may occur which can also lead to missing depth measurements [96]. For our mirror application in which the viewer is fairly close to the sensors, the primary source of noise is due to the disparity between the IR projector and the IR camera sensor. As illustrated in Figure 4.2, certain background regions visible to the IR camera are occluded by foreground objects and receive no structured light patterns. As a result, the depth values in those regions cannot be measured. Generic image denoising and complete algorithms fail to take into account the unique problems of structured-light RGB-D systems. Here we describe a novel stochastic framework that separates the depth image into foreground and background, and combines multiple RGB-D system noise models to robustly determine the depth layer label for the interpolation and completion of each erroneous or missing depth pixel. Details of this algorithm can be found in Section 4.3.

Consider the typical scenario where we want to use the entire display surface of size $w \times h$ as a mirror. It is important to determine the dimension of the moving space for the viewer so that accurate mirror rendering can be achieved. The top view of this scenario is shown in Figure 4.3. The side view is similar though the sensors can only be placed on the top and on the bottom of the display. With head and eye

movement, the human viewer has a very large field of view of 180°. On the other hand, Kinect has a horizontal field of view of $\theta = 57°$ and a vertical field of view of 43°. The Kinect also has a practical range limit from $d_{min} = 1.2$ m to $d_{max} = 6$ m. Depth cannot be reliably measured if the viewer is closer than d_{min} or farther than d_{max} away from the Kinect. To cover a large w, multiple Kinects need to be used and the resulting merged trapezoidal range field is shaded in gray in Figure 4.3. The characteristic trapezoidal shape implies that, depending on the spacing s between adjacent Kinects, there are pockets of areas beyond d_{min} that do not have depth measurements. As such, the viewer needs to be at least $d_{min} + d' = \frac{s}{2\tan(\theta/2)}$ away to prevent any restrictions in the horizontal movement. The rectangular area with diagonal lines represents the region in which the viewer can move freely and still be able to see his/her own mirror image. The background, however, requires a much bigger coverage due to the 180° field of view of the viewer as illustrated by the two pairs of incident and reflect rays between the virtual mirror and the viewer. The vast background area is scanned off-line and the scanning procedure is described in Section 4.4.

To provide an efficient computational platform, we adopt a client-server distributed system to handle multiple RGB-D cameras. Each client is responsible for one RGB-D camera. It first computes the 3D point cloud in the world coordinate system and an initial estimate of the viewpoint. Based on the viewpoint from the previous instance, the client will render the mirror image, possibly incomplete, based on its own point cloud data. As such, the 3D point cloud processing, the viewpoint estimate and the mirror image rendering are all done at the client level. The mirror

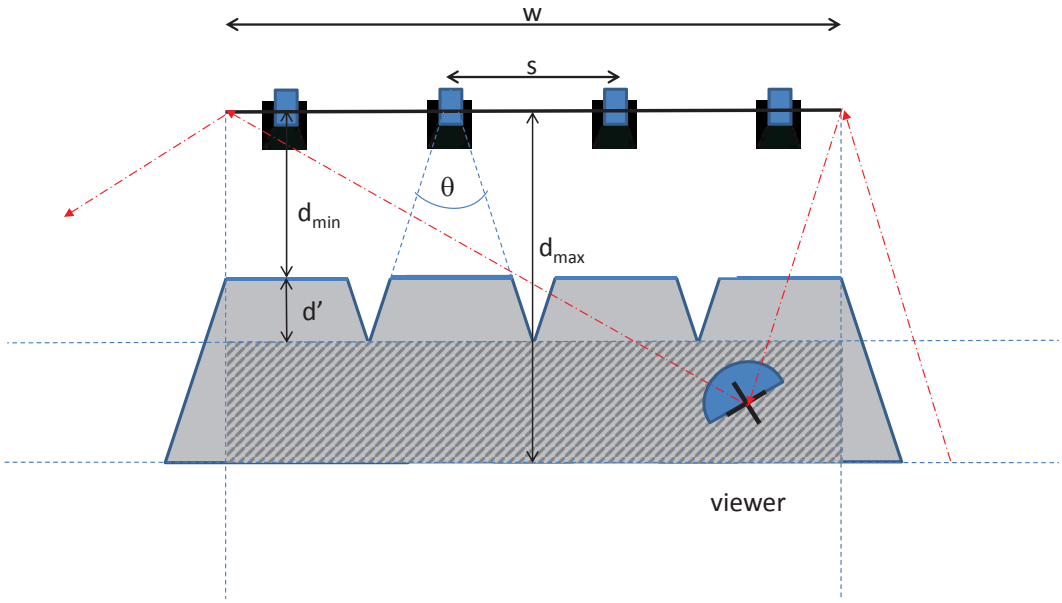


Figure 4.3: Moving space of viewer with respect to the placement of multiple RGB-D cameras

image and the viewpoint estimate are then sent to the server. The server then combines all the mirror images with the 3D background to fill in any area that may be occluded from one camera but visible at the others. The viewpoint estimate is also refined by taking a statistical average of all estimates and broadcasted back to all the clients for the rendering of the next frame. Details of the entire system can be found in Section 4.5.

4.3 Depth Denoising and Completion

A typical structured-light stereo RGB-D system such as Microsoft Kinect consists of a projector and two camera sensors. Special light patterns, typically in the infra-red (IR) spectrum, are emitted by the projector and captured by the IR CMOS camera sensor. Depth information of the sensing environment can then be inferred based

on how the light patterns are distorted in the IR images [94]. The RGB camera captures the color information and can be aligned with the IR camera by estimating the extrinsic parameters between them.

Depth images obtained by such an imaging system have two major problems: missing and distorted depth values. Figure 4.4 (left) shows a pair of typical RGB and depth images obtained by Kinect. All the black regions in the depth image contain no depth measurements. Some of these regions occur on object surface, such as the holes on the front penguin's belly. This is caused by the short distance between the object and the depth camera. Missing regions also present along the object boundaries. This represents a major source of depth error caused by the disparity between the projector and the sensor. Sometimes, they can be orders of magnitude different from their true values.

Missing and erroneous depth values can also be caused by absorption, poor reflection or even shadow reflection of the light patterns. Objects with darker colors, specular surfaces, or fine-grained surfaces like human hair are prime candidates for poor depth measurements [95]. Surface orientation also plays a role in depth measurements – as the surface normal deviates from the principal axis of the IR camera, the accuracy of the depth measurement declines and becomes unreliable near depth discontinuities [2]. Random noise in depth images may also be a result of inadequate calibration, sensor noise, and round-off error during normalization [93]. Generic image denoising and complete algorithms fail to take into account the unique problems of structured-light RGB-D systems.

Our technical contribution is the use of depth "layer" in steering the completion

process to produce well-defined depth edges. We describe a novel stochastic framework that separates the depth image into multiple layers, and combines multiple RGB-D system noise models to robustly determine the depth layer label. The goal is to denoise and complete missing values on the depth image that improve the quality for any subsequent RGB-D applications.

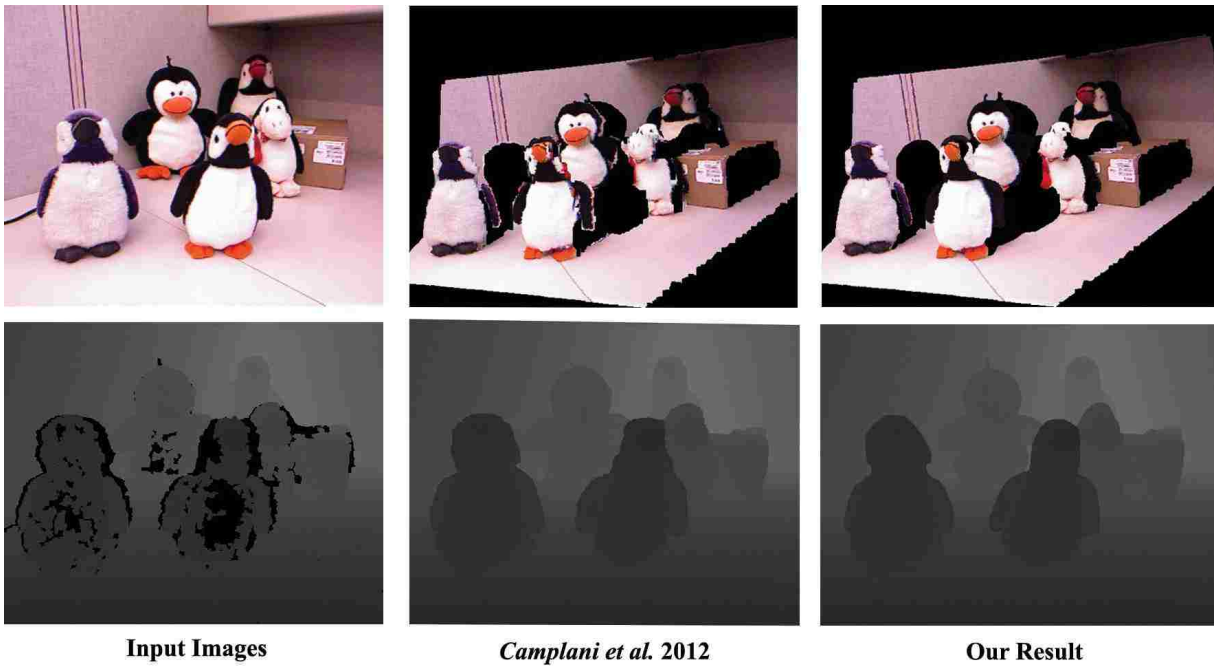


Figure 4.4: Depth Denoising and Completion. 1st column: input depth image; completed depth by Camplani et al. [1] and our method. 2nd column: input RGB image and two corresponding reconstructed virtual views

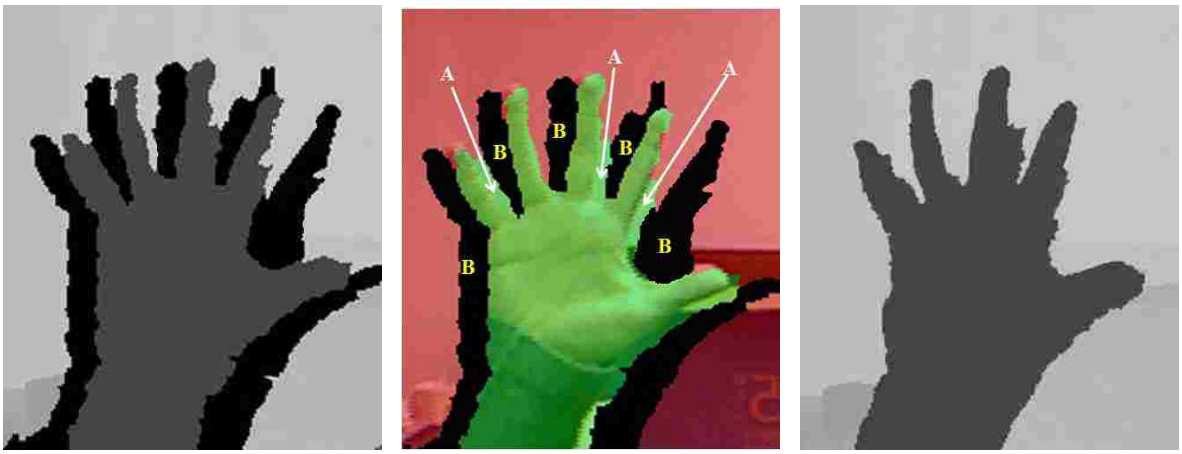
4.3.1 Problem Definition

Missing depth can be categorized into two types. The first type is the randomly distributed "small holes" on objects' surfaces. These missing values can usually be inferred by the available depth pixels in the neighboring regions. In addition, the corresponding RGB information can be used to steer the depth completion by using

techniques such as Joint Bilateral Filter [56].

The second type is the large and contiguous missing depth patches that often present along the boundary between close (foreground) and far objects (background). An example can be found in Figure 4.5a, where there are many missing depth (black) regions around the hand. This is caused by the disparity between the IR projector and the IR camera sensor. Part of background regions are visible to the IR camera but not to the projector and receive no structured light patterns. Thus, the depth values in those regions cannot be measured.

Missing and incorrect depth values due to disparity cannot be easily and correctly inferred by many existing works including Joint Bilateral Filter based schemes [56] [1] and probabilistic based schemes [3]. Figure 4.5 explains why these methods yield unsatisfactory results. The raw depth image shown in Figure 4.5a clearly indicates two distinct depth layers: foreground in dark gray and background in light gray. No depth measurements are obtained in the black region. In Figure 4.5b, we overlay semitransparent green (foreground) and red (background) layers over the RGB image as an indicator of its corresponding depth information. There are a number of background pixels wrongly labeled as foreground along the boundaries of the hand (annotated as "A"). The missing depth patches (annotated as "B") are often adjacent to "A". Most existing spatial approaches complete these missing values by using the erroneous depth in the neighboring areas. Using the color channel provides little to rectify this problem because the RGB values from "A" are very similar to the ones from "B", both of which are from the background color. The depth completion result using a joint color-depth bilateral [1] shown in Figure 4.5c is indeed quite poor.



(a) (b) (c)

Figure 4.5: (a) Depth Image; (b)Labeled Image by depth information; (c) Depth Completion by Camplani et al. [1]. The alignment between the depth and color images is based on the calibration result from Mircrosft Kinect diver

4.3.2 Proposed Method

We assume that the dynamic 3D environment can be separated into a static background and a number of dynamic foreground objects. This is a configuration typically seen in a home or office environment with a small number of individuals moving in front of the device. The RGB and depth cameras are assumed to be extrinsically aligned and temporally synchronized.

After an initial step of offline training on background-only frames, our online algorithm consists of two main phases: layer labeling followed by depth denoising and completion. In the first phase, each pixel of the incoming frame is labeled by different layers via a probabilistic framework that incorporates a data measurement model and a smoothing neighborhood model based on available observations. Maximum A Posteriori (MAP) estimation is used in the labeling to prevent blurring along depth discontinuities.

In the second phase, the labels estimated in the first phase are used to steer the removal of outlier and the completion of missing depth values, from either the background model or from neighboring depth values with the same labels. The robust labeling allows us to preserve the shape of object boundary and prevent noise propagation across objects with significant depth differences.

4.3.3 Layer-driven Stochastic Models of Depth Measurements

The probabilistic graphical model that describes the multi-layered RGB-D measurement process is shown in Figure 4.6. Let G be the support of the 2D color and depth images. At each pixel location $s \in G$, X_s denotes the latent random integer variable indicating which layer the pixel belongs to. X_s can assume any value in the set $\{-n', -n'+1, \ldots, -1\} \cup \{1, 2, \ldots, n\}$. Negative numbers are layers from the static background while positive numbers refer to layers in the foreground. The larger the layer number is, the closer it is to the camera. The number of background layers n' and the number of foreground layers n are determined based on the observed data and are the same for all pixels. The approaches to estimate n and n' will be described in Section 4.3.3.

Spatially, a layer pixel is connected to its four closest neighbors, generically referred to as X_t. All the labels over the entire image thus form a Markov Random Field (MRF) and the spatial relationship between adjacent labels is governed by an edge factor $\psi(X_s, X_t, f_{st})$, where f_{st} is the measured similarity between the two pixels. Each layer label X also has its evidence potential function $\phi(X_s)$ based on the measurement Bayesian Network (BN) shown in the lower half of Figure 4.6. As all

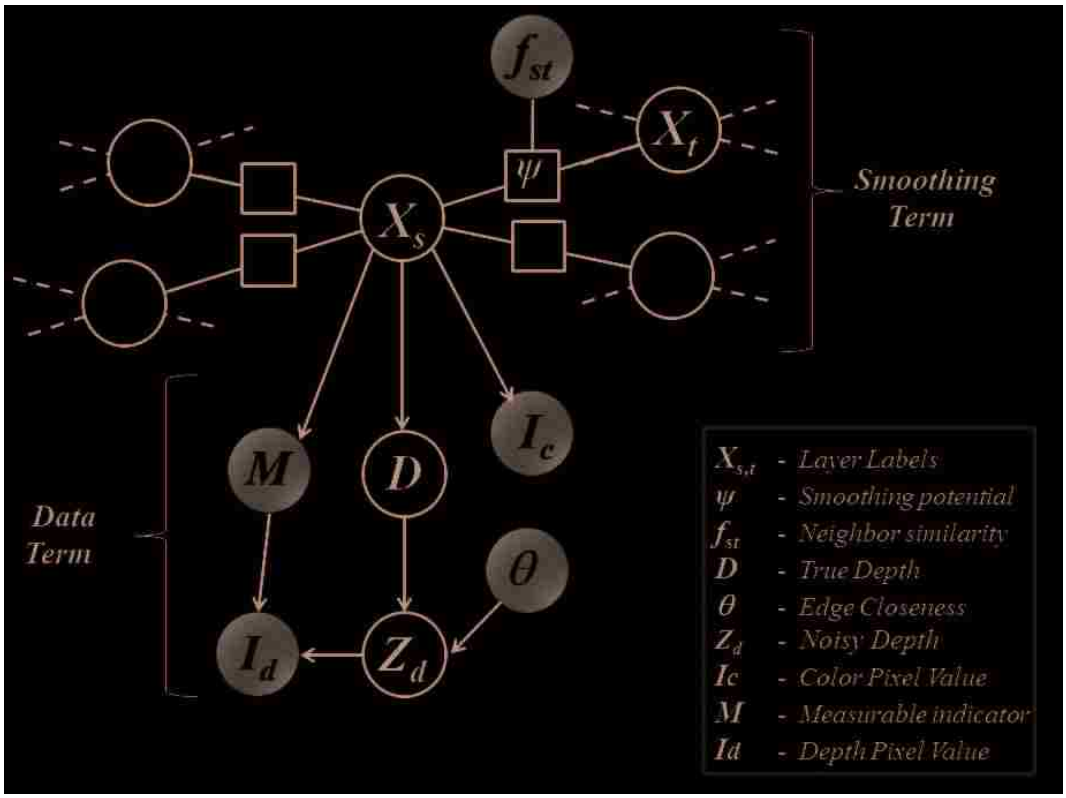


Figure 4.6: RGB-D model: the smoothing term on top specifies the constraints between neighboring labels while the data term below describes the measurement process. Darken nodes are actual measurements, white nodes are hidden variables, and square nodes are factors.

the measurements are made at the same pixel, the subscript s is omitted and $\phi(X)$ is defined as follows:

$$\phi(X) \triangleq P(X) \cdot P(I_c|X) \cdot P(M|X) \cdot P(I_d|M, \theta, X) \tag{4.1}$$

where

$$P(I_d|M, \theta, X) = \int_D P(D|X) \int_{Z_d} P(I_d|Z_d, M) P(Z_d|D, \theta) \tag{4.2}$$

I_c represents the observed color values. D represents the true but unobserved depth values. D is corrupted by an additive Gaussian noise which produces a noisy mea-

surement Z_d. The noise variance is determined by the closeness measurement θ to the nearest edge. Due to the missing depth problem, Z_d may not be directly observable. We thus introduce an observable depth indicator random variable M which is 1 if the depth value is observed and 0 otherwise. Combining these two random variables results in the observable depth value $I_d = MZ_d$. Using this probabilistic model, layer labeling can be formulated as a Maximum a posteriori or MAP problem:

$$X_G^{MAP} \triangleq \arg\max_{x_G} \sum_s \log \phi(x_s) + \sum_{(s,t)\in G} \log \psi(x_s, x_t, f_{st}) \tag{4.3}$$

Our choice of parametrization allows $\psi(x_s, x_t)$ and $\phi(x_s)$ to be computed. While the complexity of the exact solution to the MAP problem is exponential in the image size, which is known to be NP hard. To improve the computation speed, various approximation algorithms have been proposed for a global optimization, such as Graph Cuts, or Loopy belief propagation [97]. Here we used max-product based loopy belief propagation to approximate the inference. One major reason of choosing it than graph cuts is that belief propagation is more efficient in processing video sequence: the output of previous frame can be used as the initial value of messages for current frame, which makes the convergence speed improved.

Data Term

In building the data term for the graphical model, the four layer distributions $P(X)$, $P(M|X)$, $P(D|X)$, and $P(I_c|X)$ are estimated based on both offline training data and online data. The depth distribution $P(D, X) = P(X)P(D|X)$ is modeled as a mixture of Gaussian (MOG) model while the color distribution $P(I_c, X) = P(X)P(I_c|X)$

is modeled as multiple color histograms on the quantized HSV space, one for each layer. The observable depth indicator distribution $P(M, X) = P(X)P(M|X)$ is based a simple Bernoulli distribution for each layer. In fact, the parameter estimation for the depth indicator is a simpler version of the color distribution. As such, our discussion will focus on the depth and color distributions. The parameter estimation process is summarized in Figure 4.7.

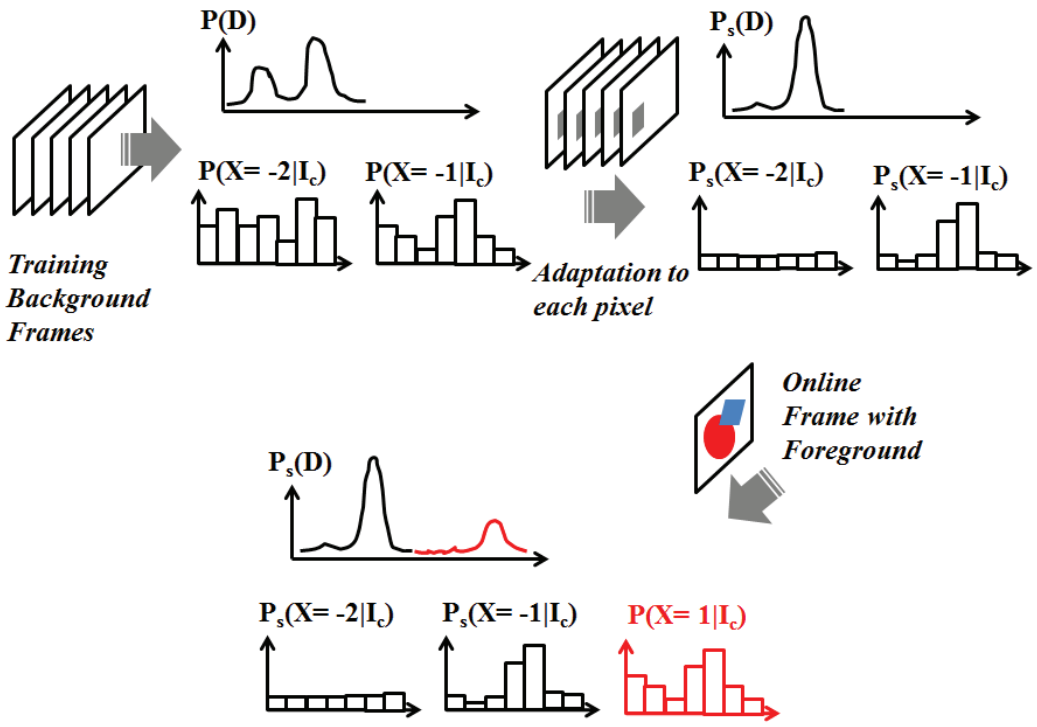


Figure 4.7: Parameter estimation for depth and color layer distributions

During the offline training phase, we estimate the parameters for the negative (background) depth layers based on a set of training RGB-D frames of the static background. There are two phases of the estimation: global estimation and local adaptation. During the global estimation, all the pixels with both color and depth

measurements will be aggregated to estimate a single pair of $P(D, X)$ and $P(I_c, X)$ using the Expectation-Maximization (EM) approach. $P(D, X)$ is initialized by K-means and $P(I_c, X)$ is initialized as an uniform distribution. It is important to note that the concept of layers is based only on depth but not on color. As such, the EM process is primarily driven by the depth data in the sense that the E-step only estimates the layer posterior $P(X|D)$ for the depth but not the color. During the M-step, we use the depth data to update the estimates for the layer prior $P(X)$ and the depth layer conditional $P(D|X)$, only use the color data to update the color layer conditional $P(I_c|X)$ using the posterior probability $P(X|D)$ of the co-located depth pixel. As a result, two pixels with the same color value can have different contributions to different layers. Different number of layers are tested and the optimal number n' is determined by using the Bayesian Information Criterion (BIC) on the depth data [98]. The example in Figure 4.7 shows this first step to have two separate background layers and obtain the global color and depth models. In the second phase, the global distributions are adapted to each individual pixel by using only the temporal data at that pixel location. Sequential exponential weighing scheme is used for the adaption. For example, the local mean for layer $X_s = i$ at location s is updated by a new depth value d_{new} as follows:

$$\begin{aligned} \mu_{s,i}^{(t+1)} := {} & \lambda P^{(t)}(X_s = i | D_s = d_{new}) \cdot d_{new} \\ & + \big(1 - \lambda P^{(t)}(X_s = i | D_s = d_{new})\big) \cdot \mu_{s,i}^{(t)} \end{aligned} \tag{4.4}$$

t represents the iteration step and λ controls the rate of adaptation which is empirically set to 0.3. All the other parameters are updated in a similar fashion. Similar

to the global distributions, the two layer conditional probabilities are updated based on the corresponding color or depth data. The layer prior is updated using the depth data if available, or the color data if the depth data is missing. After the adaptation, the parameters can better describe the characteristics of the local pixel – Figure 4.7 illustrates this idea where the local depth distribution $P_s(D)$ and the color posterior $P_s(X|I_c)$ clearly indicate that this pixel is more likely to belong to layer -1.

The foreground layer distributions are estimated online. To deliver a real-time response and cope with fast moving objects, the foreground distributions are estimated for every frame. As no temporal data is maintained, we only estimate global distributions using an approach identical to that of the background global distribution. The training data are obtained based on only those pixels with valid depth measurements and very low background posterior probability, i.e. $\max_{i=1,\ldots,n'} P_s(X_s = -i|D_s) < \epsilon$ for a small fixed ϵ. To obtain the full range of $P(X)$, $P(D|X)$, and $P(I_c|X)$, we also need prior probabilities for foreground and background. We simply set them to be equally likely for our experiments though better performance may be possible with more foreground training data. In Figure 4.7, the foreground components of the layer distributions are in red color.

Next, the noisy depth measurement Z_d is modeled based on an additive Gaussian model:

$$Z_d \triangleq D + N, \quad \text{with } N \sim \mathcal{N}(0, \sigma_\theta^2) \text{ and } N \perp D. \tag{4.5}$$

The noise standard deviation σ_θ reflects the uncertainty in the depth measurement. As argued in Section 4.3.1, erroneous depth measurements occur predominantly near

object boundaries. To model this effect, we apply an edge detector on the depth map and use the spatial distance θ to the closest depth edge as a reliability measure. The noise variance σ_θ is modeled as a deterministic logistic function given below:

$$\sigma_\theta := \frac{a}{1 + e^{-(b\theta - c)}} \tag{4.6}$$

a and b are constant scaling parameters. c/b is a distance threshold beyond which the depth value is relatively noise free. This simple model is easy to compute, though a more sophisticated one incorporating surface normal, texture, and color can be used in a similar fashion.

Finally, we present a simple approach to evaluate the integrals in Equation 4.2. Note that the actual depth measurement $I_d = MZ_d$ implies that $P(I_d = i_d | Z_d = z, M = m) = \delta_{mi_d}(z)$, the dirac delta function with the only non-zero value at $z = mi_d$. Substituting this into (4.2) results in the following simplification:

$$\begin{aligned} & P(I_d = i_d | M = m, \theta, X = x) \\ & \quad = \int_e P(D = e | X = x) P(Z_d = mi_d | D = e, \theta)\, de \\ & \quad = P(Z_d = mi_d | \theta, X = x) \end{aligned} \tag{4.7}$$

Given $X = x$, Z_d and D are multivariate Gaussian with the following distribution:

$$\begin{bmatrix} Z_d \\ D \end{bmatrix} \Bigg| X = x \;\; \sim \;\; \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_x \end{bmatrix}, \begin{bmatrix} \sigma_x^2 + \sigma_\theta^2 & \sigma_x^2 \\ \sigma_x^2 & \sigma_x^2 \end{bmatrix} \right) \tag{4.8}$$

Thus, $Z_d | X = x \;\sim\; \mathcal{N}(\mu_x, \sigma_x^2 + \sigma_\theta^2)$ and (4.7) can be numerically evaluated.

Smoothing Term

For the spatial MRF, the edge potential $\psi(X_s, X_t)$ is defined based on the similarity in color and depth between the neighboring pixels:

$$\psi(X_s, X_t, f_{st}) \triangleq \begin{cases} f_{st} & \text{for } X_s = X_t, \\ \frac{1}{n+n'}\left[1 - f_{st}\right] & \text{otherwise} \end{cases} \tag{4.9}$$

The similarity strength f_{st} is a feature based on how close the color and depth of the neighboring pixels are. It is modeled using the following equation:

$$f_{st} = \max\{\alpha \exp(-C_{st}) + (1-\alpha)\exp(-D_{st}), n_f\} \tag{4.10}$$

The color similarity ratio C_{st} and the depth similarity ratio D_{st} are defined in a similar fashion [99]:

$$C_{st} = \frac{\| I_c(s) - I_c(t) \|^2}{k_c \left\langle \| I_c(s) - I_c(\tilde{t}) \|^2 \right\rangle} \tag{4.11}$$

$$D_{st} = \frac{|I_d(s) - I_d(t)|^2}{k_d \left\langle |I_d(s) - I_d(\tilde{t})|^2 \right\rangle} \tag{4.12}$$

$\langle \cdot \rangle$ denotes the average operator. $\tilde{t}$ represents any one of the eight neighboring pixels of s (including pixel t). k_c and k_d are normalization constants so that the two terms are of the same range. n_f is the minimum similarity to prevent the potential function from becoming zero. If either depth measurement is not present, n_f will be used.

The parameter $\alpha \in [0, 1]$ is a trade-off between depth and color information. If the depth measurements are reliable, most of the weight should be assigned to depth values as they are more reliable for foreground/background labeling; if the depth measurements are unreliable, they should not be used at all in computing the edge potential. Similar to the approach used in Section 4.3.3, α is defined as the logistic

function of the distances θ_s and θ_t of the two neighboring pixels to the closest edge:

$$\alpha \triangleq f\left(\frac{\theta_s+\theta_t}{2}\right) \text{ with } f(\theta) \triangleq \frac{1}{1+e^{-(b\theta-c)}} \tag{4.13}$$

Depth Image Completion

After assigning each pixel with a layer label, we need to identify erroneous depth measurements and complete missing depth values. The erroneous depth values are essentially outliers that are significantly different from other depth values in the neighborhood. However, as most measurement errors occur around object boundaries, it is imperative not to mistake true depth discontinuities as wrong depth values. The layer labels allow us to separate pixels that are likely to have come from objects at completely different depths. To determine if a depth pixel is an outlier, we robustly estimate the depth distribution in the neighborhood around the pixel via a RANSAC-like procedure. First, we only consider depth values in the neighborhood that share the same label as the target pixel. Then, multiple small sets of random sample pixels are drawn and a Gaussian distribution is estimated for each set. If only a small fraction of the neighborhood can be fit within two standard deviations from the mean of a sample distribution, this distribution is likely to contain outlier samples and is thus discarded. Among those that survive the robustness test, the one with the smallest variance is used and the target depth pixel is declared an outlier if it is beyond two standard deviations from the mean. The outlier depth pixel will join the rest of the missing depth pixels and will be completing using a joint color-depth bilateral filtering scheme similar to that in [56]. The only difference is that we only

consider the contributions from neighboring depth pixels that have the same layer label as the center pixel.

4.4 Offline Background Scanning

For the off-line background scanning, our scheme is based on a variation of Kinect Fusion method as proposed in [2]. The scanning process involves moving an RGB-D camera to capture a sequence of RGB-D frames of the entire background environment. The heart of the scanning process is a voxel data structure called the truncated signed distance function or TSDF [100]. The TSDF structure can be used to aggregate multiple depth images into the same coordinate system and to generate 3D point clouds for rendering. Iterative Closest Point or ICP algorithm is used to estimate online the camera pose for each new depth image by computing a rigid transformation between the depth image with that predicted based on all the data already accumulated in the TSDF structure.

However, it is problematic when the above method is used in scanning large planar regions such as wall surfaces in a room. The lack of depth variations on planar surfaces makes the camera pose estimation an ill-conditioned problem. Figure 4.8a shows a virtual view of a wall rendered from a 3D structure created by applying the ICP algorithm from [2] to align 50 frames of moving depth images. One can clearly see that the scene points are grossly misaligned. The misalignment is caused by the failure of ICP in identifying correct correspondences between planar point clouds of successive frames. Such misalignment errors accumulate over multiple frames, making it impossible to process a longer sequence. This is a significant shortcoming of ICP

as planar surfaces are common in indoor environments. In Section 4.4.1, we first explain why existing 3D scanning techniques often yield unsatisfactory results for planar surface. The details of our scheme for improving the accuracy are presented in Section 4.4.2

(a) (b)

Figure 4.8: Comparison results of scanning planar surface: (a) result by [2] (b) result by our method

4.4.1 Problem Definition

To understand the problem on aligning planar surfaces, let us first review the basic procedure of ICP as summarized in Algorithm 1 [101]. Given two consecutive frames of 3D point clouds F_{t-1} and F_t, at each iteration s, the algorithm refines a rotation matrix $R^{(s)}$ and a translation vector $t^{(s)}$, which are applied to F_t to best align each point in F_{t-1} to its closest point in F_t. The empirically-determined parameter ϵ excludes correspondences that are too far apart to be considered as reasonable.

When ICP initially identifies the closest points between the two point clouds, there could be many false correspondences. The goal of ICP is to improve these correspondences by moving two point clouds closer to each other in each iteration. However, the above procedure may fail if the majority of the 3D points fall on a planar

Require: $F_{t-1} = \{p_1, p_2, \ldots, p_m\} \in \mathbb{R}^3$
$F_t = \{q_1, q_2, \ldots, q_n\} \in \mathbb{R}^3$

1: Initialization:
$s := 0$ and $F^{(s)} := F_t$

2: Identify closest points: $\forall p_i \in F_{t-1}$
$d_i := \min_{q \in F^{(s)}} \|p_i - q\|_2$
$f^{(s)}(p_i) := \begin{cases} \arg\min_{q \in F^{(s)}} \|p_i - q\|_2, d_i \leq \epsilon \\ \text{unmatched, otherwise} \end{cases}$

3: Find $R^{(s)}$ and $t^{(s)}$ to minimize the average of
$\|p_i - (R^{(s)} \cdot f^{(s)}(p_i) + t^{(s)})\|_2^2$
among all p_i's with a matching $f^{(s)}(p_i)$

4: Refinement:
$F^{(s+1)} := \{q_i' : q_i' := R^{(s)} \cdot q_i + t^{(s)}\}$

5: $s := s + 1$

6: Go back to step 2 until error in step 3 is below a threshold

Algorithm 1: ICP

surface. This is illustrated in Figure 4.9. The red points from each plane indicate the true correspondences. But the closest-point search wrongly assigns the green points from F_{t-1} to match the points in F_t. If there were significant depth variations among the 3D points, no rigid transformation could produce a good match between these wrong correspondences and step 3 of the ICP algorithm merely produces a transformation that moves the two clouds closer. However, for a planar surface, these wrong correspondences may lead to a wrong rigid transformation that can completely align the two planes. The lack of depth variations prevents the in-plane rotation and the translation along the $x - y$ plane to be effectively estimated. As such, we have an underdetermined system and the ICP prematurely terminates without providing the true alignment. Notice that such misalignment errors accumulate over time and thereby significantly affect the subsequent reconstruction of the 3D structure.

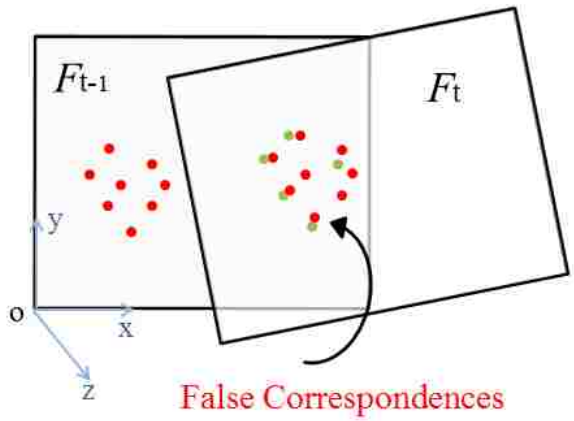


Figure 4.9: How ICP fails to align planar structures

4.4.2 Proposed Background Scanning Pipeline

Figure 4.10 is an overview of our proposed Background Scanning system. Our camera pose estimation starts by first extracting the SIFT features from the color frames and searching for the closest match between frames [102]. If all the SIFT feature points fall on the same plane, the matching correspondences between frames would be related by a planar homography. However, if the scene structure is more complex, we need a more robust procedure to identify the subset of correspondences that could fit well within a planar homography. To this end, we use a RANSAC-like procedure to identify such a subset and to estimate the optimal homography between correspondences [103]: all the correspondences are first used to estimate a homography matrix. We then apply the estimated homography to map one set of points to the other and eliminate those pairs that are too far apart. We repeat this process until it converges to a stable subset of correspondences that are well described by a single homography matrix. Then these corresponding pairs, along with the associated depth measurements, are projected back onto the 3D space to be used as the initial point

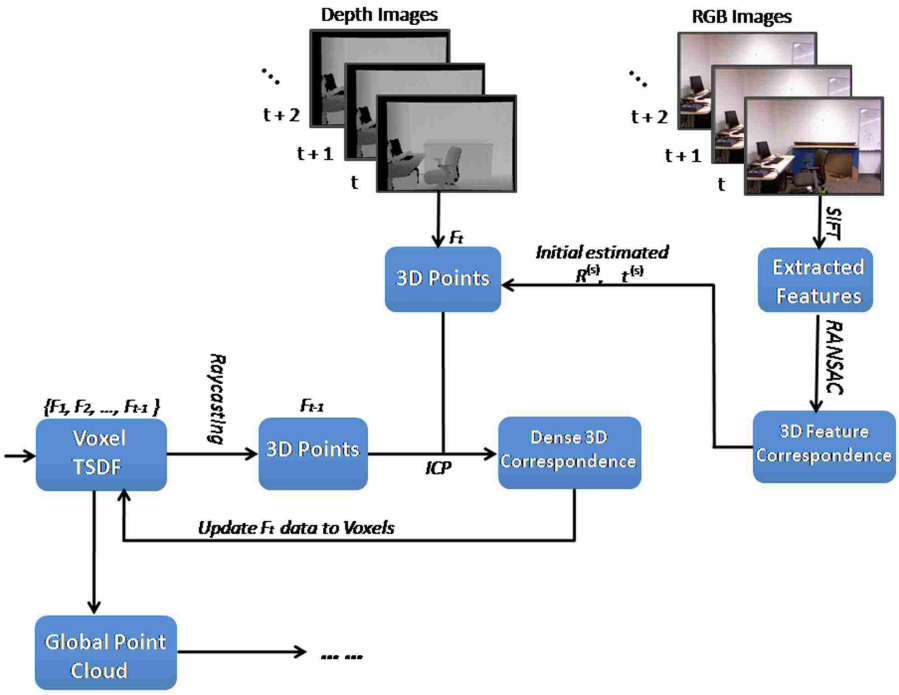


Figure 4.10: Overview of our background scanning system

correspondences for the ICP algorithm.

Note that the SIFT correspondences are based on the current and previous color frames captured by the camera while the ICP is used to align the current depth frame with the TSDF voxel structure. Thus, we need to first raycast the TSDF structure using the estimated camera pose of the previous frame to recreate the point cloud from that frame: given a TSDF structure, we can generate the depth map by first raycasting from the optical center of the virtual camera and traversing through the voxel structure. We adopt the fast voxel traversal algorithm in [104]. 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. Truncated values are skipped over to decrease the voxel traversal time. Once the zero crossing region is identified, linear interpolation is utilized to predict a surface point based on the TSDF values of the

two boundary voxels. All the identified surface points form a 3D point cloud as if they were captured from a virtual depth camera placed at (X, Y, Z).

This step serves two different purposes: first, it allows us to generate the viewpoint-dependent mirror view which will be merged with the dynamic view captured by the stationary RGB-D camera network. Details of this process are described in Section 4.5. Second, it provides a reference 3D point cloud that can be used to estimate the unknown camera pose of the moving RGB-D camera during the background scanning.

After the recreation of the point cloud from the previous frame F_{t-1}, we match it with the transformed point cloud of current frame F_t (the transformation $R^{(s)}$ and $t^{(s)}$ are estimated directly from the SIFT correspondence). To ensure a robust matching, we again use RANSAC to find the inliers as a subroutine within ICP – outliers are iteratively removed if they do not agree with the estimated transformation until the procedure converges to a stable set of correspondences. In each iteration of ICP, $R^{(s)}$ and $t^{(s)}$ are updated by minimizing the combined point-plane energy for all existing point-pairs between F_t and F_{t-1}. The projective data association method in [2] is utilized to find point correspondences. After the alignment, we can then proceed to update the TSDF structure using the current frame data: an exponentially weighted average is applied to the TSDF value at each voxel that is sufficiently close to a 3D point from the current frame. This averaging helps to remove noises inherent in the depth image data and multiple passes of all the depth images are usually required to produce a consistent and stable TSDF structure.

4.5 System Integration and Mirror Image Synthesis

The overall implementation framework of the proposed mirror rendering system is shown in Figure 4.11. There are four main components in the system:

1. Background Scanning
2. Depth Denoising and Completion (DDC)
3. Data acquisition and Mirror-view Generation (DMG)
4. Final View Synthesis (FVS)

The background scanning, as described in Section 4.4, is an off-line process that builds the 3D model of the background. To support large mirror display and to cope with the high on-line computation complexity of the remaining processes, we adopt a server-client distributed architecture. Each RGB-D camera is connected to a client computer which runs the DDC and DMG modules. The DMG module receives the input from the RGB-D camera and forwards the data to the DDC module for preprocessing as described in Section 4.3. The processed results are sent back to DMG for generating 3D points and rendering mirror views. These rendered views are delivered to the centralized server where the FVS module synthesizes the final image for display. Details of DMG and FVS are described below.

For each RGB-D camera, the DMG uses the camera's intrinsic parameters to obtain a 3D point for each depth pixel by applying the inverse camera projection operator. An off-line calibration process is used to obtain the extrinsic relationship of the depth camera with respect to the coordinate system of the color camera. These

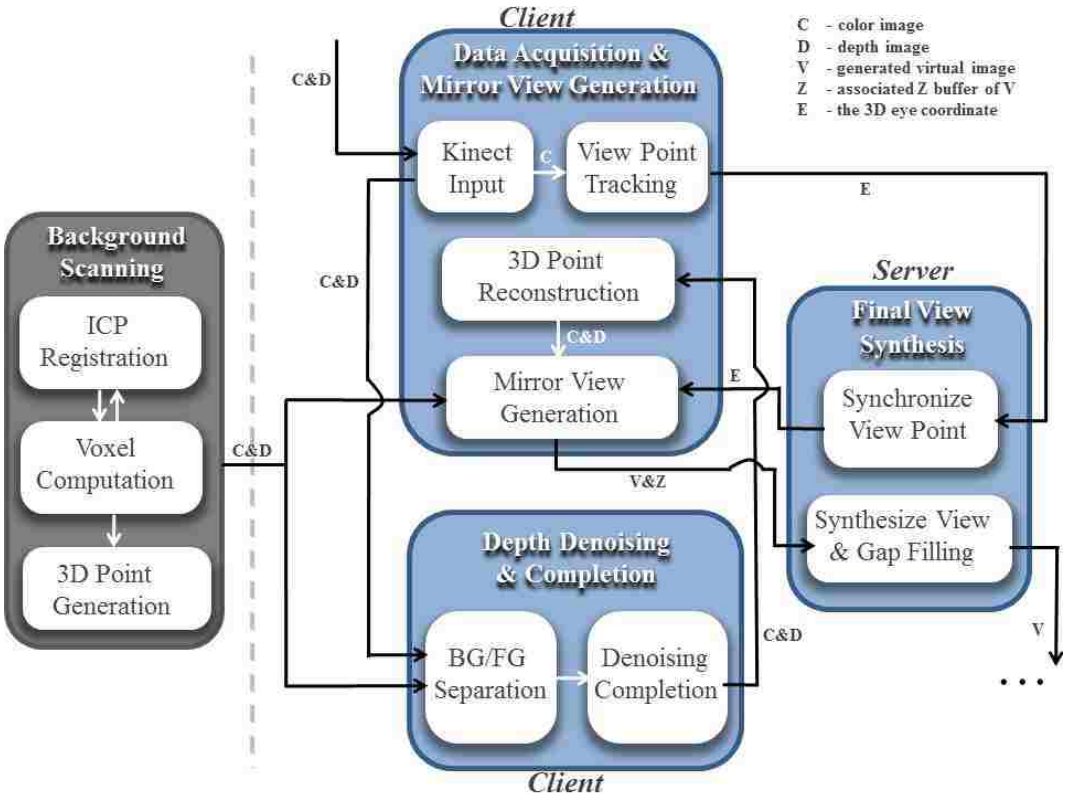


Figure 4.11: RGB-D framework for Virtual Mirror System

extrinsic parameters allow us to associate each color pixel value with a 3D point. As a result, each client's DMG provides a set of colored 3D point cloud in the form of a 6D-tuple (X, Y, Z, R, G, B). As there are multiple RGB-D cameras, we need to transform the 3D point cloud of each client to a common coordinate system. We choose one color camera (camera 1) as the reference coordinate origin. For each of the remaining color cameras i, we estimate the rotation matrix R_{1i} and translation vector T_{1i} with respect to the reference camera using the standard camera calibration toolbox. These extrinsic parameters are shared among all client's DMG's which will

apply the transformation on their own point cloud:

$$\begin{pmatrix} X_i' \\ Y_i' \\ Z_i' \end{pmatrix} = R_{1i} \cdot \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} + (-R_{1i}T_{1i})$$

where $(X_i, Y_i, Z_i)^T$ represents a 3D scene point computed from the i^{th} color camera. $(X_i', Y_i', Z_i')^T$ represents a 3D point in the reference coordinate system.

To provide viewpoint-dependent viewing, each client's DMG also tracks the viewpoint of the viewer. As our display only renders a monocular view, we track the mid-point between the two eyes of the viewer. We adopt the scheme in [105] to detect the image positions of the eyes on each RGB frame and to temporally track the center of these two detected positions with a Kalman filter. The final 3D coordinate of the viewpoint is then estimated to be the 3D point that minimizes the sum of distances to the line formed between the optical center of each camera and the center of the two eyes on the camera plane. The last step is performed at the server's FVS which will broadcast the final estimate back to all the clients for rendering.

After obtaining the transformed RGB-D data and the estimated viewpoint, the next step is to render the mirror image from an arbitrarily-positioned virtual plane mirror onto the display surface. For each 3D scene point, the rendering step is equivalent to identifying the reflection point on the mirror and the display point on the display surface. The identification of the reflection and display points can be viewed as two consecutive virtual camera projections as shown in Figure 4.12. The reflection point can be viewed as the projection of the scene point onto a virtual camera, denoted as "Virtual Camera 1" in Figure 4.12, with the image plane at the

mirror and the optical center at the mirror image V' of the viewpoint V. The display point is the projection of the reflection point onto another virtual camera, denoted as "Virtual Camera 2" in Figure 4.12, with the image plane at the display plane and the optical center at V.

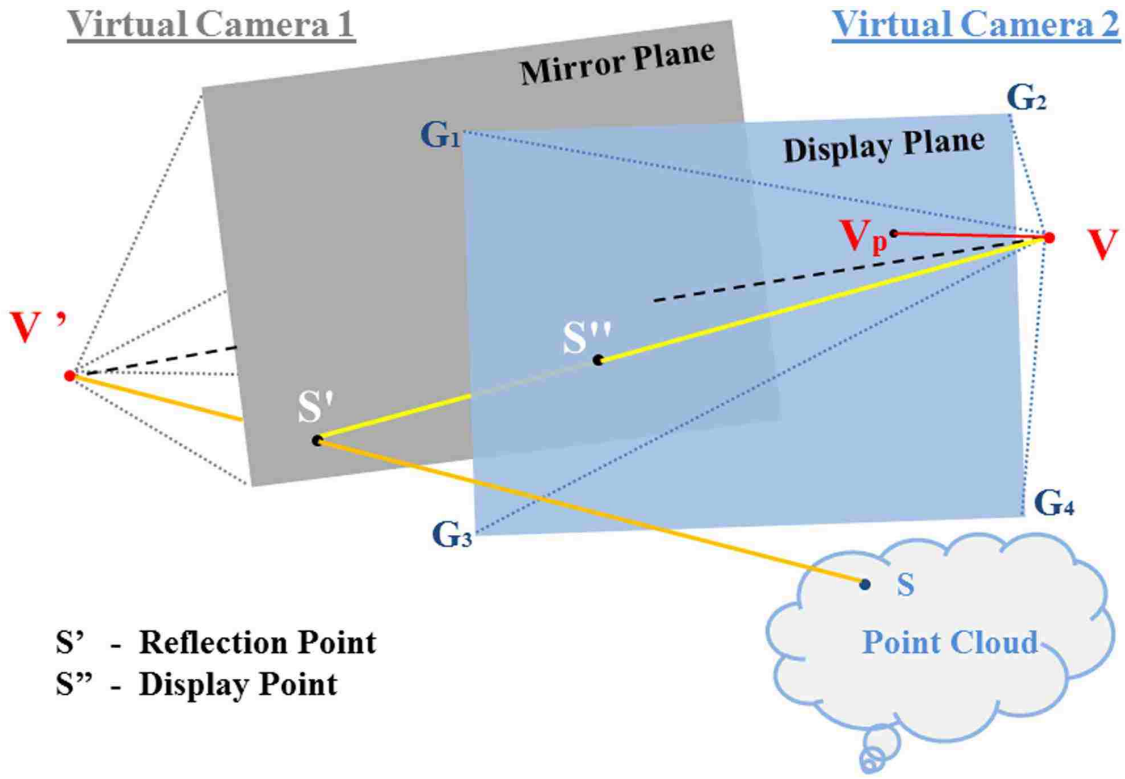


Figure 4.12: Modified camera projection models for display rendering

Thus, the rendering problem boils down to estimating the camera projection matrix for each of the two virtual cameras. The approaches to estimate the camera matrix for both cameras are identical, and we use the display surface camera as an example. The focal length f is simply the distance between V and the display surface. The image center (c_x, c_y) on the display plane is the perpendicular projection V_p of V on the display plane relative to the top-left corner G_1. As for the extrinsic parameters, the rotation matrix R can be inferred by the orientation of the display

plane. For example, the plane normal is computed as $\overrightarrow{n} = \overrightarrow{G_1G_2} \times \overrightarrow{G_1G_2}$. R is determined according to the angle subtended by $\overrightarrow{n}$ and the Z axis in the world coordinate system. The translation T can be computed directly based on the distance between V and the origin of the world coordinate system.

Once all the above parameters are computed, the corresponding image on the display can be rendered based on the display points, denoted by S''. For each pixel on the display surface, there might be zero to multiple corresponding display points. If there is only one display point, that pixel will assume the color value associated with the corresponding 3D scene point S. If there is more than one display point, they can either be from the same 3D scene point but originated from different cameras, or they can be different scene points that fall on the same 3D ray V''. For the first case, their depth values would be close to each other. For the second case, their depth values would be far apart and the one closest to the mirror would occlude the rest. This suggests a simple procedure of first clustering all the scene points that share similar depth values and then selecting the group that is closest to the mirror. To compute the final color among all points in the winning cluster, we use the scheme of *winner takes all* to select the one that best aligns with the viewpoint [106]. To handle the occlusion issue, a z-buffer value is also computed for each pixel defined as the distance from the corresponding scene point to the mirror image of the viewpoint.

Finally, each client DMG sends its mirror image and z-buffer image to the server FVS for rendering. At the FVS, it also uses the stored background model to create the associated mirror and z-buffer images. The FVS then traverses each pixel on all

the mirror images and uses the accompanied z-buffer as a guide to determine the rendering order. We separate the rendering into two phases based on the z-buffer values – those that are at or closer than the foreground viewer and those that are at the background. These two sets typically have very different z-buffer values. The FVS first starts with the latter group with scene points that are far away and performs interpolation on *both depth and color* values to fill in small gaps. These interpolated values are inserted back to the other group as if they are from the true 3D point clouds. In the second phase, the FVS will render all the foreground pixels, select the correct color value based on both 3D point clouds and interpolated results, and finally perform one more round of interpolation just on the color values. Such a layered approach provides sharper object boundaries than a simple interpolation on the color image as it respects the inherent depth values. It is possible to increase the number of depth levels to create a better rendering, but two levels are sufficient for our application.

4.6 Experimental Results

A number of experiments are conducted to demonstrate the effectiveness and accuracy of our proposed system. We first compare the proposed depth denoising and completion algorithm with other state-of-the-art schemes to demonstrate its performance in providing high-quality depth images. We then present results on our background scanning procedure. For the experiments on the entire virtual mirror system, we measure the rendering accuracy with a real mirror and present the improvement of the rendering results with depth denoising and background scanning. Computational

Table 4.1: Time Performance Evaluation

Component	Time (seconds)
Data Term	0.115
Smoothing Term	0.032
EM Iteration	0.095
LBP Iteration	0.641

performance of the mirror system is also measured and presented.

4.6.1 Evaluation of Depth Denoising and Completion

We use a Microsoft Kinect (model LPF-00004) with the PrimseSense driver [107] to capture depth and RGB images at a resolution of 640×480. The proposed algorithm is implemented in C++ with the OpenCV library. The experiment is conducted on a computer with the following hardware settings: Intel Core(TM) i7-2820QM CPU at 2.30 GHz and 8.0Gb of RAM. For the static background training, we captured 100 images for each scene to obtain the background color and depth statistics. The speed performance for each component of our system is shown in table 4.1.

Figure 4.13 shows a simple example with one foreground and one background layer. We can see that the background wall is clearly visible between the foreground leaves and the rapid spatial changes of depth layers lead to significant measurement error shown in the original depth image in the top right. Our algorithm correctly identifies the number of layers and produces an accurate segmentation mask in the bottom left. Guided by this segmentation mask, our algorithm corrects and completes the depth values in the bottom right.

Figure 4.14 presents a more complex example with two foreground and two back-

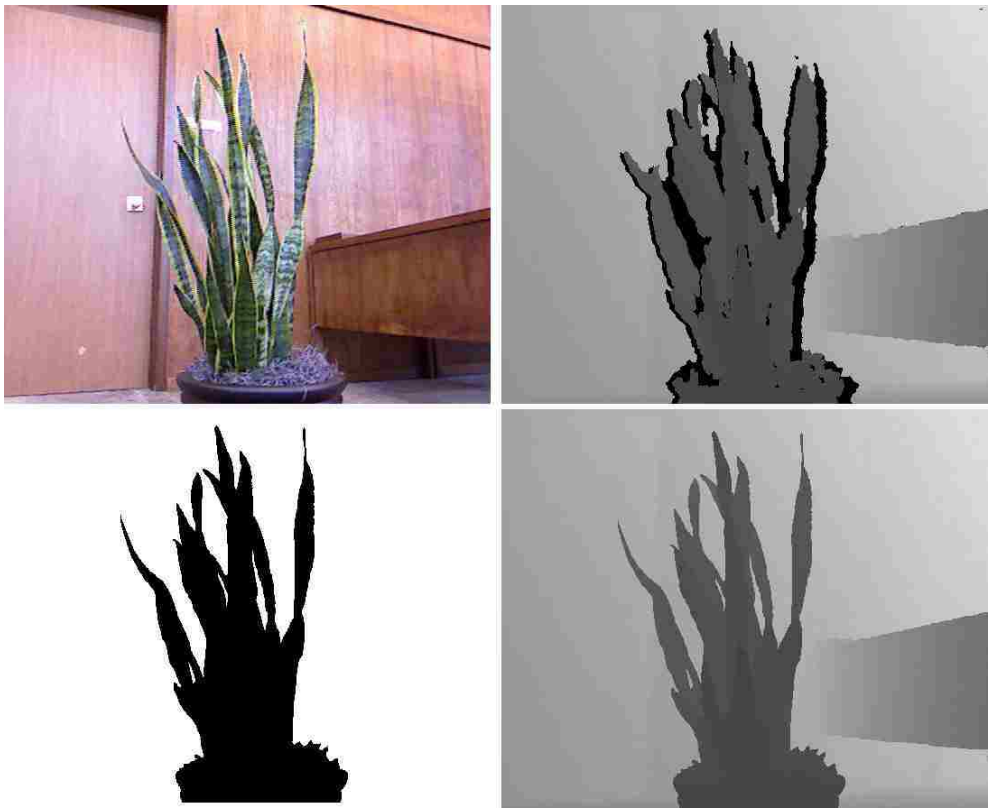


Figure 4.13: Scene with Two Layers: input RGB and depth images (top); layer labeling result and completed depth steered by layers (bottom).

ground layers. The two foreground layers are the head model (layer 2) as well as the books and the ball (layer 1). The two background layers are the white board and the table (layer -1) as well as the rest of the region (layer -2). Notice that each layer can have objects of varying depths. The curves and straight lines from different objects are well preserved in the output result shown in the bottom right.

We have also compared our scheme with other works. In particular, we have chosen [1] which represents one of the most recent efforts in using joint color-depth bilateral filtering for depth completion, and [3] which uses a similar MRF as ours in providing spatial smoothing. Figure 4.15, 4.16 presents a side-by-side comparison between these methods and ours. We have selected two sequences for comparison.

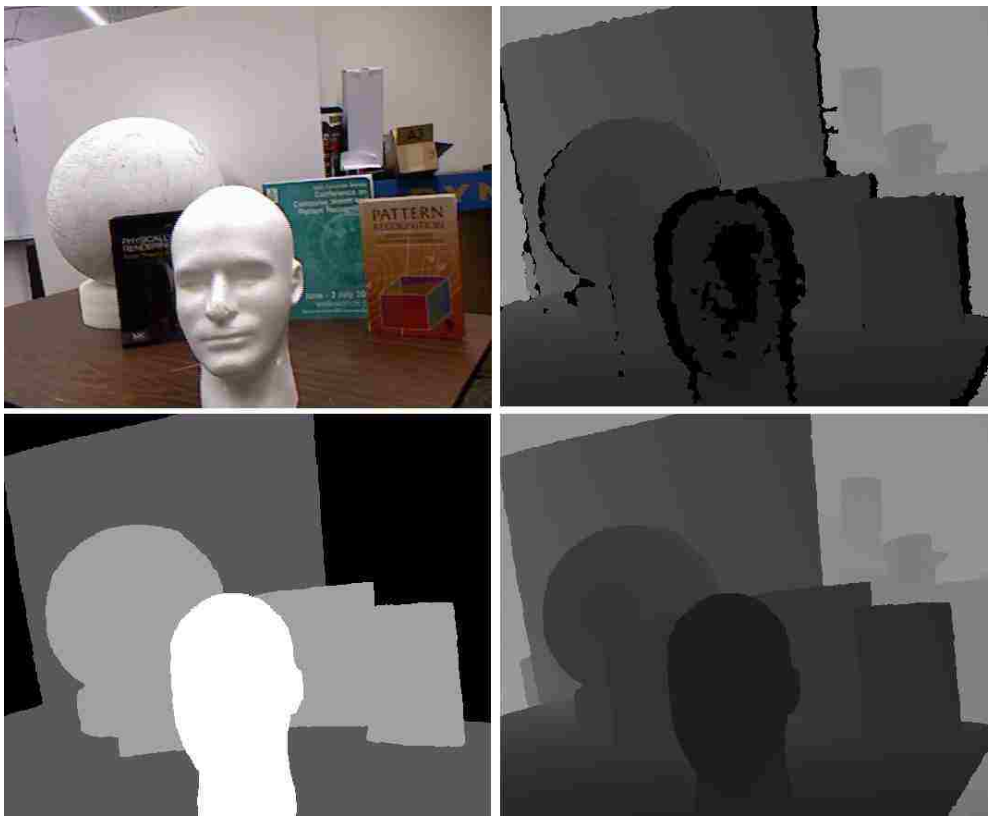


Figure 4.14: Scene with Multiple Foreground and Background Layers: input RGB and depth images (top); layer labeling result and completed depth steered by layers (bottom).

Except for the last row, the original data is shown in the first column, results from [1] in second, [3] in third, and ours in the last column. Two scenes are used in the experiments. The depth maps of the two scenes are shown in the first and third rows. To highlight the differences in depth completion, we generate a set of arbitrary views based on the produced depth image and the RGB image, which are presented in the second and fourth rows. In the last row, we zoom in on the rendered views and add pre-captured static background to fill the occluded regions.

The results from [1] are shown in the second column. As shown in the depth maps, this approach enlarges the foreground shape by attaching unrelated pixels around ob-

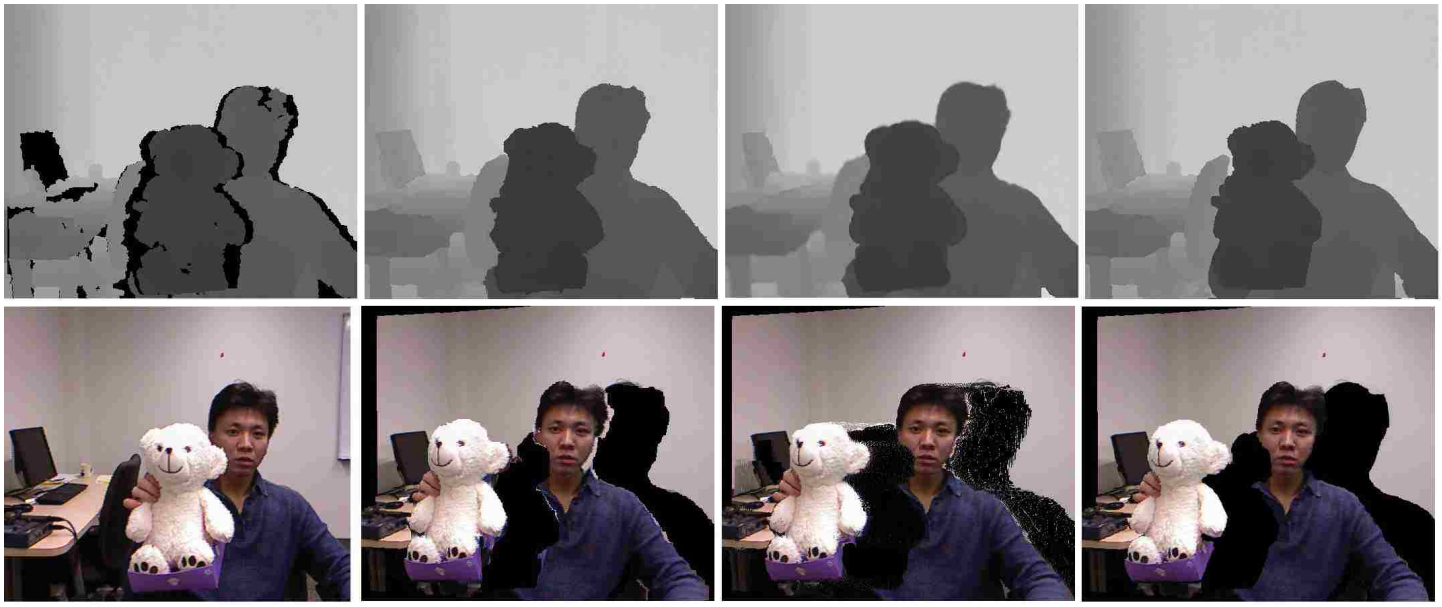


Figure 4.15: Comparison with other schemes. From left to right: input images together with our layer labeling, results from Camplani et al. [1], results from Diebel. et al. [3], and our results.

jects' boundaries. The wrongly assigned depth values move some of the background pixels to the foreground or vice versa. This error is clearly noticeable in the rendered virtual views – in the second row, a significant number of wall pixels present around the boundaries of the person's hair and arm. In the fourth row, the contours of the fingers are poorly inferred. Small gaps between fingers are naively wiped out due to erroneous color similarity or depth. Results from [3] in the third column have better contour than [1]. However, the MRF blurs the boundaries between foreground and background by generating intermediate depth values. From the virtual views, these intermediate depth values can be seen spreading across the space between the foreground and background. In contrast with these two schemes, our proposed method produces superior results. The depth completion steered by layers can better preserve the shape of object boundary and prevent noise propagation across objects

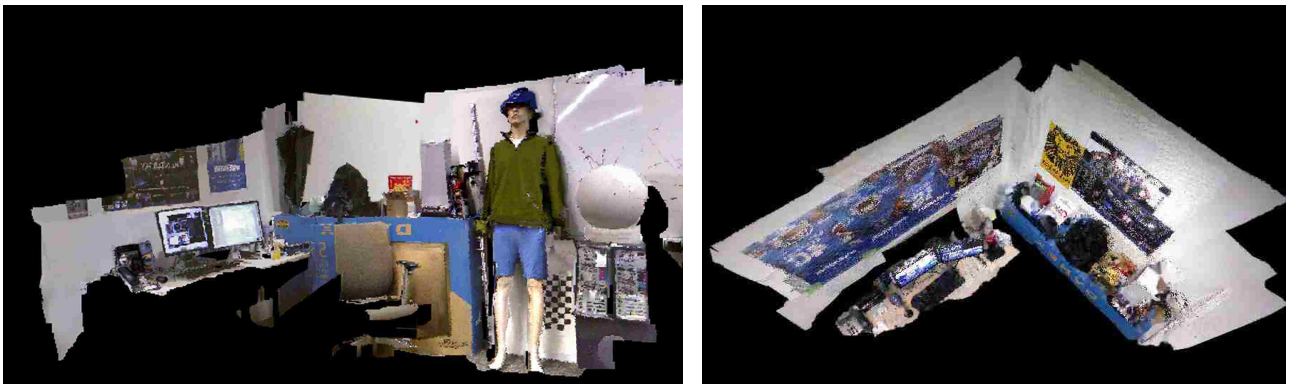
with significant depth differences.

Figure 4.16: Comparison with other schemes. From left to right: input images together with our layer labeling, results from Camplani et al. [1], results from Diebel. et al. [3], and our results.

4.6.2 Off-line Background Scanning

Figure 4.17 shows two examples of our off-line scanned backgrounds with a voxel size of $10 \times 10 \times 10$ mm. The indoor environment is scanned by a moving Kinect and is presented here as a 3D point cloud from an arbitrarily-placed virtual camera position. In terms of computation complexity, the only additional steps compared with [2] is the SIFT extraction and RANSAC matching, both of which can be run faster than

real time. Using the parallel computation strategy as described in [108], our system can achieve real-time performance.

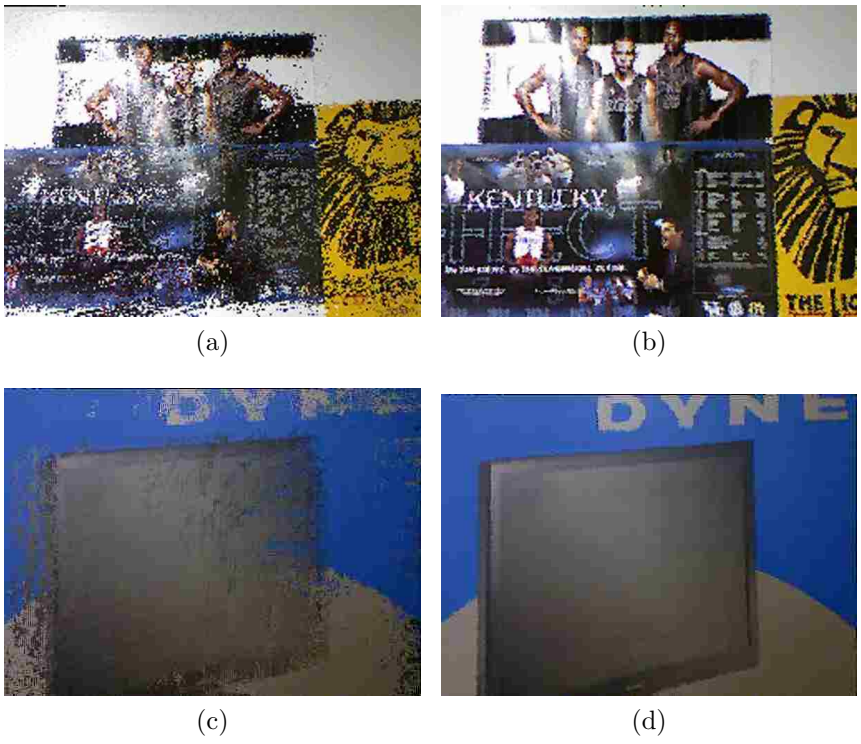


(a) (b)

Figure 4.17: Scanned background

To give a better analysis of the proposed method, we concentrate on the planar areas of the scanned environment and compare the reconstructed results with the ones by [2]. The reason for choosing [2] for comparison is that it represents a relatively popular approach that has been adopted by many other literatures [109, 110]. In Figure 4.18, the images are rendered by projecting the reconstructed 3D data to an arbitrary virtual view. Our results have significant improvements over the original scheme through a better preservation of the texture information on the planes. In particular, the text on the posters is clearly legible in the virtual views.

For quantitative evaluation, we compare the estimated camera poses and locations against the ground truth, which is manually measured. We first use the Kinect to scan the environment against a predefined path. Along the path, we pick 10 arbitrary positions and physically measure the relative translation T and rotation R of the cameras on each spot. According to the manual measurements, a sequence



(a) (b)

(c) (d)

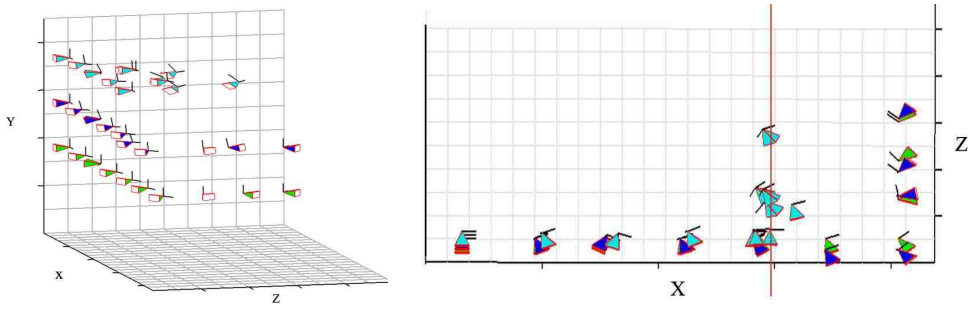
Figure 4.18: Virtual views comparisons: (a) and (c) are the results by [2]; (b) and (d) are the corresponding ones by our method.

of cameras are plotted in the 3D space as green cameras in Figure 4.19a. Based on the associated frames on the spots, two sequences of transformations are estimated respectively by our proposed method and [2]. Their results are shown in the same figure. For demonstration purpose, we arbitrarily raise the blue cameras (our result) and cyan cameras ([2]) along the y axis by a fixed distance.

Figure 4.19b provides the top view of the results: the scan starts from the left side along the x axis and ends in the z direction. The total path is about $3.5m$. For the first few camera positions, all the three results are aligned closely due to the corresponding part of the captured environment involving considerable depth variation. After the red boundary, the camera enters large plane regions (the indoor

Table 4.2: Error analysis on transformation estimation

	Category	error T	error R
By [2]	nonplanar	$0.012m$	$0.208°$
	planar	$0.731m$	$39.20°$
Ours	nonplanar	$0.009m$	$0.224°$
	planar	$0.024m$	$0.875°$



(a) (b)

Figure 4.19: Camera pose estimation results: the ground truth is physically measured as the green cameras shows; the cyan cameras and blue cameras respectively indicate the results by [2] and our method.

wall), which causes the estimated cyan cameras to mess up. In contrast, our results are not affected by the planes and remain close to the ground truth.

Table 4.2 summarizes the estimation errors: the translation error T and rotation error R are statistically computed in terms of the offsets from the ground truth when the camera is scanned with a movement of 1.0m. The analysis is conducted by two different occasions depending on whether the scanned environment has dominant plane surfaces.

4.6.3 Rendering accuracy of the virtual mirror

In order to validate the accuracy of our mirror model, we compare the generated virtual image I_1 with a real photo I_2 in Figure 4.20 taken by a digital camera looking

at a real mirror which is aligned with the display. The camera is in the same position as the asserted viewpoint that the mirror system uses for rendering. To compare the virtual mirror image with camera captured real mirror image, we project the captured mirror image on to the display plane by applying a homography using the corresponding four corners.

To measure the rendering accuracy, 100 corner points are manually selected on I_1 and I_2 for analysis. To ensure that the exact pixel locations of the corners are used, we compute the *Normalized Cross Correlation*(NCC) over a 3×3 neighborhood around each corner point and refine the corner position to that with the maximum local NCC value. All the matching pairs of corner points are shown in Figure 4.20. The position differences between the matching pair are indeed quite small – the mean is 1.4865 pixel with standard deviation 0.8156. The small position differences indicate that the rendering is sufficiently close to a real mirror.

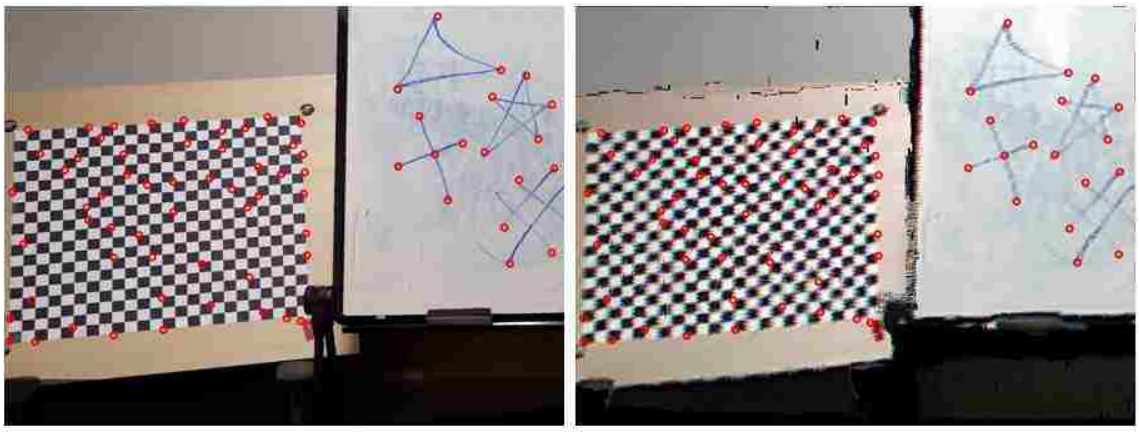


Figure 4.20: *Selected points on rectified real mirror image (left) and virtual mirror image (right)*

4.6.4 Quality Improvements on Mirror View Generation

To demonstrate the effect of adding depth denoising and background scanning to our mirror system, we first compare the original captured images (see Figures 4.21a to 4.21d) with the generated mirror views from the viewpoint of the subject without using depth denoising and background scanning (see Figures 4.21e-4.21h). The viewpoint rendering of the mirror views give the subject the illusion that the images are always captured directly from the front face. However, noticeable artifacts can be seen from those images. False outline of the person's hair and finger regions are visible near the top part of the images. These artifacts are caused by the erroneous depth values which map some of the foreground pixels to the background. Figures 4.21i to 4.21l show the improved results using our depth denoising and completion algorithm with all artifacts eliminated. Figures 4.21m to 4.21p use the same video sequences captured by the stationary RGB-D cameras but move the virtual mirror to different locations so that different parts of the background can be seen. Much of the background is beyond the field of view of the stationary RGB-D cameras and relies on the stored background model to fill in the missing details.

4.6.5 Virtual Mirror System

In this section, we present the results of our proposed virtual mirror system. Figure 5.1 shows the hardware setting. To fully capture the viewer and enhance the resolution of the generated view, we place three Kinects around the display where virtual mirror views are dynamically rendered. The results are demonstrated in Figure 4.22. Figures

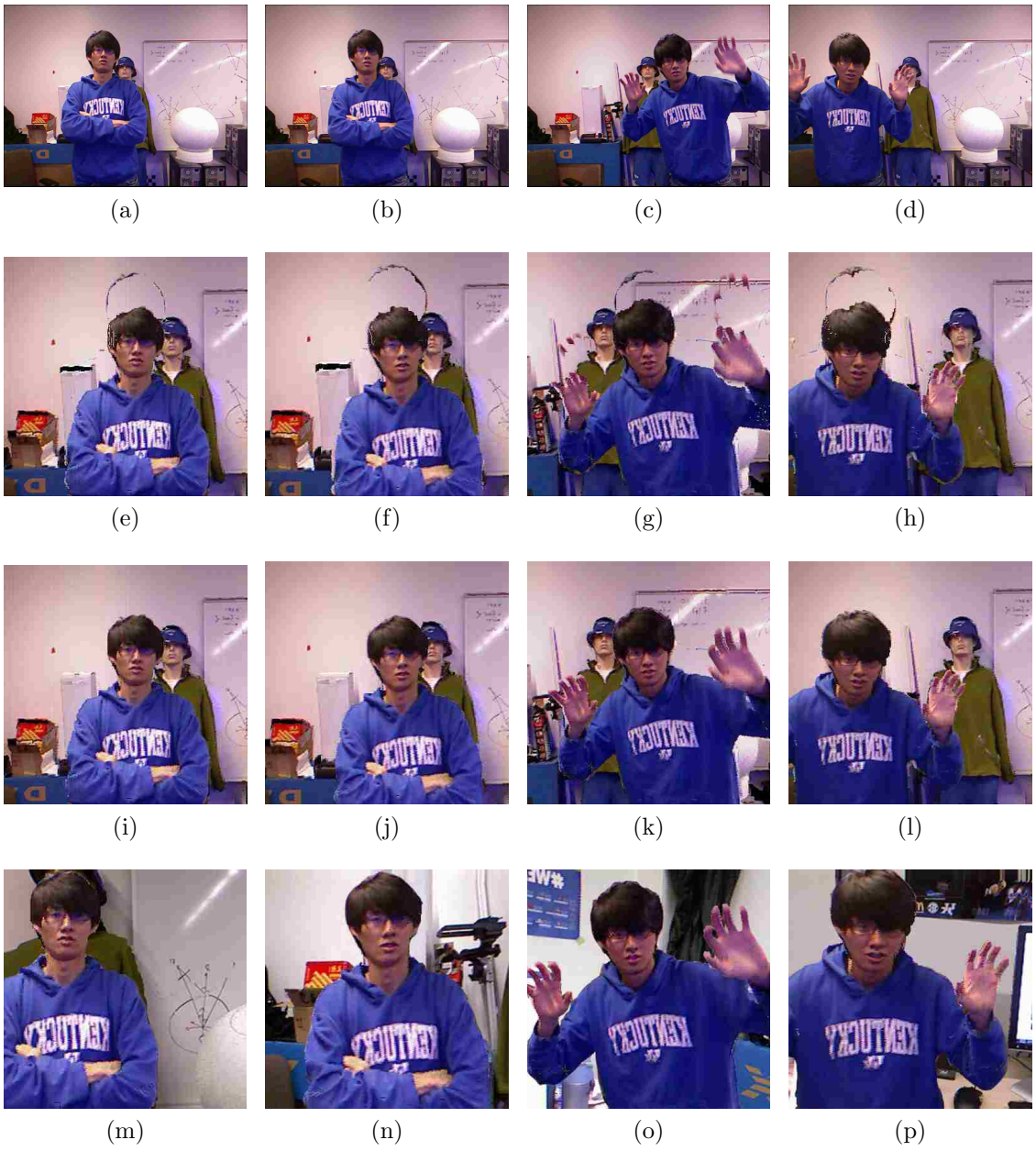


Figure 4.21: Mirror View Results I: the first row contains the original RGB frames; the second row contains the generated mirror view by adding static background with simple interpolation; the third row contains the improved mirror view by our method of depth denoising and completion; the fourth row is the generated mirror view with virtual background; Images from the same column correspond to the same frame in the captured video sequence.

4.22a to 4.22c show the input data captured by three Kinects mounted on top left, top right, and middle bottom of the display. Figure 4.22d is the synthesized frontal mirror view of the viewer. We first use our foreground/background separation scheme to extract the contour of the viewer from all the three input sources. Then the virtual camera projection model, as described in section 4.5, is applied to transform those side views into the front view. Since all the transformed views from the three inputs are aligned in the same coordinate system, we can simply merge them together in pixel domain. To ensure intensity consistency, we apply color transfer [111] on each view. In Figure 4.22e and Figure 4.22f, the person is placed in a different virtual environment, which is constructed off-line. Different 3D objects are inserted in the mirror views: in Figure 4.22e, the viewer can see himself virtually wearing a hat; in Figure 4.22f, the viewer hits a flying soccer ball with his head. In both scenarios, we use pose-tracking to automatically move the object to different locations as if the viewer is interacting with them. Figures 4.22g-4.22i demonstrate the virtual clothing effects by changing the T-shirt color and placing the viewer in different virtual environments.

4.6.6 Computational Performance

We have a preliminary prototype mirror system with two RGB-D cameras implemented using C++ and OpenCV library. Each Kinect captures 640×480 resolution video for scene points generation, and the local client renders the virtual image with resolution 1024×768, which is the same size as the final image on the display. The server and client machines are as follows:

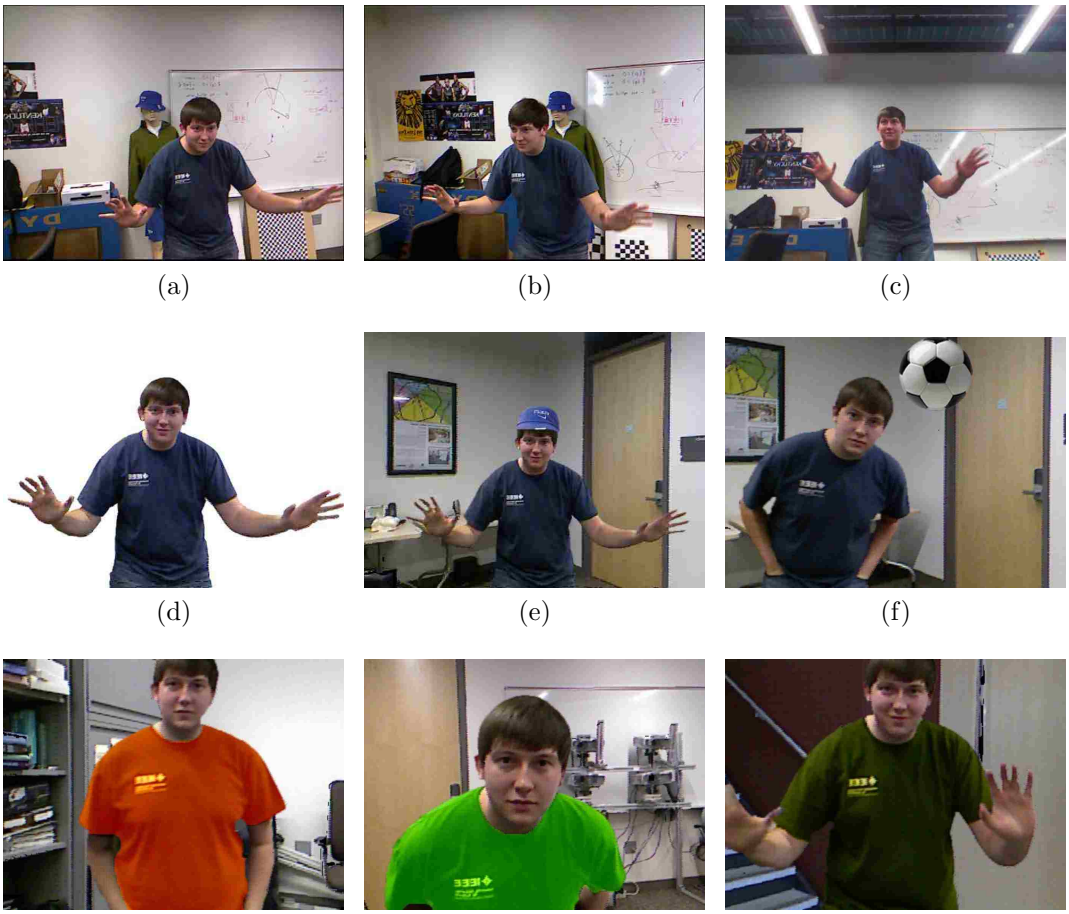


Figure 4.22: Mirror View Results II: the first row shows the original input frames from three Kinects: (a) left view, (b) right view, and (c) bottom view; (d) is the synthesized foreground view from the subject's viewpoint. (e)-(i) demonstrate different virtual mirror views by using different 3D background and inserting novel 3D objects.

- **Server** : Intel Xeon E5335 processor with 4-core CPUs at 2.0 GHz and 4.0Gb of RAM.

- **Client** : Intel Core(TM) E8400 Duo CPU at 3.00 GHz and 8.0Gb of RAM

The computation of the 2D and 3D transformations are accelerated by using optimized matrix operations from the OpenCV library. It takes approximately 12 ms

to complete all the transformation for a 320×240 frame. For the depth denoising algorithm, we take advantage of the multi-core architecture to compute the likelihood function $\phi(X_s)$ and edge potential $\psi(X_s, X_t)$ in four separate cores. This process takes 8 ms on average for each frame. The most computationally expensive part is the loopy belief propagation, which takes 23 ms to complete. Combining all the processes, our system can achieve real-time performance at 43 ms per frame or roughly 23 frames per second.

Chapter 5

3D Self Modeling

The use of a joint RGB-depth camera, like Kinect, allows us to capture the 3D geometry and color of the environment. This can provide flexibilities for a user to view self and behaviors dynamically from different perspectives. However, a camera often has a limited field of view, which constraints the scale of arbitrary views. In this chapter, I describe our calibration method for multiple cross-modal camera network to obtain globally aligned 3D point clouds.

Our work is inspired by the significant trend in recent years of using low-cost commodity depth cameras in different application fields including augmented reality, stereo vision, video surveillance, 3D reconstruction. Depth sensors such as the Time-of-Flight (TOF) and the Structured Light cameras can provide per-pixel depth value at video frame-rate. However, these depth sensors often have low resolution imaging and limited field of view. To resolve these limitations, multiple cameras are often used simultaneously to widen the capturing field of view and enhance the resolution. For example, it is common to rely on information obtained from companion RGB cameras to predict missing depth values [52][72]. Furthermore, applications like 3D reconstruction and surveillance require spatially disparate camera views to create watertight models for rendering, body pose tracking and understanding.

Figure 5.1: System setting for calibrating multiple color and depth camera network: the first row shows captured images from color cameras; the second row shows the ones from depth cameras.

As such, an accurate and robust calibration of multiple color and depth cameras is essential for applications that rely on wide-baseline networks with multiple color and depth cameras. However, it is challenging to calibrate a large network so that pixels from multiple disparate camera views can be correctly aligned into a unified coordinate system. There are three main challenges: first, the wide baseline means that the overlapping areas between camera views may be small or non-existent, thereby increasing the complexity of locating common features for calibration. Second, the capture data from color cameras and depth cameras are vastly different, and are distorted in their unique way: color cameras suffer from projective and lens distortion while depth cameras have missing and noisy measurement around depth discontinuities, transparent and reflective surfaces. Finally, multiple cameras are often cumbersome to setup and maintain. Thus, the calibration procedure should be

robust and easily adapted to changes in placement.

The main contribution of our method is a simple and robust framework for calibrating a network of multiple wide-baseline RGB and depth cameras that can produce more accurate results than other state-of-the-art techniques such as the work done by Herrera et al [68]. Our framework utilizes a spherical object for calibration and the procedure can be summarized as follows. First, we propose an effective sphere-fitting algorithm to identify the sphere centers in the RGB and depth images respectively. Second, the extrinsics are automatically obtained based on the detected corresponding sphere centers across different views. Two separate scenarios are considered in our framework: RGB-and-depth calibration and depth-and-depth calibration. An example of 3 RGB camera views and 3 depth camera views are shown in Figure 5.1.

5.1 Proposed Method

In our method, the correspondences are estimated based on the trajectory of the center of a moving sphere. There are two reasons for choosing a sphere as a calibration object. First, it is suitable for wide baseline: any part of the sphere can be used to estimate the location of its center. As such, two cameras capturing different sides of the sphere without any overlap in their fields-of-view can still use the sphere center as a correspondence. Second, instead of using error-prone small features as correspondences, we use the entire sphere to estimate the location of its center without using any a-priori depth noise model for denoising.

For any camera pair, the extrinsics are determined by a translation vector $\mathbf{t} = [t_x \;\; t_y \;\; t_z]^T$ between the two camera centers and a rotation matrix $\mathbf{R}$ parameterized

by the three rotation angles θ_x, θ_y and θ_z. With a set of estimated correspondences, an over-determined system of linear equations relating all the unknowns can be built. Solving this system gives us an initial guess of $\mathbf{R}$ and $\mathbf{t}$ between each camera pairs. After estimating all pairwise camera poses, a global alignment scheme is applied to refine the initial estimates so that they are consistent across the entire network.

Intrinsics parameters of all cameras can be easily obtained using existing calibration toolbox. To detect the sphere-center correspondences and estimate the extrinsics, different strategies for the RGB cameras and depth cameras are needed. In the following subsections, we describe each case in detail.

5.1.1 Sphere Detection in Depth Image

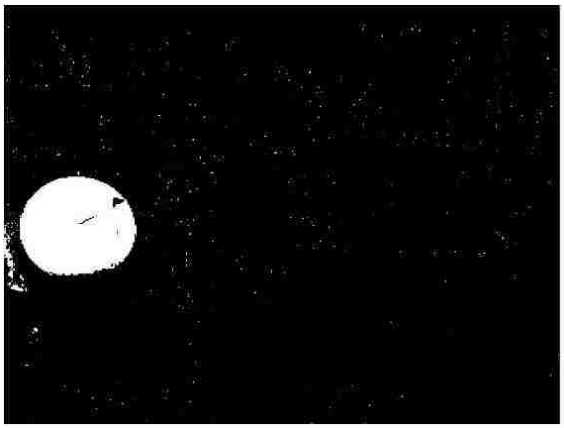
In this section, we describe the algorithm to identify depth pixels corresponding to the sphere and estimate the sphere center. First, all the depth measurements are converted back to a local 3D coordinate system using the inverse camera projection parameterized by the intrinsic parameters. Since the moving sphere belongs to foreground, we first limit our search to foreground point-clouds which are identified as those that are significantly different from a pre-captured static background. The foreground pixels may include objects other than the sphere. To search for the sphere, we coarsely partition all the foreground pixels into equal-size rectangular blocks so that point clouds within each block are spatially close. For each block, we apply a *RANSAC* procedure to iteratively identify all the 3D points that satisfy the surface equation of a 3D sphere with radius matching that of the calibration object. We use the direct least-square sphere fitting technique as described in [112]. Finally, we take

the union of identified points over all blocks and fit a spherical equation over them. The coordinates of the center can then be computed using the best-fit equation. Repeating the same procedure for each input frame yields the trajectory of estimated sphere centers in the 3D space: $\{C_0, C_1, ..., C_n\}$, where C_i represents the 3D coordinate of an estimated sphere center and the subscript i indicates the frame index in the video sequence.

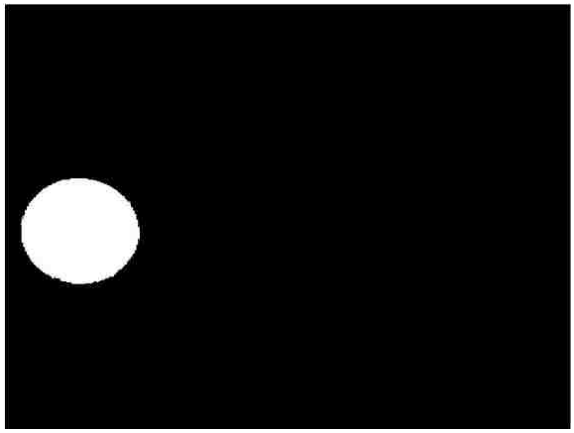
5.1.2 Sphere Detection in Color Images

For each input color frame, we first identify the foreground using the background subtraction algorithm in [113]. Due to perspective distortion, the geometry of the spherical calibration object alone is not enough for identification. Instead, we have painted the object with a distinctive color that is not present in the background. Based on the color distribution during off-line training, a rough sphere contour can be approximated from the foreground mask. To accurately compute the sphere center, we follow an approach similar to that in [114], which is based on the relationship between the image of the sphere $\mathcal{C}$ and the IAC or Image of the Absolute Conic ω. The IAC ω can be calculated using pole-polar relation with orthogonal constraints [60]: $l = \omega \cdot v$, where l and v are the polar and pole respectively. Let the camera intrinsic matrix be K, the IAC is then given by $\omega = K^{-T} \cdot K^{-1}$ [115]. In the dual space, the relation between $\mathcal{C}^*$ and ω^* can be expressed as $\mathcal{C}^* = K \cdot K^T - c \cdot c^T$ and $\omega^* = K \cdot K^T$ where c is the image of the sphere center. For any three frames from the captured video sequence, we first fit the boundaries of all the three spheres (one from each frame) by using the least-square approach [116]. The corresponding conic homography matrices

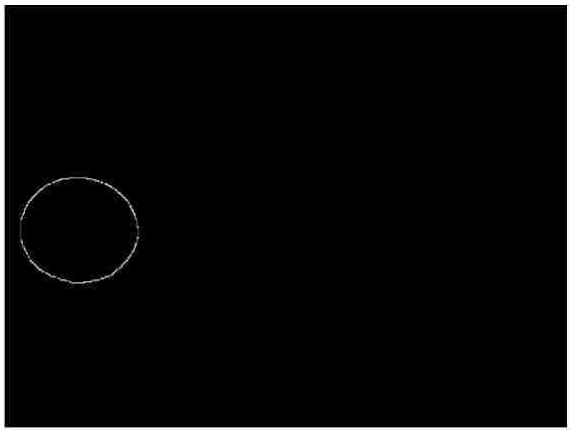
can be computed as: $H_{c_{12}} = \mathcal{C}_2 \cdot \mathcal{C}_1^*$, $H_{c_{32}} = \mathcal{C}_2 \cdot \mathcal{C}_2^*$, and $H_{c_{31}} = \mathcal{C}_1 \cdot \mathcal{C}_3^*$. For each of these three conic homography matrices, its eigenvector and eigenvalue are computed. We then choose the three eigenvectors that intersect their corresponding sphere image pairs to obtain three polars l_{12}, l_{23}, and l_{31}. The image of the sphere centers c_1, c_2, and c_3 from the three frames can then be computed as the intersections between the three polars. By following the steps above, for a sequence of frames, we can obtained the images of the sphere centers in 2D coordinate $\{c_1, c_2, ..., c_n\}$. Figure 5.2 shows the sphere detection results from the color cameras.



(a) Background subtraction



(b) Foreground mask



(c) Canny edge detection

(d) Final ellipse fitting result

Figure 5.2: Sphere detection from color camera captured images

5.1.3 Calibration between two depth cameras

In this section, we describe how we compute the extrinsics between two depth cameras. After the trajectory of the sphere center is detected from each depth camera, we can use them as correspondences to estimate the rotation matrix $\mathbf{R}$ and translation $\mathbf{t}$ between the coordinate systems of the two cameras. For the two depth cameras p and q, we denote the two trajectories as $\{C_0^{(p)}, C_1^{(p)}, ...C_n^{(p)}\}$ and $\{C_0^{(q)}, C_1^{(q)}, ..., C_n^{(q)}\}$. Time synchronization is assumed between the two cameras but some of these measurements could be empty as the sphere may not be visible in those instances. Thus, the first step is to go over the two data sequence and filter out those frames that have no sphere detected. To compute the unknown parameters, we also require at least $m > 6$ correspondences. Our goal is to find $\mathbf{R}^{(pq)}$ and $\mathbf{t}^{(pq)}$ that satisfy:

$$\begin{aligned} &[C_{s_0}^{(q)} \;\; C_{s_1}^{(q)} \;\; \cdots \;\; C_{s_m}^{(q)}]^T \\ &= \;\; \mathbf{R}^{(pq)} \cdot [C_{s_0}^{(p)} \;\; C_{s_1}^{(p)} \;\; \cdots \;\; C_{s_m}^{(p)}]^T + \mathbf{t}^{(pq)} \end{aligned} \tag{5.1}$$

Due to the orthogonality constraint on the rotation matrix $R^{(pq)}$, the usual SVD approach does not necessarily produce the optimal solution. As such, we use the least-square based method for determining rigid body transformation in [117] by first computing the covariance matrix as follows:

$$A = \sum_{i=1}^{m} [(C_{s_i}^{(p)} - \bar{C}^{(p)}) \cdot (C_{s_i}^{(q)} - \bar{C}^{(q)})^T] \tag{5.2}$$

where $\bar{C}^{(p)} = \frac{1}{m} \cdot \sum_{i=1}^{m} C_{s_i}^{(p)}$ and $\bar{C}^{(q)} = \frac{1}{m} \cdot \sum_{i=1}^{m} C_{s_i}^{(q)}$ are the respective centroids of the two correspondence sets. Then, we can compute SVD $A = USV^T$ and obtain

$\mathbf{R}^{(pq)} = VU^T$ and $\mathbf{t}^{(pq)} = \bar{C}^{(q)} - \bar{C}^{(p)}$. The output of this step is the set of extrinsics between all adjacent pairs of depth cameras.

5.1.4 Calibration between depth and color cameras

As only the images of the sphere center can be estimated from a color camera, we cannot apply the same approach in finding the extrinsics between a color camera and a depth camera. An alternate constraint is required. Consider a pair of correspondence between the 3D sphere center C_i from the depth camera and its image c_i on the color camera. Let the extrinsics that map the depth camera to the color camera be $\mathbf{R}$ and $\mathbf{t}$. Using the coordinate system at the color camera, the ray emanating from the optical center of the color camera to c_i is $[c_i \;\; f_c]^T$ where f_c is the focal length. This ray is caused by the "light" reflected by the sphere center to the optical center, and thus it must be parallel to $\mathbf{R} \cdot C_i + \mathbf{t}$. This constraint can be captured in a cross product equation as follows:

$$(\mathbf{R} \cdot C_i + \mathbf{t}) \times [c_i \;\; f_c]^T = 0 \tag{5.3}$$

which holds for all correspondences $i = 1, 2, \ldots, n$. Applying SVD on these equation results in an optimal solution in terms of $\mathbf{Rt}$ from the decomposed matrices $[U, S, V]$. To ensure the orthogonality constraint on the rotation matrix $\mathbf{R}$, we can apply the essential matrix $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$ to further extract $\mathbf{R}$ and $\mathbf{t}$.

5.1.5 Global Alignment

In this stage, we refine all the parameters estimated from previous steps to produce jointly optimal viewing parameters, including intrinsics and extrinsics for all the

cameras, and generate a unified 3D point cloud. Our global optimization algorithm is derived from the idea of Bundle Adjustment (BA) [118], which is essentially a parameter estimation problem using non-linear model and has been invariably used as the last step for many 3D reconstruction applications. The goal of bundle adjustment is to adjust the camera parameters that minimize the overall projection error between the 3D points and the 2D correspondences. Mathematically, the original format of BA can be expressed as the equation below [118]:

$$\min_{\mathbf{P}_j, \tilde{\mathbf{X}}_i} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} d(f(\mathbf{P}_j, \tilde{\mathbf{X}}_i), \tilde{\mathbf{x}}_{ij})^2 \tag{5.4}$$

where i and j represent the indices of points and cameras. m and n are the total number of cameras and points. The function f denotes the relation that maps 3D point $\tilde{\mathbf{X}}_i$ to 2D pixel $\tilde{\mathbf{x}}_{ij}$ by the corresponding projection matrix $\mathbf{P}_j$. The variable $v_{ij} \in \{0, 1\}$ indicates whether the point is visible by camera j. The function d denotes the square Euclidean distance.

For our problem, we have a slightly different version from the original bundle adjustment due to the special data types. As mentioned earlier, the input for this stage is:

1. m_1 sequences of 2D sphere center points from m_1 color cameras $\{\mathbf{c}_{c_1}, \mathbf{c}_{c_2} ..., \mathbf{c}_{c_{m_1}}\}$. Each $\mathbf{c}_{c_j}$ is a $3 \times n$ matrix, representing n 2D points in homogeneous coordinate detected from a color camera c_j.

2. m_2 sequences of 3D sphere center trajectories from m_2 depth cameras $\{\mathbf{C}_{d_1}, \mathbf{C}_{d_2} ..., \mathbf{C}_{d_{m_1}}\}$. Each $\mathbf{C}_{d_j}$ is a $4 \times n$ matrix, representing n 3D points in homogenous coordinate

extracted from a depth camera d_j.

3. Initial estimated intrinsic parameters for each camera: $\{\mathbf{K}_{c_1}, ..., \mathbf{K}_{c_{m_1}}, \mathbf{K}_{d_1}, ..., \mathbf{K}_{d_{m_2}}\}$.

4. extrinsic parameters based on rotation matrices $\{\mathbf{R}_{pq}\}$ and translation vectors $\{\mathbf{t}_{pq}\}$, where p and q are any two different camera indices.

To register all the cameras globally, a referenced position is chosen as the world coordinate origin. Based on the input 3) and 4), we can map the extracted 3D sphere center trajectories to the world system. Meanwhile, for each camera, we can obtain an initial camera projection according to the equation $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$. So our bundle adjustment equation can be developed as follows:

$$\min_{\mathbf{P},\bar{\mathbf{C}}} \sum_{j=1}^{m_1} w_{c_j} d(f_1(\mathbf{P}_{c_j}, \bar{\mathbf{C}}), \mathbf{c}_{c_j})^2 + \sum_{j=1}^{m_2} w_{d_j} d(f_2(\mathbf{P}_{d_j}, \bar{\mathbf{C}}), \mathbf{C}_{d_j})^2 \tag{5.5}$$

There are two components in equation (5.5) with the first part for the color metric and the second part for the depth metric. Function f_1 is to map a 3D point to a 2D pixel on the color image: $\hat{\mathbf{c}}_{c_j} = \mathbf{P}_{c_j}\bar{\mathbf{C}}$, where $\hat{\mathbf{c}}_{c_j}$ represents the projection 2D point. The variable $\bar{\mathbf{C}}$ is a $4 \times n$ matrix that stores the 3D coordinate of the sphere center trajectory in the world coordinate. As an initial input, we simply transfer all the extracted 3D sphere center trajectories from each local camera to the world coordinate and take the average of them. For the function f_2, different from f_1, it simply transforms $\bar{\mathbf{C}}$ from the world coordinate back to the camera coordinate. Since the $\mathbf{R}_{d_j}$ and $\mathbf{t}_{d_j}$ can be extracted from $\mathbf{P}_{d_j}$, we can obtain the transformed 3D points: $\hat{\mathbf{C}}_{d_j} = \mathbf{R}_{d_j}^{-1}\bar{\mathbf{C}} - \mathbf{t}_{d_j}$. Since the error values are measured differently for the color and depth data, we use the weight parameters w_{c_j} and w_{d_j} to control their contributions.

To find the best solution of $\{\mathbf{P}_{c_1}, ..., \mathbf{P}_{c_{m_1}}, \mathbf{P}_{d_1}, ..., \mathbf{P}_{d_{m_2}}\}$ and $\bar{\mathbf{C}}$, we need to find the minimum value of the nonlinear least-square equation (5.5). The most standard technique for solving nonlinear least-square problem is Levenberg-Marquardt (LM) algorithm [119], which uses an iterative procedure to find a local minimum of a multivariate function. LM can be considered as a combination of a steepest descent method and the Gauss-Newton method. When the current configuration is far away from a local minimum, the algorithm works like a steepest decent method which is slow but always converges. Otherwise, when the current configuration is close to a local minimum, LM has similar behavior as the Gauss-Newton method. For our problem, from equation (5.5), we define a parameter vector $\vec{\mathbf{P}}$ which can be extracted from the projection matrix set $\{\mathbf{P}_{c_1}, ..., \mathbf{P}_{c_{m_1}}, \mathbf{P}_{d_1}, ..., \mathbf{P}_{d_{m_2}}\}$ by stretching each individual matrix in row-wise and concatenating them together. We denote this mapping function as ψ:

$$\begin{aligned}\vec{\mathbf{P}} &= \psi(\mathbf{P}_{c_1}, ..., \mathbf{P}_{c_{m_1}}, \mathbf{P}_{d_1}, ..., \mathbf{P}_{d_{m_2}}) \\ &= [\mathbf{P}^{00}_{c_1}, \mathbf{P}^{01}_{c_1}, \mathbf{P}^{02}_{c_1}, ..., \mathbf{P}^{31}_{c_{m_1}}, \mathbf{P}^{32}_{c_{m_1}}, \mathbf{P}^{33}_{c_{m_1}}, ..., \mathbf{P}^{31}_{d_{m_2}}, \mathbf{P}^{32}_{d_{m_2}}, \mathbf{P}^{33}_{d_{m_2}}]\end{aligned} \tag{5.6}$$

where the superscript indicate the matrix element index. Now we can consider equation (5.5) as a new function $\dot{f}$ parameterizing on the vector $\vec{\mathbf{P}}$ that produces an estimated measurement vector $\vec{\mathbf{C}}$:

$$\vec{\mathbf{C}} = \psi([\hat{\mathbf{c}}_{c_1}, ..., \hat{\mathbf{c}}_{c_{m_1}}, \hat{\mathbf{C}}_{d_1}, ..., \hat{\mathbf{C}}_{d_{m_2}}]) = \dot{f}(\vec{\mathbf{P}}) \tag{5.7}$$

Our goal is to find the desired configuration for the vectors $\vec{\mathbf{P}}$ and $\bar{\mathbf{C}}$ that minimize the squared distance error:

$$\epsilon^T \epsilon = \min_{\mathbf{P},\vec{\mathbf{C}}} (\dot{f}(\vec{\mathbf{P}}) - \psi(\bar{\mathbf{C}}))^T (\dot{f}(\vec{\mathbf{P}}) - \psi(\bar{\mathbf{C}})) \tag{5.8}$$

As there are two variables $\vec{\mathbf{P}}$ and $\bar{\mathbf{C}}$ involved in this minimization problem, our strategy is: during the iterations, these two variables are refined alternately. When update the value for $\vec{\mathbf{P}}$, we consider $\bar{\mathbf{C}}$ as a constant variable. Then the updated $\vec{\mathbf{P}}$ is used to re-compute the value of $\bar{\mathbf{C}}$.

The basic principle of LM algorithm is to apply affine approximation to the function $\dot{f}$ by finding an appropriate step value $\Delta_{\vec{\mathbf{P}}}$ around the current $\vec{\mathbf{P}}$, such that

$$\dot{f}(\vec{\mathbf{P}} + \Delta_{\vec{\mathbf{P}}}) \approx \dot{f}(\vec{\mathbf{P}}) + \mathbf{J}\Delta_{\vec{\mathbf{P}}} \tag{5.9}$$

where $\mathbf{J}$ is the Jacobian matrix of function $\dot{f}$. At each iteration, we need to find a proper step value $\Delta_{\vec{\mathbf{P}}}$ that minimize $||\dot{f}(\mathbf{P}) - \psi(\bar{\mathbf{C}})|| \approx ||\dot{f}(\vec{\mathbf{P}}) + \mathbf{J}\Delta_{\vec{\mathbf{P}}} - \psi(\bar{\mathbf{C}})|| = ||\mathbf{J}\Delta_{\vec{\mathbf{P}}} - \epsilon||$. According to the normal equation principle [120], the minimum is yielded when $\mathbf{J}\Delta_{\vec{\mathbf{P}}} - \epsilon$ is orthogonal to the column space of $\mathbf{J}$:

$$(\mathbf{J}\Delta_{\vec{\mathbf{P}}} - \epsilon)\mathbf{J}^T = 0 \Longrightarrow \mathbf{J}^T\mathbf{J}\Delta_{\vec{\mathbf{P}}} = \mathbf{J}^T\epsilon \tag{5.10}$$

So by solving the equation (5.10), we can identify the step value $\Delta_{\vec{\mathbf{P}}}$. In fact, for the original Bundle algorithm, it is has a slight different version than (5.10) by introducing *damping term* μ: $(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I})\Delta_{\vec{\mathbf{P}}} = \mathbf{J}^T\epsilon$. Users can refer [121] for further details. By iteratively solving this equation, the step value $\Delta_{\vec{\mathbf{P}}}$ can be computed. The steps of finding an optimal camera parameter vector $\mathbf{P}^+$ and a 3D point vector $\bar{\mathbf{C}}$ is summarized in algorithm-2.

Data: Estimated Camera Projection Matrices $\{\mathbf{P}_{c_1}, ..., \mathbf{P}_{c_{m_1}}, \mathbf{P}_{d_1}, ..., \mathbf{P}_{d_{m_2}}\}$
3D sphere center trajectory matrix $\bar{\mathbf{C}}$
Result: Optimized Camera Projection Matrices $\{\mathbf{P}^*_{c_1}, ..., \mathbf{P}^*_{c_{m_1}}, \mathbf{P}^*_{d_1}, ..., \mathbf{P}^*_{d_{m_2}}\}$
and $\bar{\mathbf{C}}^*$ minimizing $||\dot{f}(\mathbf{P}) - \psi(\bar{\mathbf{C}})||$
initialization;
define minimum square error $\hat{\epsilon}$;
$\vec{\mathbf{P}} = \psi(\mathbf{P}_{c_1}, ..., \mathbf{P}_{c_{m_1}}, \mathbf{P}_{d_1}, ..., \mathbf{P}_{d_{m_2}})$;
$\epsilon = ||\dot{f}(\vec{\mathbf{P}}) - \psi(\bar{\mathbf{C}})||$;
while *epsilon* $> \hat{\epsilon}$ **do**
  Solve $(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I})\Delta_{\vec{\mathbf{P}}} = \mathbf{J}^T\epsilon$;
  if $\Delta_{\vec{\mathbf{P}}} < \hat{\epsilon}(\hat{\epsilon} + ||\vec{\mathbf{P}}||)$ **then**
    break;
  else
    $\vec{\mathbf{P}} = \vec{\mathbf{P}} + \Delta_{\vec{\mathbf{P}}}$;
    update μ; update $\bar{\mathbf{C}}$ by $\vec{\mathbf{P}}$;
  end
end
$\{\mathbf{P}^*_{c_1}, ..., \mathbf{P}^*_{c_{m_1}}, \mathbf{P}^*_{d_1}, ..., \mathbf{P}^*_{d_{m_2}}\} = \psi^{-1}(\vec{\mathbf{P}})$; $\bar{\mathbf{C}}^* = \bar{\mathbf{C}}$

Algorithm 2: Global Optimization Algorithm on Camera Parameters

5.2 Experiments

The Microsoft Kinect sensors, which have both RGB and structure-light depth cameras, are used in our experiments. The experiment is conducted in an indoor environment with the size of $9.6m \times 7.2m$. Three Kinect cameras are mounted around the capturing space. Each Kinect is connected to a separate computer. To ensure time synchronization, we set up a local NTP time server to synchronize all computers. We install a GPS Board at the time server which can output a precise PPS (pulse per second) signal. After synchronizing with the local time server, offset of system time among all computers are within 4 ms. Our system can achieve near realtime performance on the captured 640×480 resolution video running on the computers with the hardware setting: Intel Core(TM) E8400 Duo CPU at 3.00 GHz and 8.0Gb

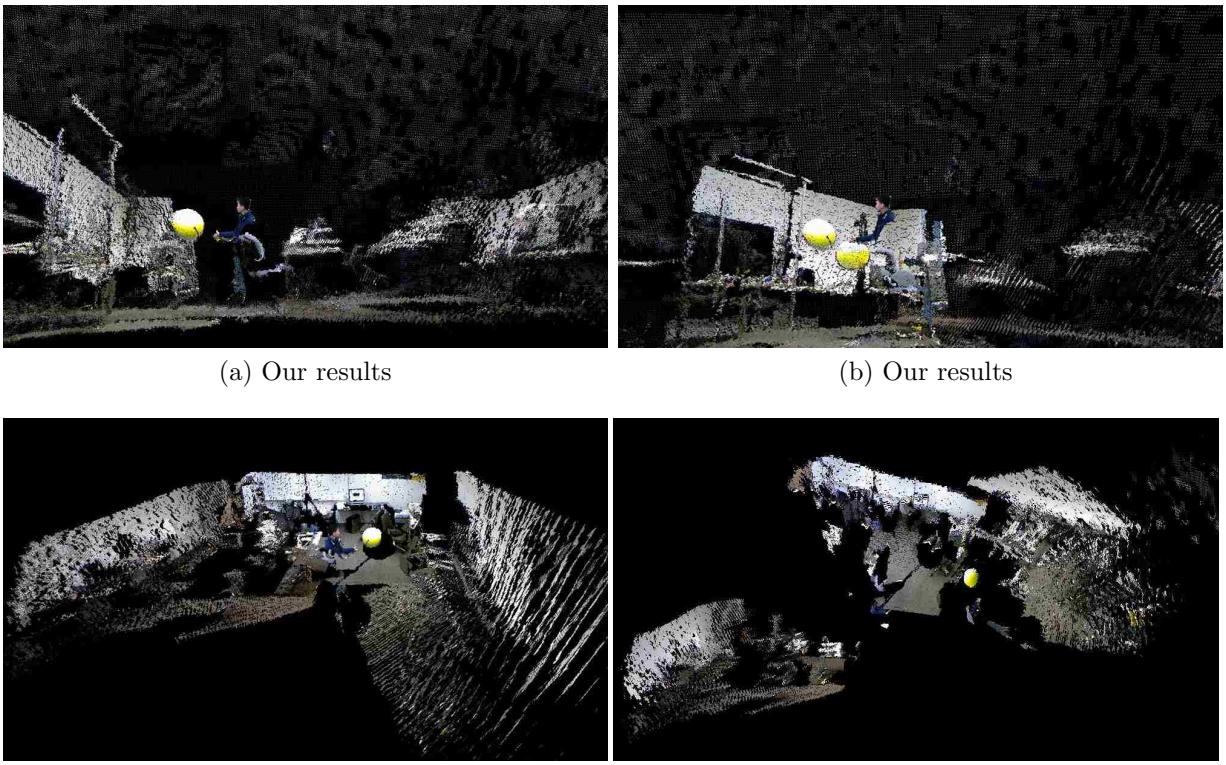
of RAM.

5.2.1 Surface Mesh

Our surface mesh generation is based on [122]. The surface are reconstructed by taking into account that the point cloud is organized in rows and columns, so adjacent pixels' points will be connected by triangles. Since a depth image's size is 640*480, each pixel $P(x, y)$ in a depth image can generate triangle $T_A = [P(x, y), P(x, y + 1), P(x + 640, y)]$ and $T_B = [P(x, y + 1), P(x + 640, y), P(x + 640, y + 1)]$. These generated triangles are back-projected to our global space. A threshold T_s is used to check length of triangle sides. If it is too large, the triangle will be discarded in rendering process. The surface mesh usually have overlapping problems due to triangles are generated from multiple camera, and it results in a nonuniform surface. The overlapping triangles can be removed by projecting triangles generated by Camera C_i to adjacent Cameras C_{i-1} and C_{i+1}, such that the triangle from C_i which is located on image space of C_{i-1} and C_{i+1} will not be rendered. Finally, we utilize the method of re-triangulation to recover the boundary regions between two adjacent cameras. A unified surface mesh created by multiple cameras is formed.

5.2.2 Qualitative Evaluation

We compare the performance of our system with the recent work in [68]. During the calibration procedure, a checkerboard and a sphere are used separately to capture the data as the input for [68] and our framework. In Figure 5.3, it shows the reconstructed 3D views of the indoor environment that are captured by three pairs of color and depth

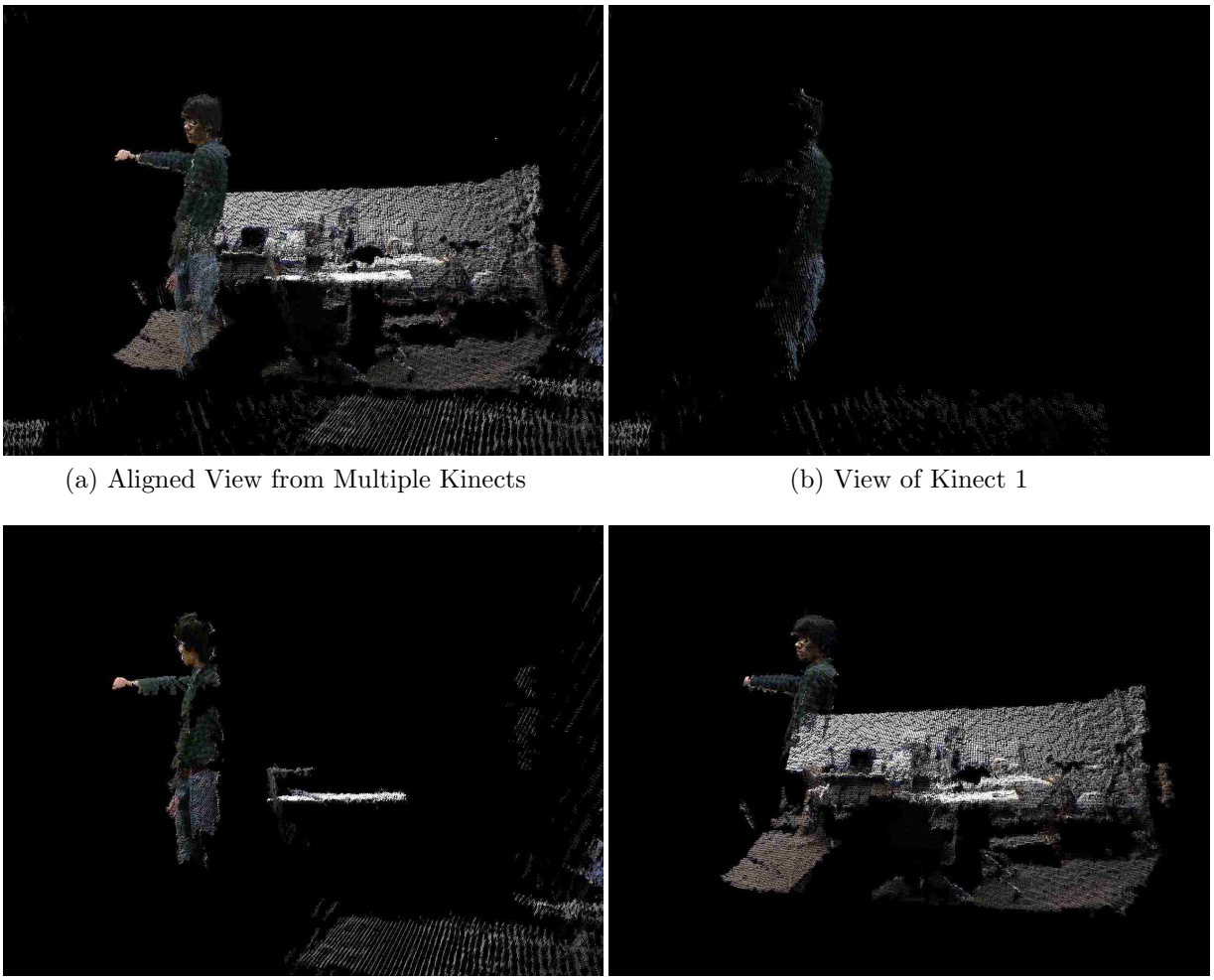


(a) Our results (b) Our results

(c) Our results (d) Herrera et al. 2012 [68]

Figure 5.3: Scene reconstruction results by using the estimated extrinsics.

cameras. Our results (shown on the left) achieve better accuracy in aligning multiple views. By contrast, the method in [68] does not work well for the wide baseline environment. One reason is that as the baseline gets wider, the checkerboard has to be placed far away from each camera to make sure it is captured in multiple views. This affects the captured image resolution and the correspondence matching between different views. In figure 5.4,5.5, we show aligned view from multiple Kinects that are registered via our estimated extrinsics. The results show satisfactory alignment of the person's face and body.



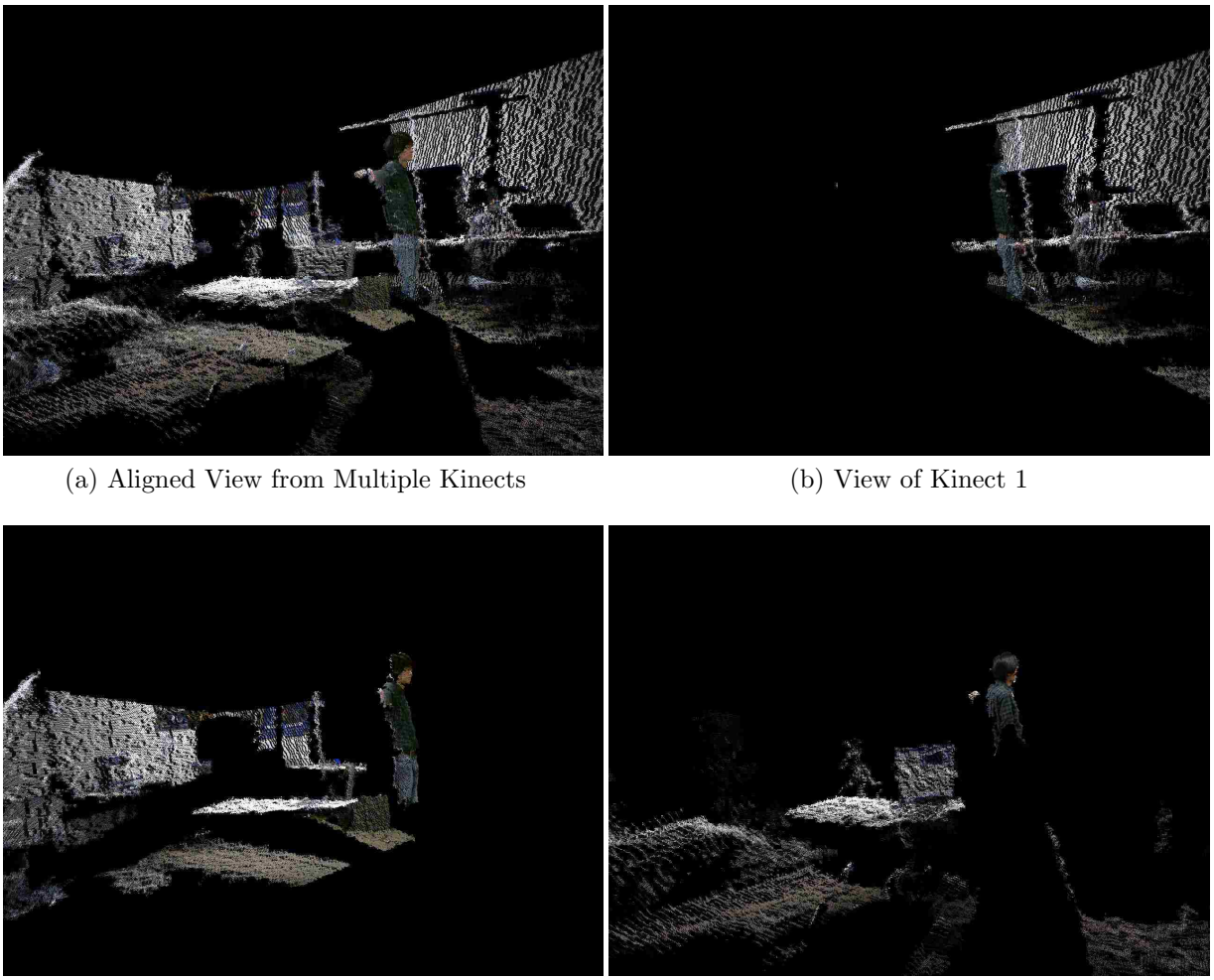
(a) Aligned View from Multiple Kinects (b) View of Kinect 1

(c) View of Kinect 2 (d) View of Kinect 3

Figure 5.4: Example 1: multi-views and each individual views from different cameras

5.2.3 Quantitative Evaluation

For the quantitative analysis, we evaluate the two calibration components in our framework: RGB-and-depth and depth-and-depth calibration, by comparing the estimated extrinsics against the ground truth, which is measured manually. For the RGB-and-depth calibration, we use one Kinect's RGB camera to align with another Kinect's depth camera. 18 pairs of color and depth images are captured from the two cameras. For the translation vector $\mathbf{t}$, the mean error along x, y, and z-axes is



(a) Aligned View from Multiple Kinects

(b) View of Kinect 1

(c) View of Kinect 2

(d) View of Kinect 3

Figure 5.5: Example 1: multi-views and each individual views from different cameras

3.374 cm with standard deviation 1.934 cm. For the rotation angles θ_x, θ_y, and θ_z, the mean error is 2.661° with standard deviation 1.157°. After aligning the color and depth image, the average offset between the corresponding sphere centers is 4.325 pixels.

For the depth-and-depth calibration evaluation, we compare our results with the ones by [68] against the ground truth. We place two kinects with a distance ranging from $0.3m$ to $3m$, and with the angles from 0° to 90°. In table 5.1, it shows the

Table 5.1: Quantitative error analysis of depth-and-depth calibration. The translation t is in centimeters. To save the table space, the decimal part is removed.

Ground Truth	Ours	by [68]
t_1(30, 0, 0) $R_1(0°, 0°, 0°)$	t_1(28, 2, 9) $R_1(3°, 2°, 2°)$	t_1(33, 13, 1) $R_1(2°, -2°, -1°)$
t_2(135, 0, 120) $R_2(0°, 60°, 0°)$	t_2(134, -2, 123) $R_2(-2°, 62°, 3°)$	t_2(187, 13, 93) $R_2(23°, 47°, -9°)$
t_3(269, 0, 135) $R_3(0°, 90°, 0°)$	t_3(265, -3, 132) $R_3(1°, 88°, 0°)$	t_3(209, 8, 267) $R_3(-34°, 68°, 26°)$

calibration results based on three different camera settings. We can find as the baseline gets wider, the calibration error by [68] increases, while our method maintains relatively consistent accuracy. Figure 5.6 demonstrates the aligned sphere paths that are captured by three Kinects. By computing the distances between the aligned correspondences, the root MSE can be reduced to no more than 2 cm.

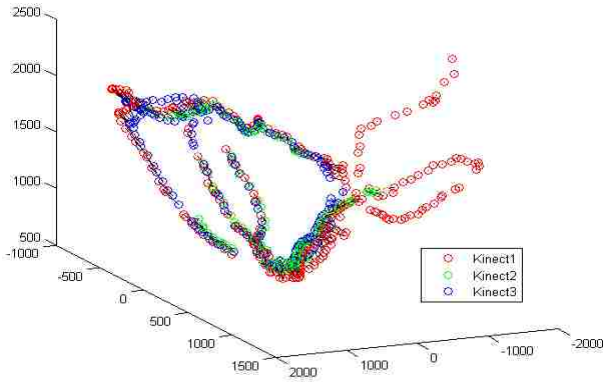


Figure 5.6: Aligned sphere movement trajectories by using our estimated extrinsics

Chapter 6

Conclusions and Future directions

In this dissertation, I have developed novel multimedia processing algorithms and demonstrated the use of computational multimedia techniques in automatically generating video material for video self modeling intervention. The advantage of computational techniques lies in its flexibility in creating unseen behaviors.

Two VSM based systems have been developed and evaluated. For the SpeakToMe device, it is designed specifically for voice therapy and produces a video with the patient's coarse voice replaced by a healthy voice. Experimental results have shown that natural human voice selected through speaker similarity provides the best subjective results. No additional benefit has been found by using voice conversion techniques due to the inaccurate target models created with the coarse voice. Optimal alignment between the original and replacement speech has been accomplished through a combination of automatic audio segmentation and lip-state extraction. Based on the alignment, an adaptive re-sampling algorithm has been proposed to preserve the motion energy during the lip-synchronization process. Extensive objective and subjective evaluations have demonstrated the advantages of our design and a clinical test is currently underway to study the effectiveness of our system in a larger scale.

For the Magic mirror system, we have presented a framework for rendering vir-

tual mirror by fusing multiple color-and-depth cameras. Our system can be easily implemented with modern PCs and commodity RGB-D cameras such as the Microsoft Kinect. The initial depth data has been enhanced using a depth denoising and completion algorithm which takes advantage of a novel probabilistic background/foreground separation to eliminate outliers and complete missing values. To support a large mirror surface and wide viewing angle, an off-line background scanning is used to capture the background environment. Dynamic RGB-D data captured by each client is used to estimate the viewpoint and create a 3D point cloud to render a viewer-dependent mirror image. The server aggregates all the partially rendered mirror images to compute the final result. Experimental results have demonstrated its accuracy and consistency with respect to a real mirror. Our current implementation can achieve the real-time performance for low resolution frames. We are currently exploring GPU implementation and algorithmic speedup on the multiple-round belief propagation in depth denoising.

While our proposed systems are domain specific, we believe that the concept of using multimedia techniques for video self modeling has far-reaching importance in many different areas of health care and behavioral intervention. My future research plan will continually focus on the Magic Mirror System by developing it to a more practical stage and further exploring its therapeutic values for VSM.

1. The visual field of the mirror must be large, preferably room-size. This is because many children with ASD are also hyperactive and have short attention span. A large visual field allows the subject to not only explore the self-image,

but also the environment and social partners.

2. A consequence of the hyperactivity and hypersensitivity commonly seen in children with ASD is that it would be difficult to acquire proper training data to build multimedia models for rendering.

3. Most children with ASD are visual learners and tend to concentrate on visual details. Unwanted visual side effects caused by rendering delay and poor image quality can be highly distracting.

4. In addition to plane mirrors, other types of reflective surfaces such as curved mirrors, composite mirror systems or even arbitrary reflective surfaces such as one side of a race car may enhance the interest level and prolong the attention span of the subject.

6.1 Room-Size Virtual Mirror

These design objectives challenge the state-of-the-art technologies in visualization. In this section, I address these challenges by proposing a novel design of a room-size virtual mirror of arbitrary surface geometry. This design is illustrated in Figure 6.2. In the small prototype discussed in Chapter 4, the cameras are located around the display. Even with a relative small-size display (50" diagonal), the limited field of view of the cameras and the short distance between the subject and the display result in the rendering of only the top half of the mirror. In order to capture the full image of a subject in front of a large display, the only feasible solution is to replace the HDTV with a partially seethrough projector screen and place the cameras directly

behind the screen. Researchers have tried different techniques in creating a see-through display, such as half-silver mirrors[23], switchable liquid crystal diffuser[68], holographic projection screens [80], and weave fabric screens [81]. The weave fabric screen used in [81] is the preferred choice due to the relatively high percentage of pass-through light flux and its independence on the polarity and angle of the incident light. To prevent the light of the projector from entering the cameras, the combined projector and cameras system must be synchronized so that they will not be turned on at the same time. The consumer-grade Kinect devices do not support cross-device synchronization and their capturing frame rate is too slow. Instead, I plan to use an array of RGB cameras and two professional grade depth cameras to capture the light field and detailed depth field of the subjects in front of the screen. These cameras have much higher frame rates and can be easily synchronized with the projector. A network of Kinect devices will be placed outside of the light path of the projectors to capture the full view of the subject's body and the ambient environment. Interference between adjacent Kinects would be minimized through careful placement and calibration.

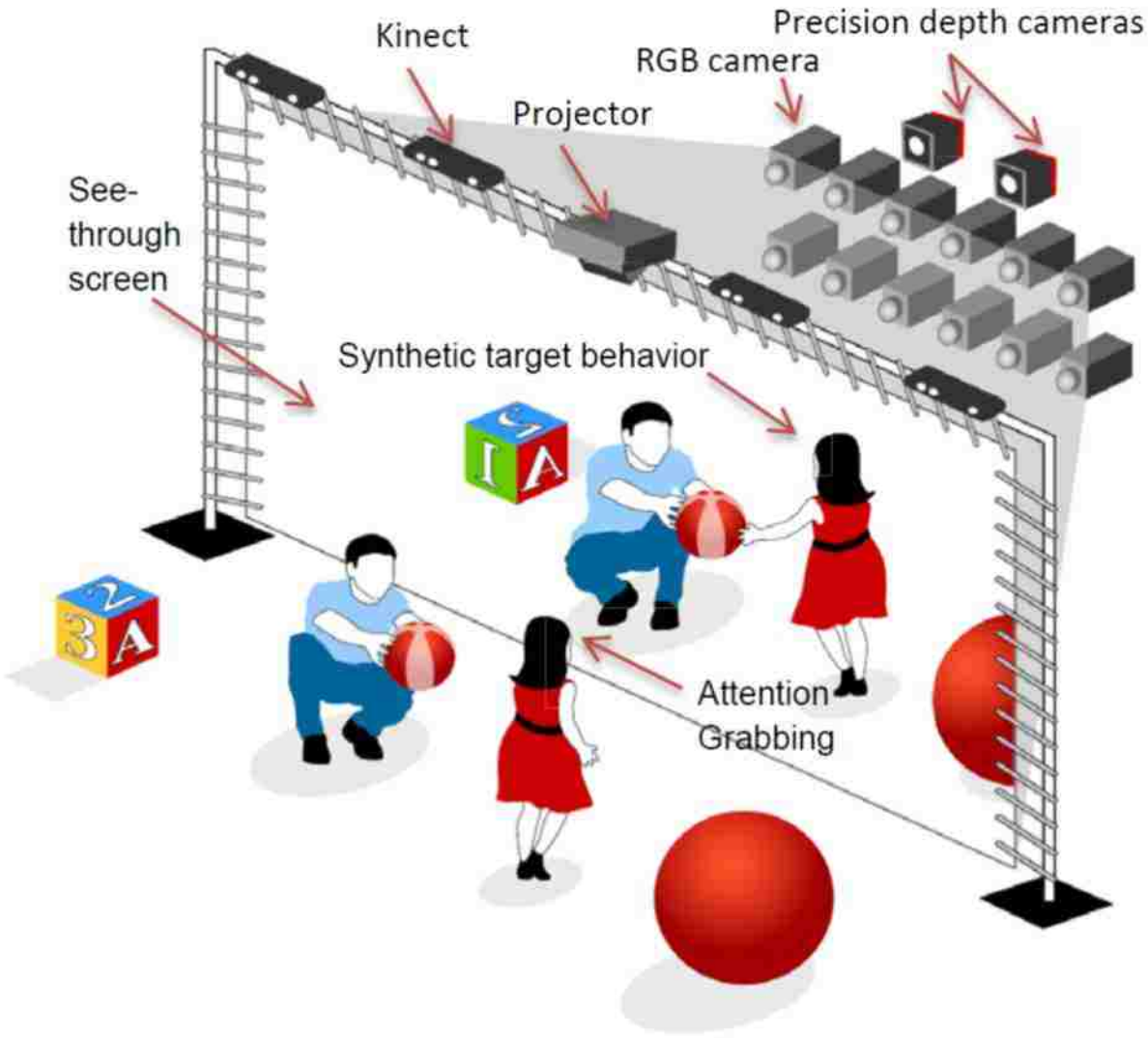


Figure 6.1: Aligned sphere movement trajectories by using our estimated extrinsics

6.2 Curved Virtual Mirror

To simulate the mirror experience, the proposed camera-display system will capture the 3D world, the viewer's position, and then renders what a viewer should see on a virtual mirror. The basic work-flow of our system is illustrated in Figure 11. The process is fundamentally different from our earlier prototype in order to cope with the rendering of curve surfaces. It is well known that for an arbitrary mirror surface,

there are few viewpoints at which a single perspective image can be obtained [123]. 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 singleviewpoint image. As such, using a virtual pinhole camera at the viewpoint to render the mirror image as described in Chapter 4 is no longer applicable. An alternative approach is to use the computer graphics approach of ray tracing by following the light rays from the virtual camera center [124]. This is problematic as a typical graphics rendering system requires a complete knowledge of the surface meshes of the 3D world, and can easily determine if the light ray has hit an opaque scene point. The depth cameras in our system only provide 3D point clouds. As our 3D point clouds spread over many cameras, geometric registration followed by surface meshing of all point clouds are computationally intensive and not suitable for real-time implementation.

Instead, I plan to traverse each 3D point S in the cloud to find the corresponding reflection point R on the mirror, which determines its reflection ray to hit the view point V. The determination of the reflection point is typically formulated as an iterated minimization problem based on the Snell law or the Fermat principle. Significant speedup can be achieved by pre-computing surface properties of the virtual mirror [125], restricting to special surfaces such as a quadric mirror [126], and taking advantages of the fast processing pipeline of GPUs. Once the reflection point is obtained, we can compute the intersection point P between the display surface and the light ray RV to determine the image location of the scene point. A Z-buffer is

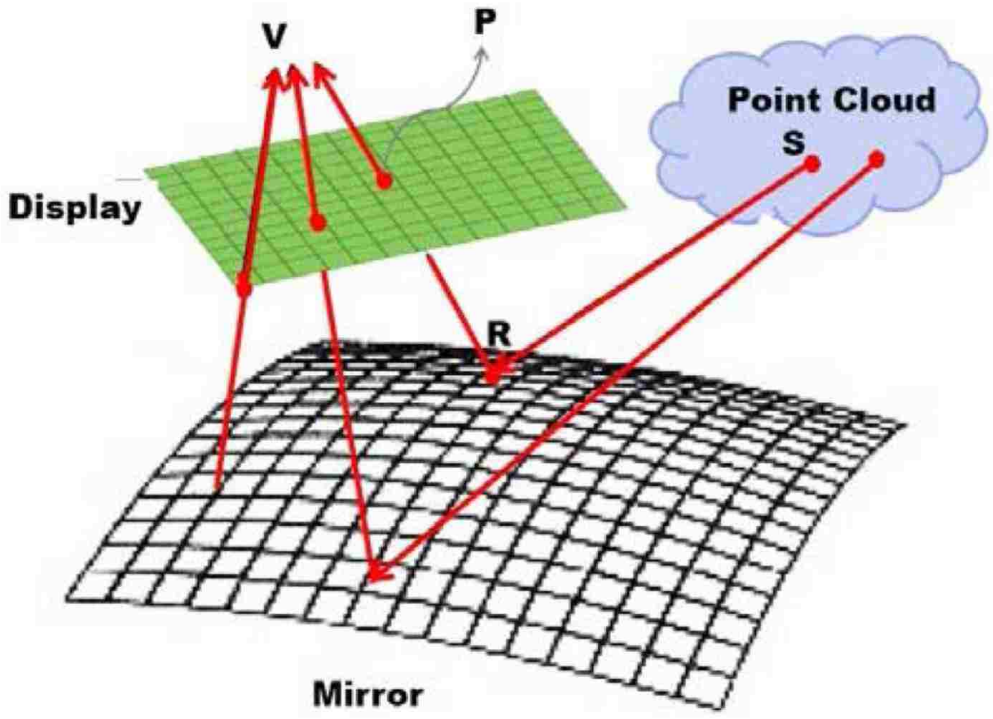


Figure 6.2: Aligned sphere movement trajectories by using our estimated extrinsics

used to determine if S is not occluded and indeed visible to the viewer based on the depth value. After identifying the closest scene point S, the light ray SR will be used to access the light field captured through the array of camera networks. The concept of light field was proposed by Levoy and Hanrahan [127] as a simplified 4D representation of the Plenoptic function. It has been extensively used for various real-time image-based rendering tasks of dynamic scenes [128][129]. Here we can use the light field to interpolate the closet spectral radiance as viewed by the subject at viewpoint V.

6.3 Self/Other Cognition in ASD

We believe that the virtual mirror system may be helpful in understanding some of the self/other social deficits associated with ASD. Many children with ASD appear to be highly interested in their own image in mirrors (in contrast to their typical lack of interest in social interactions). This is presumably due to the fact that images/actions in mirrors have a 100 percent contingency with one's image/actions and provide immediate and sustained feedback. Additionally, children with ASD appear to respond well to others imitating their actions (e.g., [92]). The Virtual Mirror will allow us to take advantage of both of these factors to enhance the understanding of self/other in children with ASD, resulting in superior social functioning. Success in this venture will indicate that the self/other system in ASD is amenable to modification, and suggest that a fundamental deficit in ASD is subject to environmental modification. More specifically, success in enhancing social behavior in children with ASD by contrasting their own image versus the images of another person imitating them will support the hypothesis by Lombardo et al. [16] that a key aspect of ASD is the failure to adequately understand "self" as an autonomous being in the social world.

The Virtual Mirror will be used to substitute the child's own image with another person's image resulting in real time imitation which will provide immediate and contingent feedback. Whereas individuals with ASD do not normally attend to others, the immediate feedback and contingency provided by the Virtual Mirror should enhance their interest and enable them to attend to another person imitating them in

the mirror. Moreover, back-to-back contrasts of ones own image with that of another individual made possible by the Virtual Mirror might enhance the understanding of the self/other distinction. So my next step is to test these possibilities and explore further of the therapeutic value of virtual mirror.

Bibliography

[1] M. Camplani and L. Salgado. Efficient spatio-temporal hole filling strategy for kinect depth maps. *Three-Dimensional Image Processing (3DIP) and Applications*, 8290(82900E-82900E), 2012.

[2] R.A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A.J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 127–136. IEEE, 2011.

[3] J. Diebel and S. Thrun. An application of markov random fields to range sensing. *Advances in neural information processing systems*, 18:291, 2006.

[4] H. J. Krouse. Video modeling to educate patients. *Journal of Advanced Nursing*, 33:748–757, 2001.

[5] R. W. McDaniel and v. A. Rhodes. Development of a preparatory sensor information videotape for women receiving chemotherapy for breast cancer. *Cancer Nursing*, 21:143–148, 1998.

[6] D. Nielsen, S. O. Sigurdsson, and J. Austin. Preventing back injuries in hospital settings: the effects of video modeling on safe patient lifting by nurses. *Journal of Applied Behavioral Analysis*, 42(3):551–561, 2009.

[7] A. M. Alvero and J. Austin. The effects of conducting behavioral observations on the behavior of the observer. *Journal of Applied Behavior Analysis*, 37:457–468, 2004.

[8] C. H. Hitchcock, P. W. Dowrick, and M. A. Prater. Video self-modeling intervention in school-based settings: A review. *Remedial and Special Education*, 24(1):36–45, January/Februrary 2003.

[9] V. S. Ramachandran, D. C. Rogers-Ramachandra, and S. Cobb. Touching the phantom. *Nature*, 377:489–490, 1995.

[10] T. Buggey. *Seeing Is Believing: Video Self Modeling for people with Autism and other developmental disabilities*. Wodbine House, 2009.

[11] P. W. Dowrick. Self-modeling. In *Using Video: Psychological and Social Applications*. New York: Wiley, 1983.

[12] A. Bandura. *Self-effiicacy: The Exercise of Control*. New York: Freeman, 1997.

[13] J. Ehara and H. Saito. Texture overlay for virtual clothing based on pca of silhouettes. In *Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality*, ISMAR '06, pages 139–142, Washington, DC, USA, 2006. IEEE Computer Society.

[14] 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 (CVPR2008)*, Anchorage, Alaska, USA, 24-26th June 2008. CVPR 2008.

[15] A. Taguchi, T. Aoki, and H. Yasuda. A study on real-time virtual clothing system based on two-dimensional plane model. In *Information and Telecommunication Technologies, 2005. APSITT 2005 Proceedings. 6th Asia-Pacific Symposium on*, pages 126 –130, nov. 2005.

[16] V. S. Ramachandran and E. L. Altschuler. The use of visual feedback, in particular mirror visual feedback, in restoring brain functions. *Brain*, 1:132:16931710, 2009.

[17] C. Rice. Disease control and prevention. *Autism spectrum disorders - data and statistics*, 1 2010.

[18] L.Q. Uddin, M.S. Davies, A. Scott, E. Zaidel, S. Bookkheimer, M. Lacoboni, and M. Dapreppo. Neural basis of self and other representation in autism: An fmri study of self-face recognition. *PLoS ONE*, 3(10), 2008.

[19] J. Shen, A. Raghunathan C. Ti, S.-C. Cheung, and R. Patel. Automatic lip-synchronized video-self-modeling intervention for voice disorders. In *e-Health Networking, Applications and Services (Healthcom), 2012 IEEE 14th International Conference on*, pages 244–249, 2012.

[20] M. Gurban and J.P. Thiran. Audio-visual speech recognition with a hybrid svm-hmm system. In *13th European Signal Processing Conference*, 2005.

[21] I. Arsic and J.P. Thiran. Mutual information engenlips for audio-visual speech. In *14th European Signal Processing Conference*, 2006.

[22] N. Kazuhiro, M. Noriaki, T. Kazuyoshi, and T. Naofumi. A real-time lip reading lsi for word recognition. In *Proc. IEEE Conf. ASIC*, pages 303–306, 2002.

[23] J Shen, PC Su, SC Cheung, and J Zhao. Virtual mirror rendering with stationary rgb-d cameras and stored 3d background. *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, 2013.

[24] Ju Shen and S.-C.S. Cheung. Layer depth denoising and completion for structured-light rgb-d cameras. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1187–1194, 2013.

[25] J. Ma, R. Cole, B. Pellom, W. Ward, and B. Wise. Accurate visible speech synthesis based on concatenating variable length motion capture data. *IEEE Transactions on Visualization and Computer Graphics*, 15:485–500, 2005.

[26] R. Queiroz, M. Cohen, and S. R. Musse. An extensible framework for interactive facial animation with facial expressions, lip synchronization and eye behavior. *ACM Computers in Entertainment*, 7(4):58:1 – 58:20, December 2009.

[27] Z. Deng and U. Neumann. Expressive speech animation synthesis with phoneme-level controls. *Computer Graphics Forum*, 27:2096–2113, 2008.

[28] Mermelstein P. Automatic segmentation of speech into syllabic units. *J. Acoust. Soc. Am.*, 58:880–883, 1975.

[29] Z. Xie and P. Niyogi. Robust acoustic-based syllable detection. In *INTERSPEECH'06*, 2006.

[30] Howitt A. Automatic syllable detection for vowel landmarks. *PhD Thesis*, 2000.

[31] Mertens P. Automatic segmentation of speech into syllables. *ECST*, pages 2009–2013, 1987.

[32] N. Eveno, A. Caplier, and P.-Y. Coulon. Key points based segmentation of lips. In *IEEE International Conference on Multimedia and Expo, 2002.*, 2002.

[33] Z. HAMMAL, N.EVENO, A.CAPLIER, and PY. COULON. Parametric models for facial features segmentation. In *IEEE Journal in Signal Processing*, 2005.

[34] W.H. Lau A.W.C Liew, S.H. Leung. Lip contour extraction using deformalbe model. In *International conference on Image Processing*, Sep 2000.

[35] N. Duy and H. David. Real-time face detection and lip feature extraction using field-programmable gate arrays. In *IEEE Trans. Systems Man Cybernet*, pages 902–912, 2006.

[36] Jingying Chen, Bernard Tiddeman, and Gang Zhao. Real-time lip contour extraction and tracking using an improved active contour model. In *Lecture Notes in Computer Science*, volume 5359, pages 236–245, 2008.

[37] Robert Kaucic, Barney Dalton, and Andrew Blake. Real-time lip tracking for audio-visual speech recognition applications. In *Lecture Notes in Computer Science*, volume 1065, pages 376–387, 1996.

[38] P.S. Aleksic and A.K. Katsaggelos. roduct HMMs for audio-visual continuous speech recognition using facial animation parameters. In *International Conference on Multimedia and Expo (ICME)*, pages 481–484, 2003.

[39] G. Potamianos C. Neti G. Gravier A. Garg and A. Senior. Recent advances in the automatic recognition of audio-visual speech. In *Proc. IEEE*, volume 91, pages 1306–1326, 2003.

[40] J.N. Gowdy A. Subramanya C. Bartels and J. Bilmes. Dbn based multi-stream models for audio-visual speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 993–996, 2004.

[41] M. Biehl, A. Ghosh, and B. Hammer. Learning vector quantization: The dynamics of winner-takes-all algorithms. *Neurocomput.*, 69:660–670, March 2006.

[42] V. Kitanovski and E. Izquierdo. 3d tracking of facial features for augmented reality applications. In *International Workshop on Image Analysis for Multimedia Interactive Services, Delft, The Netherlands*, 2011.

[43] C. Cullinan and S. Agamanolis. Reflexion: a responsive virtual mirror for interpersonal communication. In *European Conference on Computer Supported Cooperative Work*, 2003.

[44] A. R. J. Francois and E.-Y. E. Kang. A handheld mirror simulation. In *Proceedings of the 2003 International Conference on Multimedia and Expo - Volume 1*, ICME '03, pages 745–748, Washington, DC, USA, 2003. IEEE Computer Society.

[45] A. Hilsmann and P. Eisert. Realistic cloth augmentation in single view video. *Proc. of Vison, Modelling, and Visualization Workshop*, 2009.

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

[47] M. Straka, S. Hauswiesner, M. Ruther, and H. Bischof. A free-viewpoint virtual mirror with marker-less user interaction. *17th Scandinavian Conference on Image Analysis , SCIA 2011*, 2011.

[48] D. Porquet, J.M. Dischler, and D. Ghazanfarpour. Real-time high-quality view-dependent texture mapping using per-pixel visibility. In *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, GRAPHITE '05, pages 213–220, New York, NY, USA, 2005. ACM.

[49] C. Eisenacher, Q. Meyer, and C. Loop. Real-time view-dependent rendering of parametric surfaces. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, I3D '09, pages 137–143, New York, NY, USA, 2009. ACM.

[50] P. Debevec, Y. Yu, and G. Boshokov. Efficient view-dependent image-based rendering with projective texture-mapping. Technical report, University of California at Berkeley, Berkeley, CA, USA, 1998.

[51] V. Garro, C. dal Mutto, P. Zanuttigh, and G.M. Cortelazzo. A novel interpolation scheme for range data with side information. In *Visual Media Production, 2009. CVMP'09. Conference for*, pages 52–60. IEEE, 2009.

[52] 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.

[53] Q. Yang, R. Yang, J. Davis, and D. Nistér. Spatial-depth super resolution for range images. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[54] L. Wang, H. Jin, R. Yang, and M. Gong. Stereoscopic inpainting: Joint color and depth completion from stereo images. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[55] L.A. Torres-Méndez and G. Dudek. Reconstruction of 3d models from intensity images and partial depth. In *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 476–481. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

[56] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele. Joint bilateral upsampling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, 26(3), 2007.

[57] Q. Zhang, M. Ye, R. Yang, Y. Matsushita, B. Wilburn, and H. Yu. Edge-preserving photometric stereo via depth fusion. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

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

[59] R. Tsai. versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. In *ieee transactions on robotics and automation*, volume 3, pages 323–344, 1987.

[60] R.I. Hartley and A. Zisserman. Multiple view geometry in computer vision. In *Cambridge University Press*, 2000.

[61] J.H. Kim and B.K. Koo. Convenient calibration method for unsynchronized camera networks using an inaccurate small reference object. In *Opt. Express 20*, page 2529225310, 2012.

[62] D. Demirdjian, A. Zisserman, and R. Horaud. Stereo autocalibration from one plane. *european conference on computer vision (ECCV)*, 2000.

[63] Y. Xiao and R.B. Fisher. Accurate feature extraction and control point correction for camera calibration with a mono-plane target. *3D Processing on Visualization and Transmission, Paris*, 2010.

[64] J. Mallon and P.F. Whelan. Which pattern? biasing aspects of planar calibration patterns and detection methods. *Pattern Recognition Letter*, 28(8):921–930, 2007.

[65] R. Horaud, G. Csurka, and D. Demirdijian. Stereo calibration from rigid motions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1446–1452, 2000.

[66] C. Stauffer and K. Tieu. Automated multi-camera planar tracking correspondence modeling. *CVPR*, 2003.

[67] 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.

[68] 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.

[69] G. Pandey, J. R. McBride, S. Savarese, and R. Eustice. Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information. *AAAI*, 2008.

[70] W. Maddern, A. Harrison, and P. Newman. Lost in translation (and rotation): Rapid extrinsic calibration for 2d and 3d lidars. *ICRA*, 2012.

[71] 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.

[72] J. Zhu, L. Wang, R. Yang, and J. Davis. Fusion of time-of-flight depth and stereo for high accuracy depth maps. In *Computer Vision and Pattern Recognition*, pages 1–8, 2008.

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

[74] Y. Kim, D. Chan, C. Theobalt, and S. Thrun. Flexible camera calibration by viewing a plane from unknown orientations. *CVPR Workshop on time-of-light Camera based Computer Vision*, pages 1–7, 2008.

[75] M. Silva, R. Ferreira, and J. Gaspar. Camera calibration using a color-depth camera: Points and lines based dlt including radial distortion. In *Proc. of IROS Workshop in Color-Depth Camera Fusion in Robotics*, 2012.

[76] K. Verdolini and L. O. Ramig. Review: occupational risks for voice problems. *Logopedics, Phoniatrics, Vocology*, 26(1):37–46, 2001.

[77] D. R. Boone and S. C. McFarlane. *The Voice and Voice Therapy*. Prentice Hall, 2006.

[78] K. MacKenzie, A. Millar, J. A. Wilson, C. Sellars, and I. J. Deary. Is voice therapy an effective treatment for dysphonia? *A randomized controlled trial. British Medical Journal*, 323:658–661, 2001.

[79] N. Roy, B. Weinrich, S.D. Gray, K. Tanner, J. C. Stemple, and C. M. Sapienza. Three treatments for 2 teachers with voice disorders: a randomized clinical trial. *J Speech Lang Hear Res*, 46:670–688, Jun 2003.

[80] Johns MM Hapner E, Portone-Maira C. A study of voice therapy dropout. j voice. *Journal of Voice*, 23:337–40, May 2009.

[81] N. Roy, D.M. Bless, and N.C. Ford. Manual circumlaryngeal therapy for functional dysphonia: an evaluation of short- and long-term treatment outcomes. *Journal of Voice*, 11:321–331, 1993.

[82] L. O. Ramig and K. Verdolini. Treatment efficacy: Voice disorders. *Journal of Speech, Language and Hearing Research*, 41:101–116, 1998.

[83] R. Patel, D. Bless, and S. Thibeault. A novel intensive approach to voice therapy. *Journal of Voice*, 25:562–569, 2011.

[84] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[85] N. Eveno, A. Caplier, and P.-Y. Coulon. Accurate and quasi-automatic lip tracking. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14:706 – 715, 2004.

[86] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 14:321 – 331, 1988.

[87] Qineen Zheng and Rama Chellappa. Automatic feature point extraction and tracking in image sequences for arbitrary camera motion. *International journal of computer vision*, 15:31 – 76, 1995.

[88] CereProc, http://www.cereproc.com. *Text to Speech Technology.*

[89] J. F. Bonastre et al. Nist'04 speaker recognition evaluation campaign: new lia speaker detection platform based on alize toolkit. In *Proceedings of NIST Speaker Evaluation*, 2004.

[90] D. Sundermann et al. Time domain vocal tract length normalization. In *Signal Processing and Information Technology*, 2004.

[91] TIMIT acoustic-phonetic continuous speech corpus.

[92] B. Ingersoll. Teaching imitation to children with autism: A focus on social reciprocity. *Journal of Speech-Language Pathology and Applied Behavior Analysis*, 2(3):267–277, 2008.

[93] K. Khoshelham. Accuracy analysis of kinect depth data. In *ISPRS Workshop Laser Scanning*, volume 38, page 1, 2011.

[94] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli. Depth mapping using projected patterns. Technical Report Pending U.S. Patent No. 20100118123, Primse Sense Ltd., 2010.

[95] J.H. Cho, S.Y. Kim, Y.S. Ho, and K. Lee. Dynamic 3d human actor generation method using a time-of-flight depth camera. *Consumer Electronics, IEEE Transactions on*, 54(4):1514–1521, 2008.

[96] A. Maimone and F. Fuchs. Reducing interference between multiple structured light depth sensors using motion. *Virtual Reality Workshops (VR)*, 2012.

[97] W.T. Freeman, E.C. Pasztor, and O.T. Carmichael. Learning low-level vision. *International journal of computer vision*, 40(1):25–47, 2000.

[98] H. S. Bhat and N. Kumar. On the derivation of the bayesian information criterion. Technical report, University of California, Merced, 2010.

[99] C. Rother, A. Blake, and V. Kolmogorov. Grabcut - interactive foreground extraction using iterated graph cuts. page 309314, 2004.

[100] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM.

[101] J. Besl and N. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14:239–256, Feb 1992.

[102] D. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[103] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[104] J. Amanatides and A. Woo. A fast voxel traversal algorithm for ray tracing. In *In Eurographics 87*, pages 3–10, 1987.

[105] C. Yan, Y. Wang, and Z. Zhang. A novel real-time eye detection in human-computer interaction. *International Conference on Innovative Computing and Information*, 2011.

[106] M. Biehl, A. Ghosh, and B. Hammer. Learning vector quantization: The dynamics of winner-takes-all algorithms. *Neurocomputing*, 69(7-9):660–670, March 2006.

[107] PrimeSense Inc. *Prime Sensor NITE 1.3 Algorithms notes*, 2010.

[108] 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.

[109] J. Tong, J. Zhou, L. Liu, Z. Pan, and H. Yan. Scanning 3d full human bodies using kinects. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):643–650, April 2012.

[110] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, and B. Guo. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Trans. Graph.*, 31(6):136:1–136:11, November 2012.

[111] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley. Color transfer between images. *Computer Graphics and Applications IEEE*, 21:34–41, Oct 2001.

[112] V. Pratt and P. Point. Direct least-squares fitting of algebraic surfaces, 1987.

[113] Yongbin Li, Feng Chen, Wenli Xu, and Youtian Du. Gaussian-based codebook model for video background subtraction. 4222:762–765, 2006.

[114] H. Zhang, K.-Y. K. Wong, and G. Zhang. Camera calibration from images of spheres. In *IEEE Transactions on Pattern Analysis and Machine Intelligence. PAMI*, volume 29, page 499C503, 2007.

[115] O.D. Faugeras, Q.-T. Luong, and S.J. Maybank. Camera calibration from images of spheres. In *European Conference on Computer Vision (ECCV)*, pages 321–334, 1998.

[116] A.W. Fitzgibbon, M. Pilu, and R.B. Fisher. Camera calibration from images of spheres. In *IEEE Transactions on Pattern Analysis and Machine Intelligence. PAMI*, volume 21, pages 476–480, 1998.

[117] J. H. Challis. A procedure for determining rigid body transformation parameters. *Journal of Biomechanics*, 28(6):733 – 737, 1995.

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

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

[120] G. GOLUB and C. VANLOAN. Matrix computations. In *3rd edition, Johns Hopkins University Press, Baltimore, MD*, 1996.

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

[122] 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.

[123] S. Baker and S. Nayar. A theory of single-viewpoint catadioptric image formation. In *International Journal on Computer Vision*, volume 35(2), pages 175–196, 1999.

[124] M. Pharr and G. Humphreys. Physically based rendering: from theory to implementation. In *Morgan Kaufmann*, 2011.

[125] D. Roger and N. Holzschuch. Accurate specular reflections in real-time. In *In Proceedings of Eurographics*, 2006.

[126] N. Goncalves. On the reflection point where light reflects to a known destination on quadratic surfaces. In *Optics letters*, volume 35(2), pages 100–102, 2010.

[127] M. Pharr and G. Humphreys. Light field redering. In *In ACM SIGGRAPH*, 1996.

[128] X. Cao, Y. Liu, and Q. Dai. A flexible client-driven 3dtv systemfor real-time aquisition, transmission, and display of dynamic scenes. In *EURASIP Journal on Applied Signal Processing*, 2009.

[129] A. Isaksen, L. McMillan, and S. Gortler. Dynamically reparameterized light fields. In *In ACM SIGGRAPH*, 2000.

Vita

Ju Shen

Education:

- Ph.D. in Computer Science, University of Kentucky, United States, 2014.
- M.Sc. in Computer Science, University of Birmingham, United Kingdom, 2006.
- B.Sc. in Computer Science, Tianjin Normal University, China, 2005.

Selected Publications:

Journal Publications

- Ju Shen, Po-Chang Su, and Sen-ching Cheung. "Virtual Mirror Rendering with Stationary RGB-D Cameras and Stored 3D Background", IEEE Transactions on Image Processing, vol. 22, issue 9, pp. 1-16.

- Ju Shen, Changpeng Ti, Anusha Raghunathan, "Sen-ching Cheung, and Rita Patel. Automatic Video Self Modeling for Voice Disorder", Multimedia Tools and Applications, 2014.

Conference Proceedings

- Ju Shen, Wanxin Xu, Ying Luo, Po-Chang Su, and Sen-ching Cheung. "Extrinsic Calibration for Wide-Baseline RGB-D Camera Network", IEEE International Workshop on Multimedia Processing (MMSP2014), Jakartr, Indonesia 2014.

- Ju Shen and Sen-ching Cheung. "Layer Depth Denoising and Completion for Structured-Light RGB-D Cameras", IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2013), Portland, USA 2013.

- Ju Shen and Wai-Tian Tan. "Image-Based Indoor Place-Finder Using Image to Plane Matching", IEEE International Conference on Multimedia and Expo (ICME 2013), San Jose, USA 2013.

- Po-Chang Su, Ju Shen, and Sen-ching Cheung. "A Robust RGB-D Slam System for 3D Environment with Planar Surfaces", IEEE International Conference on Image Processing (ICIP 2013), Melbourne, AUS 2013.

- Nchuaobah Ukaobah, Ju Shen and Sen-ching Cheung. "MEBOOK: a novel device using video self-modeling to enhance literacy among children with ASD", The International Meeting for Autism Research (IMFAR 2013), San Sebastin, Spain 2013.

- Ju Shen, Changpeng Ti, Sen-ching Cheung and Rita Patel. "Automatic Lip-synchronized Video-Self-Modeling Intervention for Voice Disorders", IEEE International Conference E-Health Networking Application and Service (Healthcom 2012), Beijing, China 2012.

- Ju Shen, Jian Zhao, and Sen-ching Cheung. "Virtual Mirror By Fusing Multiple RGB-D Cameras", Asia-Pacific Signal and Information Processing Association (APSIPA 2012), California, USA 2012.

- Ju Shen, Anusha Raghunathan, Sen-ching Cheung and Rita Patel. "Automatic Content Generation for Video Self Modeling", IEEE International Conference on Multimedia and Expo (ICME 2011), Barcelona, Spain 2011.

Professional Services:

- TPC Member for ICME 2014.

- Co-Chair for the 2014 International Workshop on Wireless Vehicular Networks (WVN 2014)

- Invited reviewer for IEEE transactions on Multimedia 2012, 2013.

- Invited reviewer for Pattern Recognition 2012, 2013.

- Invited reviewer for IEEE International Conference on Multimedia and Expo (ICME), 2013.

- Invited reviewer for IEEE International Symposium on Circuits and Systems (ISCAS), 2013.

Talks:

- Hewlett-Packard Labs, Palo Alto, California, "Imaged-based Indoors navigation Cloud Service", Intern Talk, Host: Dr. Wai-Tian Tan, August 2012.

- IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Portland, Oregon, "Layer Depth Denoising and Completion for Structured-Light RGB-D Cameras", June, 2013.

- IEEE International Conference on Multimedia and Expo (ICME), San Jose, California, “Image-Based Indoor Place-Finder Using Image to Plane Matching”, July 2013

- IEEE International Conference on E-Health Networking Application and Service (Healthcom), Beijing China, “Automatic Lip-synchronized Video-Self-Modeling Intervention for Voice Disorders”, October 2012.

- IEEE International Conference on Multimedia and Expo (ICME), Barcelona, Spain, “Automatic Content Generation for Video Self Modeling”, July 2011.

- Halcomb Fellowship Presentation, Lexington Convention Center, “Video Self Modeling for Medical Therapies”, hosted by UK Center for Clinic and Traditional Science,2011.