REAL TIME 3D HUMAN CAPTURE SYSTEM FOR MIXED-REALITY ART AND ENTERTAINMENT

TA HUYNH DUY NGUYEN

NATIONAL UNIVERSITY OF SINGAPORE

2005

REAL TIME 3D HUMAN CAPTURE SYSTEM FOR MIXED-REALITY ART AND ENTERTAINMENT

TA HUYNH DUY NGUYEN

(B.Eng.(Hons.), Ho Chi Minh City University of Technology)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2005

Abstract

In this thesis, research into combining real world physical humans with virtual objects and scenes is detailed, which can be used for live 3D remote collaboration. This is achieved by the development of a research prototype which can capture humans live and in 3D and place them into a mixed reality environment. The users can then see the humans captured in 3D seamlessly interlinked with the virtual environment.

A robust and fast shape-from-silhouette algorithm is used to construct the 3D images of the subject. This thesis presents techniques to produce good quality and increase in speed of the whole system. The system frame rate is around 25 fps using only standard Intel processor based personal computers.

This research has an important focus on interactive media. We describe an application of the system in art and entertainment, named Magic Land, where 3D captured avatars of humans and 3D computer generated virtual animations can interact with each other to create an interactive story. This application demonstrates many technologies in human computer interaction. The user study results show the benefits and address some issues of these technologies.

Acknowledgement

I would like to express my sincere thanks to my supervisor, Dr Adrian David Cheok, for his invaluable guidance and support in my research work. Working under his supervision, I have received a lot of international research experiences.

I would like to thank Mr Lee Shang Ping, Xu Ke, Goh Kok Hwee, Siddharth Singh, Teo Hui Siang Jason, Teo Sze Lee, Ms Li Yu Jessie, Veron Ng, Liu Wei, and all other colleagues of mine in Mixed Reality Research lab, Singapore. Especially, I would like to thank Mr Tran Cong Thien Qui, my closely team mate during the last 3 years, for all his help and great collaboration. Those people helped me a lot during my 2 years in the lab in one way or another, and it has been always very cheerful working together with them in a team.

My thanks also go to my friends in Singapore: Xuan Linh, Mai Lan, Minh Hang, Thuy Tien, Lan Anh, Le Van, Nam Thang, for all their help, support and encouragement from my very early days at Singapore.

Last but not least, I would like to thank my parents and my sister for all their love and encouragement. They are the reason for all my success. This work is dedicated to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Distant communication and remote collaboration have always been significant needs of human beings. In recent years, many research has been conducted to develop new forms of remote collaboration and human-human communication which can eliminate the physical distances between people. Basing on Mixed Reality which is an advanced technology in human computer interaction, this research work, 3D Live system, is a novel remote collaboration system that can really bring remote people close together and put them in the real 3D physical space.

The 3D Live interface uses multiple video cameras to capture a remote participant and generates a virtual image of that person from the mixed reality user's viewpoint. In head mounted display, user can see real time 3D images of the remote participant overlaid onto her physical space as if he were really standing in front of her (see Figure 2.4). She can move around him or turn him in her hands to see from all sides in 3D form. In this way, 3D Live brings distant collaborators close

together and put them together in the real 3D physical space. As users can observe each other in 3D at any viewpoint easily, the nonverbal communication cues are completely transmitted. Consequently, 3D Live has large potential to become a new form of human communication and remote collaboration in near future [1], [2] .

With the 3D live capture characteristic, 3D Live application is not limited to human communication and remote collaboration only, but also can be extended to many other fields such as education, interactive art, entertainment, 3D mixed reality movies, 3D mixed reality books, etc. Especially in entertainment area, many advanced technologies in human computer interaction such as mixed reality and tangible interaction have been gradually changing 3D entertainment world. The application of 3D Live technology to 3D mixed reality games is expected to create a very new and innovative kind of games with more special effects and features.

However, the real implementation of 3D Live system has not reached the final goal of being a *real time* 3D capturing and rendering system. Some disadvantages of this system are:

- The processing and rendering speed is slow and there is no optimization in network transmission protocol, which causes large delay from capturing stage to final rendering stage.

- The image processing and rendering quality is not good enough.

The purpose of this thesis is to improve and optimize 3D Live system in term of quality and speed in order to bring 3D Live technology closer to our final goal.

This research looks at each processing stages in details to address the problems and propose ways to optimize its speed and quality. Another purpose of this thesis is to study the application of 3D Live technology in art, entertainment, and especially mixed reality game by building a novel mixed reality game system called Magic Land. The disadvantages and improvements of each processing stage of 3D Live system, together with specification of Magic Land system will be discussed in more details in the subsequent chapters. The remaining parts of this chapter will briefly present the main contributions of this work and the overview of the thesis content.

1.1 Contributions

The thesis studies on how to improve the speed and quality of 3D Live system so that it can run in *real time* with good quality and as small delay from capturing to rendering stage as possible. Details about the whole system processing stages will be discussed later in Chapter 3. However, in general, there are three main parts that need improving significantly as the following:

- Image processing: This stage is responsible for pre-processing the images after they have been captured from the cameras. Because of the characteristics of the visual hull rendering algorithm, foreground images, which are parts of images representing for the captured subjects only, must be extracted from the background scene. This step, called background subtraction, is the most critical step in the image processing stage.

- Networking: After processed at image processing stage, the foreground images are transferred through the network to the rendering machines, where they are used to re-generate images of the captured subjects in mixed reality space. The networking part needs to be optimized so that the transferring time is reduced as small as possible whereas the quality of service is still guaranteed.

- Rendering: This stage is responsible for producing the final images of the captured subjects in mixed reality space corresponding to the users' viewpoint. Basing on visual hull rendering algorithm of Matusik [3] , this stage is very computational expensive, which need to be improved in both processing speed and quality.

The major concerns of this work are in image processing and networking parts. Improvements of the rendering part, which is also very important, is presented in [4]. The main contributions of this thesis in 3D Live image processing and networking parts are summarized as followed:

- Image processing: As mentioned above, background subtraction is the most critical step in this stage which needs to be improved to make the system real time. This thesis proposes a new general framework for real time background subtraction algorithms which guarantees the very fast processing speed. The framework is general enough to apply for any pixel level and small region level background subtraction algorithms. A specific application of this frame-

work with a pixel level background subtraction algorithms is also presented. Moreover, this thesis also proposes post-processing techniques such as error filtering and image compression which improve both the speed and quality of the image processing stage.

- Networking: To reduce the data transferring time, IP multicast is proposed to send the foreground images to many users at the same time. The thesis also presents a design of RTP protocol for 3D Live data transmission. RTP/RTCP are famous real time media streaming protocols which supports IP multicast and guarantees the quality of service with many functions such as sending rate adjustment, congestion control, error correction, etc.

Beside the detailed improvements of image processing and networking parts of 3D Live system, this thesis also presents one typical application of 3D Live technology in art and entertainment. The application, named Magic Land, is the cross-section where art and technology meet. It not only combines latest advances in human-computer interaction and human-human communication: mixed reality, tangible interaction, and 3D Live technology; but also introduces to artists of any discipline intuitive approaches of dealing with mixed reality content. It brings together the processes of art creation, acting and reception in one environment, and creates new forms of human interaction and self reflection. Moreover, future development of this system will open a new trend of mixed reality games, where players can actively play a role in the game story.

1.2 Thesis Organization

Chapter 2 will discuss about the modern development of human communication and remote collaboration together with applications of mixed reality technology in this field. In this context, the contributions and the significant role of 3D Live technology in human communication and remote collaboration are emphasized. This chapter will also discuss about some state-of-the-art real time 3D capturing and rendering systems, and briefly compare them with 3D Live system.

Chapter 3 will describe the design of 3D Live system in details. The hardware and software structure of the system will be presented here. We will also emphasize the differences between the new system and the previous one. The system setup, including camera adjustment and calibration, will also be presented.

Chapter 4 is the detailed description of image processing module. This chapter presents all improvements of background subtraction algorithms, image post-processing and image compression. The novel general framework for real time background subtraction algorithms are presented there together with a specific implementation of this framework with a pixel level background subtraction algorithm. The optimization of these algorithms to make them faster are also presented.

Details of improvements in the networking part is presented in Chapter 5. In this chapter, we will discuss about an implementation of IP multicast and RTP protocol for 3D Live data. These improvements help the system run in real time for multiuser but still guarantee the quality of service at the same time.

Although this thesis does not concentrate on improving the 3D Live rendering stage, we will also describe the main ideas of this algorithm in Chapter 6 for the completeness. The problems and challenges of this stage will be discussed together with some basic ideas of how to solve them.

Chapter 7 presents the detailed design and implementation of Magic Land system, a typical mixed reality application of 3D Live system in art and entertainment. The hardware and software design of this system is presented. This chapter also discusses about some modern well known mixed reality games, and makes a detailed comparison of Magic Land with these games.

Finally, Chapter 8 concludes the thesis and presents some important publications and exhibitions of our work. The user study results of Magic Land system is also presented in Appendix A basing on a user study that was conducted during the exhibition at Singapore Science Center in September 2004.

Chapter 2

Background and Related Work

In this chapter, we will briefly discuss about the development of human communication means with mixed reality technology. The concept of mixed reality will be introduced together with its applications in human communication and remote collaboration. In this context, 3D Live is one of the advanced mixed reality applications in remote collaboration, in which user can see the instant 3D images of their remote friends in the real physical space when talking with them.

The second part of this chapter will discuss about some real time 3D capture systems. For readers' ease of following, other background and related work of each specific parts such as background subtraction, real time media networking and mixed reality applications in art and entertainment will be discuss later in each corresponding chapters.

2.1 Mixed Reality and Human Communication

Communication is one of the fundamental needs of human beings. Since the very beginning history of mankind, various means of communication, such as sound, voice, symbols, characters, languages, etc., were invented and have never stopped developing up till today. However, after thousand years of revolution with many breakthroughs in communication means such as telephone, mobile phone, etc., the question of how to communicate with other people at other places more and more effectively is still a main concern of human beings nowadays.

In recent years, the fast development of digital and computer technologies has created many breakthroughs in human-human communication. For example, mobile phone is one of those significant breakthrough by connecting people at anywhere and anytime. The invention of mobile phone has made a very large impact in our modern society by changing people's habits of communication and lives. On the other hand, 2D video-conferencing is another breakthrough by allowing people at far distances to see each other when communicating. Following this trend, the integration of video conference in mobile phone will soon become popular, so that people can communicate with and concurrently see their remote friends at anywhere and anytime [5] .

Seeing each other when communicating is a very significant need in human-human communication, as stated by Arthur Strand in 1898: *"If, as it is said to be not likely in the near future, the principle of sight is applied to the telephone*

as well as that of sound, earth will be in truth a paradise, and distance will lose its enchantment by being abolished altogether." However, the 2D images in video-conferencing cannot fully satisfy people's perception of sight, as human beings sense the world in 3D space and 2D images cannot convey non-verbal cues such as body motion fully and completely. Moreover, in 2D video-conferencing, users have to stay at one specific place, in front of their monitor. This limitation of communication location can be solved by integrating 2D video-conferencing in mobile devices, which will allow people to communicate and see their distant friends at anywhere and anytime. Nevertheless, the limited 2D screen of mobile devices may worsen users' viewing satisfaction [5] .

The problem of how to make human communication available at anywhere and anytime, and, at the same time, allow remote users to see each other when communicating has been addressed by many researchers around the world. Mixed Reality (MR) technology is the successful approach for this problem. Many work have been conducted to explore how mixed reality technology can create fundamentally new forms of human communication and remote collaboration.

In general, mixed reality is a form of environment in which the real world and virtual objects, or the real objects and virtual world are merged together. Figure 2.1 is the very famous depiction of Milgram about mixed reality and reality-virtuality continuum concept [6] . Typically, users view mixed reality environment through a handheld or head mounted display (HMD) and they will see the overlay of computer-generated graphical content onto the real scene.

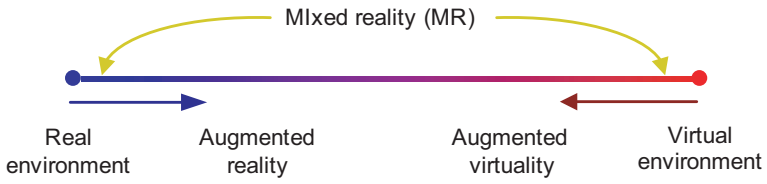

Figure 2.1: Milgram's representation of reality-virtuality continuum

Previous research developed interfaces that do overlay virtual 2D video of remote participants on the real world. In the Augmented Reality Conferencing work, users wear a video see-through HMD connected to a computer. When they look at a real name card, they can see a life-sized, virtual video image of a remote collaborator (see Figure 2.2). The cards are physical representations of remote participants. Users can arrange multiple cards about them in space to create a virtual meeting space and the cards are small enough to be easily carried, ensuring portability [7] .

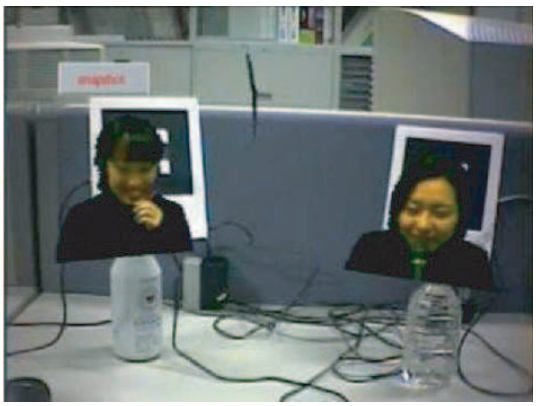

Figure 2.2: AR Conferencing - Users see remote collaborators at their real name cards. *(Developed by Prof. Mark Billinghurst and Prof. Hirokazu Kato)*

With this technique, users are no longer constrained in the limited 2D screen,

but can place and see the real-time images of their remote friends at anywhere in the physical space. Moreover, the current increase in power of mobile devices and availability of wireless service are enabling the integration of this technique in mobile devices, thus, opening the prospect of talking and seeing anyone at anywhere and anytime. However, in this application the live video was texture mapped onto a flat polygon, so the remote users appeared flat. This made it still difficult to convey some nonverbal cues, such as pointing and body motion.

The purpose of 3D Live research is to overcome this limitation. Let's imagine that we can see our distant friends in full 3D, standing in front of us in the real world, while talking with them over the phone. This vision of human communication was long appeared in human's dreams, and fairy tales, just like the image of the future presented in the holographic form of Princess Leia in Star Wars (Figure 2.3). The 3D Live technology, which captures humans in 3D and placing them into a mixed reality environment at other places in real time (Figure 2.4), is the attempt to make this vision real. It opens a new chapter for human communication, where not only voice but also gesture and body motion are transmitted completely in full 3D. Thus, it eliminates the distances and really brings people close together. Furthermore, the technology allows humans to participate directly and in real time as an embodied person in the 3D virtual world.

There is a number of other significant differences between this and traditional video conferencing. The remote user can appear as a life-sized image and can view an arbitrary number of remote users simultaneously. Because we can place the

Figure 2.3: Princess Leia appears in holographic form in Star Wars, a very famous film by George Lucas.

Figure 2.4: 3DLive's goal - User see his distant friend in full 3D in real time.

3D Live mixed reality human characters about the users in the physical space, the system can restore spatial cues to the human remote collaboration. Perhaps the greatest advantage to this system is that users are no longer tied to the desktop and can conference from any location. This means that the remote collaborators can become part of any real-world surroundings, potentially increasing the sense of social presence. We believe that 3D Live technology will be the future of human communication, where the effect of distances will be eliminated and people will be brought closer together.

2.2 Realtime 3D Capture Systems

Up to now, the idea of capturing human beings for virtual reality has been studied and discussed in quite a few research articles. In [8], Markus et al. presented "blue-c", a system combining simultaneous acquisition of video streams with 3D projection technology in a CAVE-like environment, creating the impression of total immersion. Multiple live video streams acquired from many cameras are used to compute a 3D video representation of a user in real time. The resulting video inlays are integrated into a virtual environment. In spite of the impression of the total immersion provided, blue-c does not allow tangible ways to manipulate 3D videos captured. There are few interactions described between these 3D human avatars and other virtual objects. Moreover, blue-c is currently a single user per portal [8] , and thus does not allow social interactions in the same physical space. Our system, in contrast, supports multi-user experiences. Furthermore, mixed reality system makes the 3D images appear as if they are in the real world physical environment and thus creates social interactions.

Another 3D capture system was also presented in [9]. In this paper, the authors demonstrate a complete system architecture allowing the real-time acquisition and full-body reconstruction of one or several actors, which can then be integrated in a virtual environment. Images captured from four cameras are processed to obtain a volumetric model of the moving actors, which can be used to interact with other objects in the virtual world. However, the resulting 3D models are generated

without texture, leading to some limitations in applying their system.

A system which has the most similar goals with 3D Live was developed in [10]. This system captures the whole soccer stadium when the soccer game is currently progressing, and renders the whole 3D images of this game in mixed reality scene. However, using image mosaicking technique, the system is not fast enough to update real image situation of the fast soccer games in real time.

Chapter 3

3D Live System Description

This chapter will describe the 3D Live system in details. Firstly, we will look at the hardware components, their functions and connection diagram. After that, some system setup procedures such as camera adjustment and calibration will be described. Last but not least, we will look at the software components of the system in general. Some differences in hardware between this system and the previous one will also be discussed in this chapter.

3.1 Hardware

Figure 3.1 represents the overall system structure. Eight Dragonfly FireWire cameras from Point Grey Research [11], operating at 30fps, 640 x 480 resolution, are equally spaced around the subject, and one camera views him/her from above. Three Sync Units from Point Grey Research are used to synchronize image acqui-

sition of these cameras across multiple FireWire buses [11]. Three Capture Server machines, each one being DELL Precision Workstation 650 with Dual 2.8GHz Xeon CPUs and 2GB of memory, receive the three 640x480 video-streams in Bayer format at 30Hz from three cameras each, and pre-process the video streams. The pre-processing stage will be described later in more details.

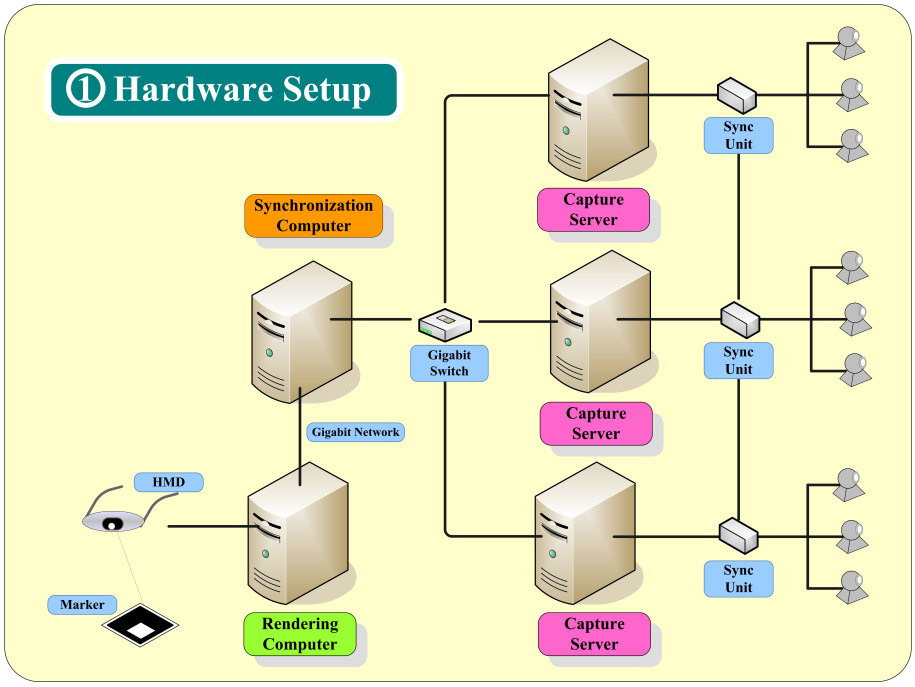

Figure 3.1: Hardware Architecture

The Synchronization machine is connected with three Capture Sever machines through a Gigabit network. This machine receives nine processed images from three Capture Server machines, synchronizes them, and sends them also via gigabit Ethernet links to the Rendering machine, which is another DELL Precision Workstation 650.

The user views the scene through a video-see-through head mounted display

(HMD) connected directly to the Rendering machine. A Unibrain firewire camera, capturing 30 images per second at a resolution of 640x480, is attached to the front of this HMD. The Rendering machine obtains images from this Unibrain camera, tracks the marker pattern on these images, calculates the position of the virtual viewpoint, generates a novel view of the captured subject from this viewpoint and then superimposes this generated view to the images obtained from the Unibrain camera and displays it on the HMD. Details of each step will be discussed later in Section 3.3 of this Chapter.

There are several important differences between this system and the previous one. Not to mention about the more powerful machines, the first obvious difference is the number of cameras. Whereas the previous system uses 15 cameras to capture the subjects, only 9 of them are used in our new system. Generally, the more cameras are used, the better the quality of the final rendering result is. However, due to the limited bandwidth of the firewire bus, only 3 cameras operating at 30fps and 640x480 resolution can be connected to one firewire bus of one computer at one time. Consequently, the decreased number of cameras helps to reduce the complexity and expensiveness of the system a lot.

Moreover, although the new system uses only 9 cameras, it still guarantees the good quality of the rendering result due to the special placement of the cameras around the captured subjects. This special placement of cameras has one camera on top to capture the subjects from above. The top camera is very important in producing the good result, which will be discussed later, and it is another differ-

ence with the previous system, which only places the cameras equally around the captured subjects without any camera on top.

Another important difference between the current system and the previous one is the camera type and the sync units. It is very important that, at one frame, all images received from 9 cameras are captured at approximately the same time to guarantee the good rendering result. With some specific camera types, the cameras can be automatically synchronized if they are connected to the same firewire bus. However, to synchronized cameras on different firewire buses, for example those connected to different computers, is a difficult problem if the camera does not have external trigger signal input pin which is used by an external device to control the camera capture. For Dragonfly cameras, the problem is easily solve without the need of building such an external device. Dragonfly cameras which are connected to the same firewire bus are automatically synchronized, and the sync units are used to synchronize the capturing trigger signal of Dragonfly cameras across multiple firewire buses. There are also the timestamp encoded by the camera firmware in the first 4 bytes of each capture image to denote the unique time when this image was captured. This timestamp information are very useful for the software to synchronize images captured from cameras connected to different computers. More details about this synchronization problem will be discussed later in software components section.

3.2 System Setup

First of all, in order to generate the novel view of the subject from any angle/position of the virtual viewpoint, the zoom level, angle and position of each Dragonfly camera must be adjusted so that it can capture the whole subject even as he/she moves around. Moreover, to guarantee that the constructed visual hull is close enough to the object's shape, the zoom level and the position of each camera should be adjusted so that the camera looks at the subject at a far enough distance. The camera on top to view the subject from above is also to serve this purpose.

The system is very sensitive to the cameras' intrinsic and extrinsic parameters, because the visual hull construction algorithm bases on the relative distances among cameras as well as the distances between the subject and the cameras. Consequently, after being adjusted, the position, zoom level, and angle of each camera have to be fixed, so that the camera's parameters are not changed anymore. The next step is to calibrate all the cameras to get the necessary parameters. Both the Unibrain camera attached to the HMD, and the Dragonfly cameras which capture the subject have to be calibrated. The intrinsic parameters of these cameras can be estimated using standard routines available with ARToolkit [12] or MXR-Toolkit [13] .

For the Dragonfly cameras, we must not only estimate the intrinsic parameters, but also the extrinsic parameters to get the spatial transformation between each of the cameras. Calibration data is gathered by presenting a large checker-board to

all of the cameras. For our calibration strategy to be successful, it is necessary to capture many views of the target in a sufficiently large number of different positions. Standard routines from Intel's OpenCV library [14] are used to detect all the corners on the checkerboard, in order to calculate both a set of intrinsic parameters for each camera and a set of extrinsic parameters relative to the checkerboard's coordinate system. Where two cameras detect the checkerboard in the same frame, the relative transformation between the two cameras can be calculated. By chaining these estimated transforms together across frames, the transform from any camera to any other camera can be derived [1],[2] .

3.3 Software Components

3.3.1 Overview

All basic modules and the processing stages of this system are represented in Figure 3.2. The Capturing and Image Processing modules are placed at each Capture Server machine. After Capturing module obtains raw images from the cameras, the Image Processing module will extract parts of the foreground objects from the background scene to obtain the sillhouettes, compensate for the radial distortion component of the camera mode, and apply a simple compression technique.

The Synchronization module, on the Synchronization machine, is responsible for getting the processed images from all the cameras, and checking their timestamps to synchronize them. If those images are not synchronized, basing on the timestamps,

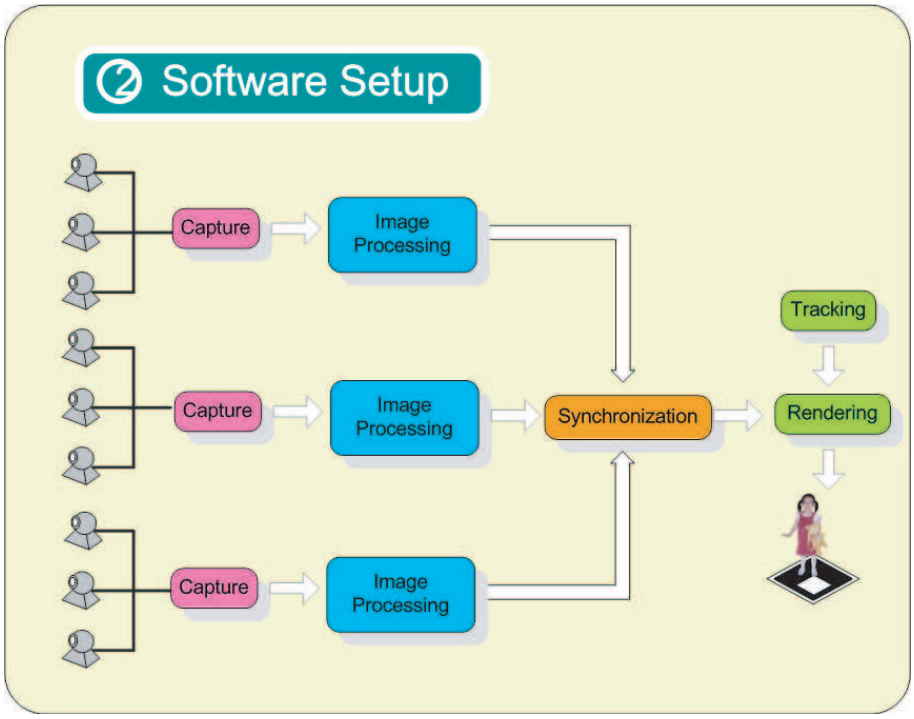

Figure 3.2: Software Architecture

Synchronization module will request the slowest camera to continuously capture and send back images until all these images from all nine cameras appear to be captured at nearly the same time.

The Tracking module will obtain the images from the Unibrain camera mounted on the HMD, track the marker pattern and calculate the Euclidian transformation matrix relating the marker co-ordinates to the camera co-ordinates. Details about this well-known marker based tracking technique can be found at [2], [12], or [13].

After receiving the images from the Synchronization module, and the transformation matrix from the Tracking module, the Rendering module will generate a novel view of the subject based on these inputs. The novel image is generated such that the virtual camera views the subject from exactly the same angle and position

as the head-mounted camera views the marker. This simulated view of the remote collaborator is then superimposed on the original image and displayed to the user.

The subsequent parts will discuss more detail about the techniques we use in each module.

3.3.2 Image Processing Module

The Image Processing module processes the raw captured images in three steps: background subtraction (which extracts parts of the foreground objects from the image to obtain the silhouettes), radial distortion compensation, and image size reduction. The second step is done by applying the intrinsic parameters of the camera to estimate the correct position of each pixel. The other two steps are among the critical parts in 3D Live system, and they are the concentration of this thesis. The details of this part will be presented in Chapter 4.

3.3.3 Synchronization

The main function of Synchronization module is to receive and synchronize images which have been processed by Image Processing module. The purpose of synchronization is to ensure that all images are captured at the same time.

Figure 3.3 describes the data transferred from Image Processing to Synchronization. It includes three parts. The first part is the image which is processed by Image Processing Module. Instead of sending the whole image, we only transmit the smallest rectangle area of the original image that contains the silhouette. This

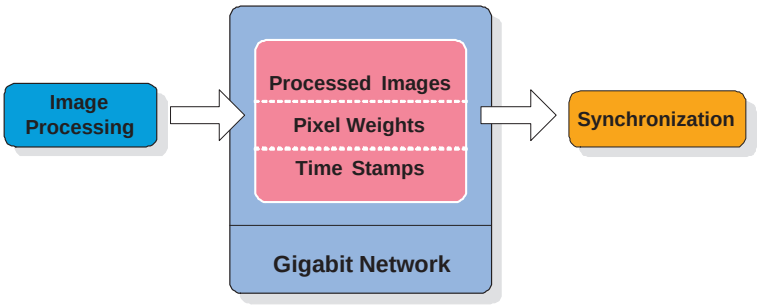

Figure 3.3: Data Transferred From Image Processing To Synchronization

significantly reduces the amount of data to be transmitted. The second part is the pixel-weights for this image. These weights will be used for blending color in the rendering steps. We will present more about this weight in the Rendering chapter of this thesis. The last part to be transmitted is the Time Stamp, which is the time when this image is captured. Using this timing information, the Synchronization module will synchronize images captured from all nine cameras.

Once receiving one set of images from nine cameras, the time stamp of each image will be compared. If the difference in time between the fastest camera and the slowest camera is larger than 30 ms, the Synchronization Module will require Image Processing Module to provide a new image from the slowest camera. This synchronizing process will keep looping until the difference is smaller than 30ms. The reason to choose 30ms as the threshold is because our cameras operate at 30 fps.

3.3.4 Networking

To transfer processed images data from Capture Server machines to Synchronization machine, TCP/IP protocol can be used to guaranteed the correctness of the transferred data. However, previous version of 3D Live system also uses TCP/IP to transfer data from Synchronization machine to Render machine, which slows down the system severely and makes it cannot run in real time.

The Networking part mentioned here refers to the networking protocols used in transferring data from Synchronization machine to Render machines. In this new version of 3D Live system, an IP multicast and RTP-like protocol is designed and implemented to guarantee the fast and continuous streaming of data and good quality of service. Details of this part will be presented in Chapter 5.

3.3.5 Rendering

The rendering algorithm used in this system is a new development over our previous algorithm which is described in [1]. To improve the speed and quality, we introduce new ways to compute visibility and blend color in generating images for novel viewpoints. More details of this part will be describe in Chapter 6.

Chapter 4

Image Processing

The Image Processing module processes the raw captured images in three steps: background subtraction (which extracts parts of the foreground objects from the image to obtain the silhouettes), radial distortion compensation, and image size reduction. The second step is done by applying the intrinsic parameters of the camera to estimate the correct position of each pixel. The remaining of this chapter will concentrate on the background subtraction and image size reduction steps.

4.1 Background subtraction

4.1.1 Literature review

Background subtraction, as known as foreground segmentation, refers to techniques to extract parts of the image corresponding to the objects of interest from the background scene. Background subtraction is one of the significant pre-processing

steps in many vision systems. It has large number of applications in widely different fields, such as surveillance systems, human-computer interface, video compression, etc. Consequently, background subtraction is one of the very attractive research topics nowadays.

Some challenges of background subtraction problem can be listed as the followings:

- Dynamic background scene: The background scene can be changed. With outdoor systems, it can be large changes or small but frequent changes such as waving trees. The indoor system is also not always static. It can have large changes such as the moving or disappearing of some background objects like chairs, tables, etc.

- Illumination changes: For outdoor system, the illumination can be gradually changed following the time of day as it depends on the sun lighting outside. For indoor system, sudden changes in illumination can be caused by the switching of indoor lights.

- Camouflage: Foreground objects' characteristics, such as color, may be similar to that of the background scene covered by those objects.

- Shadows: Shadows caused by the foreground objects may easily be classified as foreground regions.

Many approaches have been proposed to solve this problem. However, none of them has reached the final goal of being real time and producing good quality. This

part will give a review on the achievements of background subtraction techniques that have been developed recently.

Classification of background subtraction techniques

The term "*technique*" is used instead of "*algorithm*" here to emphasize that background subtraction problem can be solved by using not only software, but also some types of hardware devices. There is a class of background subtraction techniques that use hardware devices specially designed for extracting foreground objects out of the background scene. One example of this class is the IR keying system proposed in [15], which uses an infrared camera to extract silhouette images from the retroreflective scene. This technique is developed basing on the fact that the intensity of infrared reflected from foreground objects is lower than that reflected from the screen. This approach has many advantages, such as very fast speed, robustness in lighting condition, very good capability of eliminating shadows and coping with changing background scene, etc. These advantages make it suitable for being used with indoor vision systems. However, not to mention about the expensiveness of infrared cameras, this approach can only be used with live captured subjects, such as animals or human beings. Other foreground subjects that reflect the same or lower intensity of infrared than the background scene does will not be distinguished from the background scene.

This thesis concentrates on software background subtraction algorithms only. There are several ways to classify background subtraction algorithms basing on

their characteristics or specific purposes of development. For example, they can be classified as indoor versus outdoor algorithms, algorithms for static versus dynamic scene, or statistical versus non-statistical approaches, etc. In this thesis, the background subtraction algorithms are classified basing on which level of image information they used for calculation and classification: at each pixel, at small region, from whole image or any combinations of them. Consequently, we can classify background subtraction algorithms into 4 categories: pixel level algorithm, region level algorithm, image level algorithm and their combinations (Figure 4.1).

Pixel level background subtraction algorithms

Pixel level background subtraction algorithms are those that only base on the properties of each pixel, such as color, depth, etc., to classify it. From this point of view, background subtraction is a type of classification problems which classifies each pixel into two classes: background and foreground. To classify pixel A, the algorithms use the information of A only, but do not utilize high-level information or correlation of A with its neighbor pixels. These algorithms, thus, are simpler and faster than but normally not as qualitatively efficient as other categories. Some examples of this types are [16], [17], [18],etc.

Most of algorithms in this categories utilize pixel color as the major information to classify the pixels. A few of them use depth information at each pixel. With color information, perhaps, the simplest approach of this type is to use statistical mean value of the pixel color, and classify a pixel as foreground if its new color

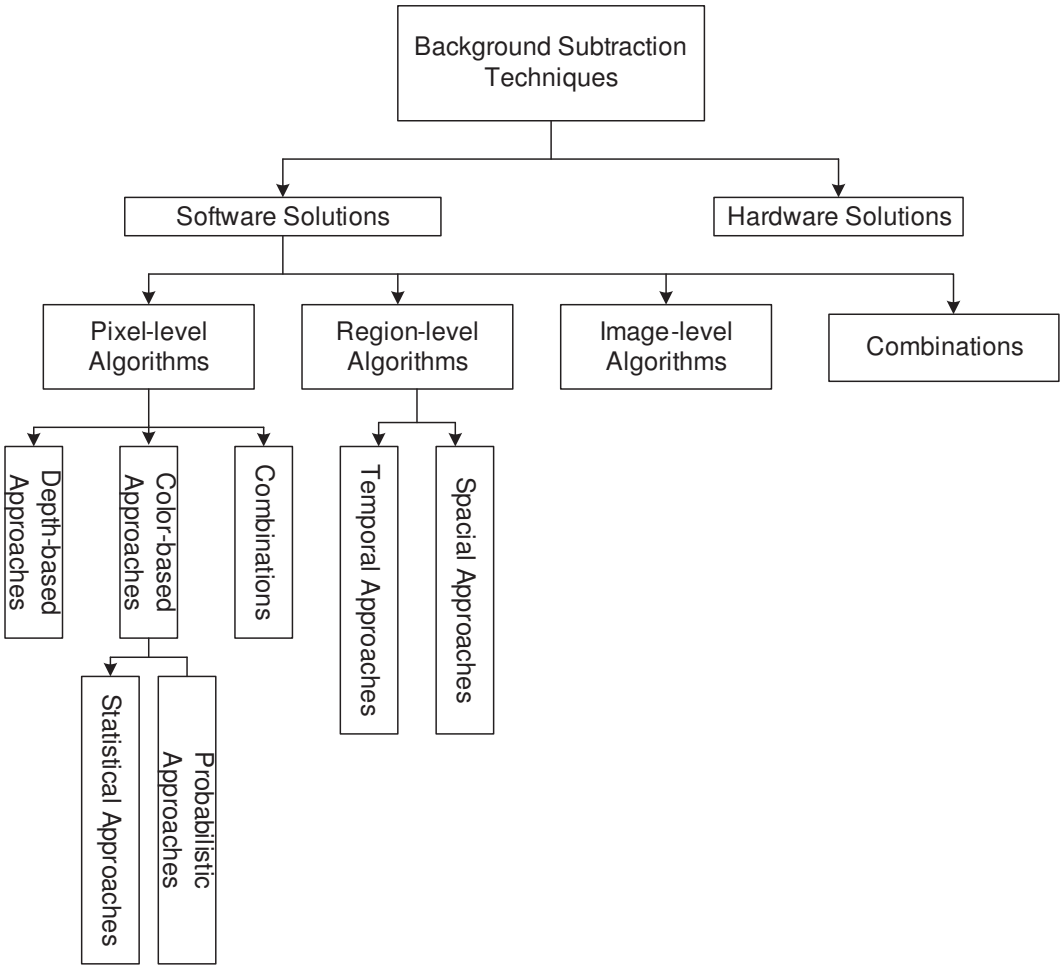

Figure 4.1: A simple classification of Background Subtraction techniques

is largely different from the mean value. However, this simple approach suffers from all types of misclassification errors that have been mentioned above: noise, illumination changes, shadows, camouflage, etc. To overcome these challenges, more complicated approaches have been proposed. Among those most common approaches are probabilistic approaches and statistical approaches.

The very first probabilistic approach for this problem was proposed in [16]. This approach can detect slow-moving objects, identify and eliminate shadows effectively. The main idea of this method is to classify each pixel into "moving object", "shadow", or "background" classes using a probabilistic model of its color. Basically, this method maintains a Mixture-of-Gaussians classification model for each pixel. Each Mixture-of-Gaussians model at each pixel includes 3 Gaussian probability distributions modeling for the color distribution when the pixel belongs to 3 corresponding classes "moving object", "shadow" or "background". A heuristic is used to assign which Gaussian components corresponding to which classes. Consequently, when given a new value for that pixel, it can be classified to the corresponding class basing on which Gaussian distribution gives the highest posteriori probability. Once the current pixel value is classified, its probabilistic classification is then used to update the models appropriately. This update helps to classify "slow-moving" objects by preventing the object's pixel value not to mix with the Gaussian distribution which models for the "background". With this approach, the most important and basic task is to learn the parameters of each Gaussian of each pixel. This is done by using EM algorithm [17] for learning the mixture models.

To deal with the slow speed of the traditional EM algorithm, the incremental EM algorithm is proposed.

The idea of automatically updating the background model makes the algorithm adaptive to deal with the large changes of the background scene, such as illumination changes. The algorithm is also tested in the application of locating and tracking moving vehicles in freeway traffic. However, the results are not very good as the camouflage effect can be noticed easily. Another drawback of this method is that it will not work well in the extreme lighting conditions and cannot detect dark foreground objects, because it uses an heuristic assuming that the color of foreground pixels are normally lighter than that of background pixels.

Another well-known method in "probabilistic approach" group is proposed in [18]. It can work with outdoor environment in real time, and can cope with lighting changes, repetitive motions of the scene elements (such as the movement of fan), slow-moving objects and large changes of background object position. It is an adaptive technique which can adapt with the changes of the background scene over time.

The basic idea of this approach is also to use a Mixture-of-Gaussians to model the probability distributions of colors at each pixel over time. However, unlike the previous method which uses each Gaussian component representing for each class and uses heuristics to assign the correspondences between Gaussian components and their corresponding classes, this algorithm maintains a number of Gaussian components representing for each class and assigns them to the corresponding class

automatically.

The criteria of automatically choosing which Gaussian components represent for the background class is that these components must have large contribution to the mixture (specified by their weights) and small variance parameters. So, the Gaussian components are sorted in the increasing order of their ratios of weight over variance, and then, the first B distributions that have their summary of weights exceeds a threshold T are chosen to represent for the background model.

By automatically determining which Gaussian components corresponding to the background pixel class, the current pixel is classified to this class if its value matches the distribution of one of those Gaussian components. That values are then used to update the Gaussian distributions. So, the parameters (means and variances) and the contributions to the mixture of these component distributions are changed over time. This helps the model adapt with background changes effectively.

It is a real time method that can effectively cope with global and local illumination changes, and repetitively moving background objects. It also can adapt quickly with large changes of the background scene. However, if the foreground objects are static, they will be classified wrongly as background over time. More importantly, this method does not provide a mechanism to eliminate objects' shadows.

Similar to [18], another approach is proposed in [19] adding depth information as another dimensions of each Gaussian distributions together with YUV color values. However, this approach is not efficient. It is very computational intensive, because of not only the increasing number of dimensions of each Gaussian distribution but

also the complexity of calculating depth information at each pixel.

Another group of this category uses statistical approach instead of probabilistic approach. A statistical approach proposed in [20] can detect moving objects and their shadings. It can cope with global and local illumination changes, such as highlights and shadows. However, this method requires the background scene to be static, but, as stated in [20], the algorithm also works fairly well with outdoor environment. It is not highly adaptive and cannot cope with large changes in the background scene, nevertheless, it is suitable for applications with static foreground objects.

The algorithm requires a learning stage to get the statistics of background scene features. This learning stage is computational intensive, but the classification stage after that is very fast and can be considered as "real-time". The main idea of this method is to calculate the statistics of background pixel properties over time to get the statistical values modeling for the background pixel. The pixel properties to be calculated here are the chromaticity and the brightness getting from a new model of pixel color. By computing the statistics of chromaticity distortion and brightness distortion of each background pixel in the learning stage, the algorithm can then classify each pixel into "foreground", "background", "highlighted background" or "shadow/shading background" classes, after getting its new brightness and chromaticity values.

Among a few methods using depth information, with multiple cameras, the approach proposed in [21] uses stereo disparity as the property to classify each

pixel. Instead of directly calculating depth information at each pixel, which is very computational intensive, the algorithm only uses stereo disparity between a primary and an auxiliary camera views to classify the pixel. It states that for static background scene the stereo disparity is also static. Thus, it builds a disparity map for all pixels in the background scene as a background pixel-to-pixel transformation from the primary to the auxiliary camera views. The disparity map is built off-line, and any pixels of the new incoming frame that violate this map will be classified as foreground pixels in real time. This method is simple, fast and can deal with shadows and illumination changes. However, it works with static scenes only and the quality has much noise as can be seen in the paper [21].

Region level background subtraction algorithms

Region level background subtraction algorithms are those that utilize information of the whole regions around a pixel not the pixel itself. Compared with pixel level algorithms, these algorithms have applied higher abstract level of information in term of putting a pixel in the relations with its neighbors. In general, due to the increase of information sources and constrains, these algorithms are better than pixel level algorithms in term of quality, but they are more complicated and computational expensive. Some algorithms of this type also have problem with foreground objects' boundaries, as they cannot classify each pixel in a region. Consequently, some post-processing techniques, such as edge detection, have been used to detect the objects' boundaries. Typical examples of this type are [22], [23] and [24].

The approach in [22] divides the image into small blocks and classify the new incoming blocks basing on the changes in the correlation of pixel colors inside each block. The vector represent for the correlation of pixel colors in the region is created by putting the color values of all pixels into one column. Then, the statistical values of each vector are calculated, and new incoming vectors are classified basing on the Mahalanobis distance between them and those statistical values. To reduce the computational cost, it reduces the vector dimension significantly by using an eigenspace.

Another region level approach was proposed in [23]. The aim of this method is to maintain many competing Hidden Markov Models (HMMs) at each image position to model for suddenly and frequently changing background scene. However, maintaining many HMMs at each pixel needs a large amount of processing and memory space. Thus, the approach divides the images into many 8x8 blocks and utilizes DCT (Discrete Cosine Transform) coefficient values of these blocks, which are provided in JPEG compression form of the images, for its calculation.

Not using information of the block itself, the approach given in [24] even utilizes the information from neighbor blocks for its classification of that block. This method aims to deal with dynamic background scene with small but frequently changing background objects such as waving trees or fluttering flags. Thus, it bases on the co-occurrence property of image variations at neighboring image blocks. Consequently, unlike other statistical or probabilistic approaches which only uses temporal image variation property by observing and collecting data of sequence

of images over time, this approach utilizes the spatial property of the background image itself. It helps to deal with sudden image variations, small but frequent moving background objects and illumination changes.

Image level background subtraction algorithms

Image level algorithms are those that use information at highest abstraction level, such as objects' shapes, categories, or other pre-known knowledge of the environment. However, as stated in [25] as the first principle in background subtraction: *"Semantic differentiation of objects should not be handled by the background maintenance module"*, using this approach only is not efficient enough due to its complexity and large computational power needed.

Combined background subtraction algorithms

There are also combinations of pixel level, region level and image level categories, such as those proposed in [26], [25], etc., in order to balance the tradeoff between quality and time.

The method proposed in [26] can solve many background subtraction problems, such as quick illumination changes, relocation of background objects, initialization with moving objects, and shadows. It solves the problems totally in 3 levels: pixel level, region level and image level. At the pixel level, it utilizes both color information and edge information. At region level, it combines results of color-based segmentation and edge-based segmentation getting from pixel level to decide which

regions are the correct foreground regions. The information of falsely detected regions when using color-based method only is then used to update the model of pixel color. The image level segmentation is applied to fix large errors in color-based segmentation when the result of color pixel based segmentation is unreliable. In this case, the method simply ignores color-based segmentation result, and gives the final decision basing on the results of edge-based segmentation.

The paper [25] also uses three-level approach to solve as many challenges of background subtraction problem as possible. At the pixel level, it maintains background models for each pixel using Weiner filter linear predictor to predict the new pixel color basing on a history of its values. The pixel level can directly solve many problems such as dynamic background scene, gradual illumination changes, and camouflage. The region level refines the classification results of the pixel level. Using inter-pixel relationships, it tries to solve the problem of pixel level approach which cannot detect changes in the interior pixels of a moving homogeneously colored object. The image level solves the sudden illumination changes problem, such as light switching, by maintaining a set of background models and providing an mechanism for automatically switching between them.

4.1.2 A novel framework for real time background subtraction algorithms

The purpose of this framework is to increase the speed of pixel level background subtraction algorithms. However, it can also be applied for small region level

algorithms.

In this part, *foreground pixel* term is used to denote pixel that belongs to image regions of the objects of interest, and *background pixel* term is used to denote pixel that belongs to the background scene.

For a general pixel level background subtraction algorithm, we assume that it has a function *isBackground(x,y,image)* which returns:

- *true* if the pixel of *image* at (x, y) coordinate is a *background pixel*
- $false$ if the pixel of *image* at (x, y) coordinate is a *foreground pixel*

It is obvious that for all background subtraction algorithms the function *isBackground(x,y,image)* to calculate and classify each pixel into background pixel or foreground pixel classes is very computational extensive and time consuming. Consequently, instead of calling *isBackground(x,y,image)* function for all pixels of the image to extract the foreground part, the main idea of this framework is to reduce the number of times to call this function as much as possible.

In order to achieve this, the framework first tries to find the smallest rectangular region of the image containing the foreground objects, with the minimum number of times to call *isBackground(x,y,image)* function. After that, it calls *isBackground(x,y,image)* function again to classify all pixels in that small rectangle only. In this way, the number of times to call *isBackground(x,y,image)* function depends on how large the region of the foreground objects cover the whole image. If the camera is adjusted to view the objects far away, the region of the

foreground objects in the images will be small, and thus, the number of times to call *isBackground(x,y,image)* function is reduced significantly. Normally, the camera is adjusted so that the largest and nearest foreground objects cover about 1/8 of the images. Consequently, even in the worst case with the largest and nearest foreground objects, the background subtraction is speeded up approximately eight times.

Table 4.1 is the pseudo-code of this framework. Generally, the framework has 6 major steps:

1. Test each pixel of the image at coarse resolution level to see whether it is a foreground pixel or not. If it is, the algorithm will proceed to step 2. Otherwise, it keeps on testing the next pixel at coarse level. If all pixels of the image at coarse level are tested, jump to step 5.

2. With the foreground pixel $A(x_A, y_A)$ found at the previous step, find the start point $B(x_B, y_B)$ of the contour of the foreground object region that contains $A(x_A, y_A)$.

3. The framework maintains a list of contours of foreground object regions found during its execution. At this step, the contour start point B found at the previous step will be checked to see whether or not it belongs to one of the contours in the contour list. If there is a contour in the list that contains this start point B already, the framework will jump back to the first step to continue testing with the next pixel. Otherwise, it will continue the 4^{th} step.

Table 4.1: Novel framework for pixel-level background subtraction algorithm.

```
{1}   for each pixel A(x_A, y_A) of image at coarse resolution level
          if isBackground(x_A,y_A,image)
{2}           B(x_B, y_B) = search_for_contour_start_point(x_A,y_A,image)
{3}           if (B ≠ NULL) and (∀C, C ∈ ContourList : B ∉ C)
{4}               newC = trace_new_contour(x_B,y_B, image)
                  add newC to ContourList
              endif
          endif
      endfor
{5}   xmin = +INFINITY; ymin = +INFINITY;
      xmax = 0; ymax = 0;
      for each C ∈ CL
          for each M(xM, yM) ∈ C
              xmin = min(xmin,xM); ymin = min(ymin,yM)
              xmax = max(xmax,xM); ymax = max(ymax,yM)
          endfor
      endfor
{6}   for x= xmin → xmax
          for y= ymin → ymax
              if isBackground(x,y,image)
                  foreground(x,y) = TRUE;
              else
                  foreground(x,y) = FALSE;
              endif
          endfor
      endfor
```

4. Starting from B, all pixels belong to the new contour of this new foreground object region will be found and added to the contour list. After that, it goes

back to the first step to continue testing the next pixel at coarse resolution level.

5. Basing all contours in the contour list, calculate the smallest rectangular region of the image that contains all these contours.

6. Classify all pixels in this rectangular to background pixels or foreground pixels.

Finding the contour start point

The algorithm to find the start point of the contour is presented in Figure 4.2. From a foreground pixel at coarse resolution level, it simply moves upward horizontally to test the next pixel until a background pixel is found or the nearest upper pixel at coarse resolution is reached. If the background pixel is found, the previous foreground pixel right below it is the start point of the contour. Otherwise, in the later case, the start point of this contour and the contour itself have already been added when the framework testing the nearest upper pixel at coarse resolution level.

Tracing pixels along the contour

For tracing the whole new contour starting from the start point found in the previous step, we use the contour tracing algorithm proposed in [27]. Our detailed implementation of the algorithm is presented in Figure 4.3. In general, for a given

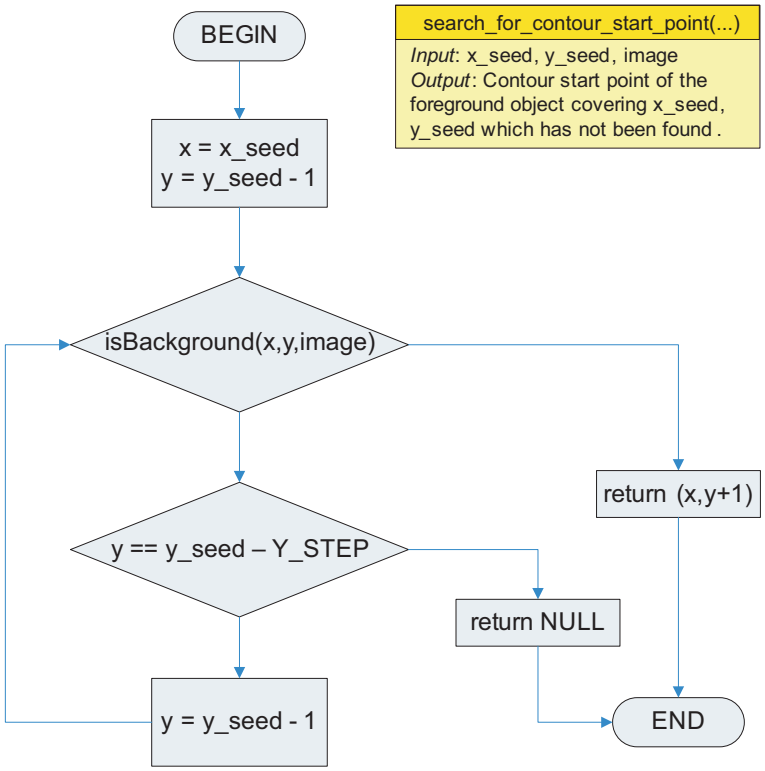

Figure 4.2: Finding the contour start point

contour pixel A, the algorithm will find the next contour pixel by testing all neighboring pixels of A in clockwise order.

As shown in Figure 4.4, eight neighboring pixels of A are numbered from 0 to 7 in clockwise order, starting from A's upper neighboring pixel.

If A is the start point of the contour found in the previous step, the initial searching position is set to 1. The reason is that the neighboring pixel at 0 has already been tested and proved to be a background pixel in the previous step.

If A is not the contour start point and the previous contour pixel P is the n_th neighboring pixel of A, then the initial searching position is set to $(n+2)\ mod\ 8$. This is because pixel at position $(n+1)\ mod\ 8$ must have been visited already. In

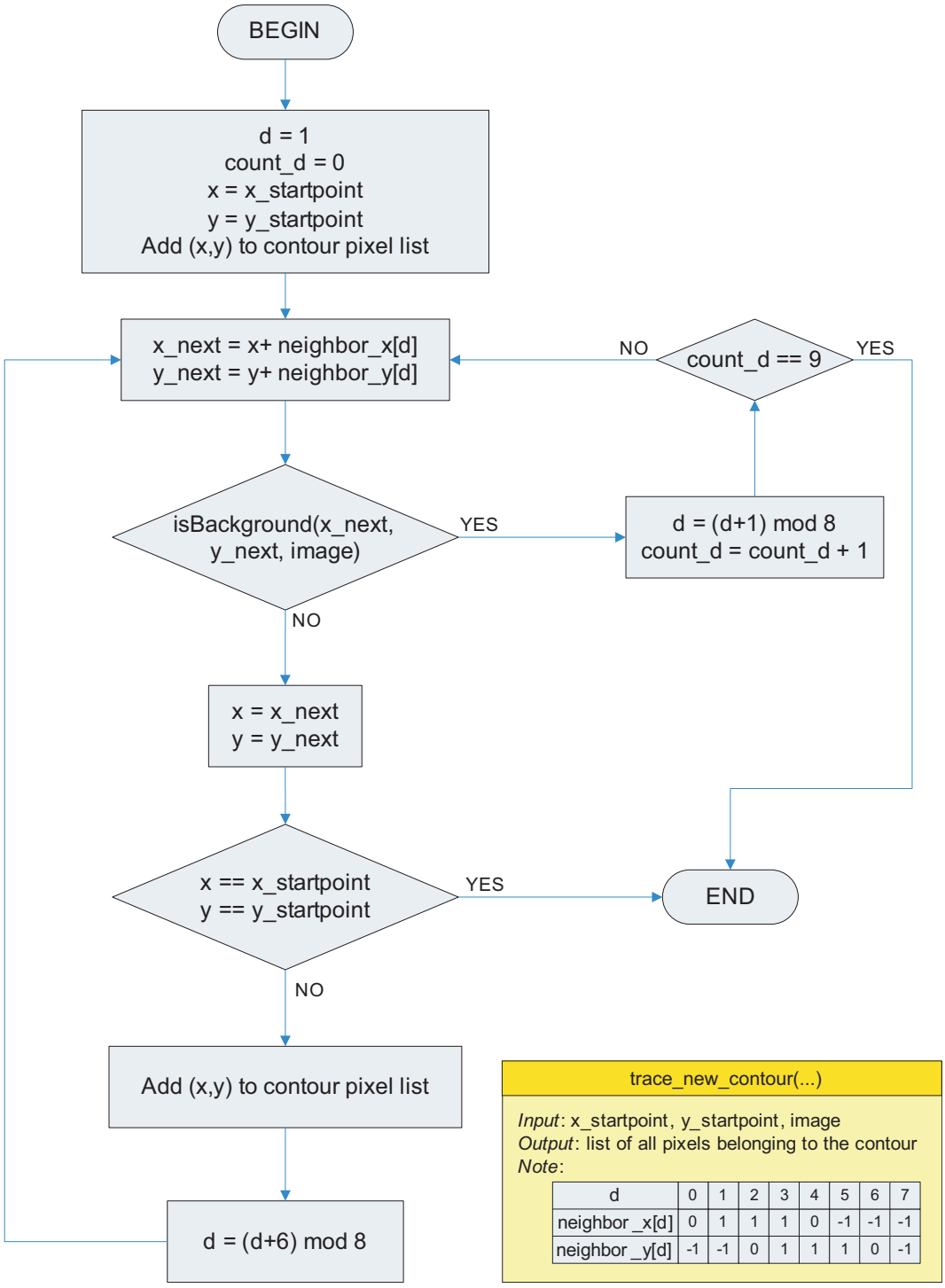

Figure 4.3: Contour tracing algorithm

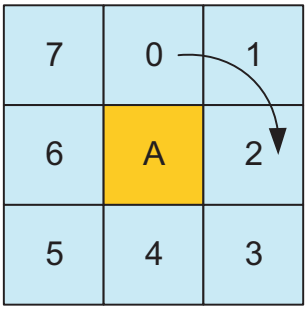

Figure 4.4: Indexes of neighboring pixels

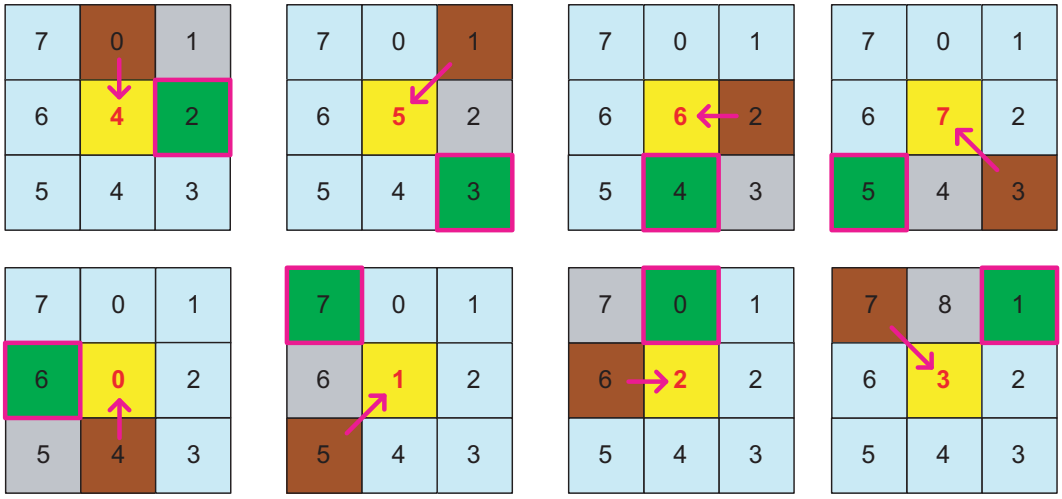

Figure 4.5: Relationship of the previous and current contour pixels, and the initial searching position for the next one

this way, the algorithm keeps on finding all pixels along the contour until it meets the contour start point again.

It is easy to see that if P is the n_th neighboring pixel of A, then A is the $(n-4)\ mod\ 8$_th neighboring pixel of P. Consequently, the algorithm maintains a variable d as the counter to visit all neighbors of the current pixel. After A is found as the neighboring pixel of P, d is equivalent to $((n-4)\ mod\ 8)$. Then, the next value of d as the initial position for next contour pixel is calculated as $(d+6)\ mod 8$,

which equals to $(n + 2)\ mod\ 8$.

Figure 4.5 clarifies the above ideas. It shows all possible cases of the relationship between P (the brown squares) and A (the yellow squares), and the initial position to search for the next contour pixel (the green squares). Note that the red numbers in each yellow squares specify the current value d_0 of variable d, and the numbers in each respective green squares are the next value d_1 of variable d. We can see that $d_1 = (d_0 + 6)\ mod\ 8$.

Filtering function of the framework

The proposed framework also helps to pre-filter some small misclassified regions caused by the specific background subtraction algorithm. Firstly, any foreground region which does not contain any tested pixel at coarse resolution level will not be found. For those regions, their maximum heights and/or widths are respectively smaller than $xStep$ and $yStep$, which values specify the coarse resolution level. Consequently, if $xStep$ and $yStep$ are small enough, those regions are either too thin or too narrow to be considered as correct foreground objects.

Secondly, the contour length is defined as number of pixels belonging to that contour. The contour is added to the contour list only if its length is larger than a threshold. Otherwise, it is not added because its length is so small that it is apparently a misclassified foreground region. In this way, most of the small misclassified foreground regions are filtered out. One special case is that all neighboring pixels of the contour start point are background pixels. Then this contour has 1-pixel

length, and this misclassified pixel is filtered out.

Demonstration of the framework's working mechanism

Figures from 4.6 to 4.16 demonstrate the results of this framework at each step.

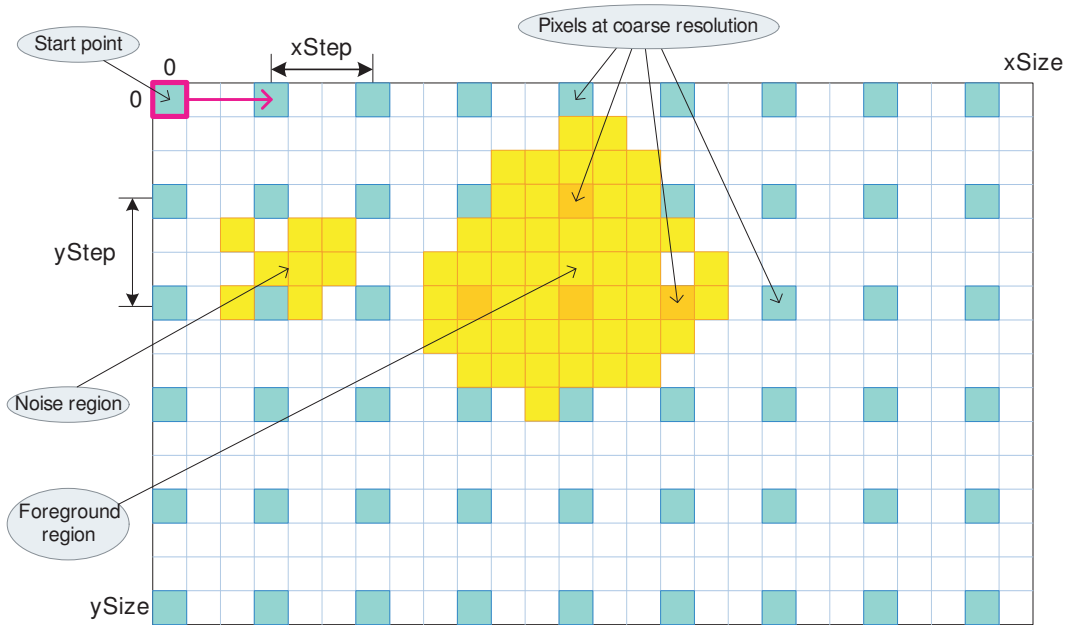

Figure 4.6: Image and its pixels at coarse resolution

4.1.3 Real implementation of background subtraction in 3D Live

The result of visual hull construction in the Rendering module depends largely on the output of background subtraction step. This pre-processing step is one of the most crucial steps to determine quality of the final 3d model. Not only having to produce the correct foreground object, the chosen background subtraction algorithm must be very fast to fulfill the realtime requirement of this system.

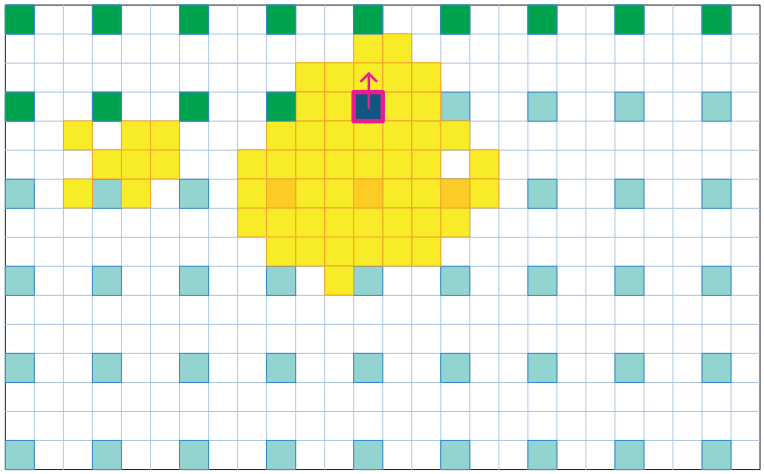

Figure 4.7: Step 1: First foreground pixel at coarse resolution level has been found

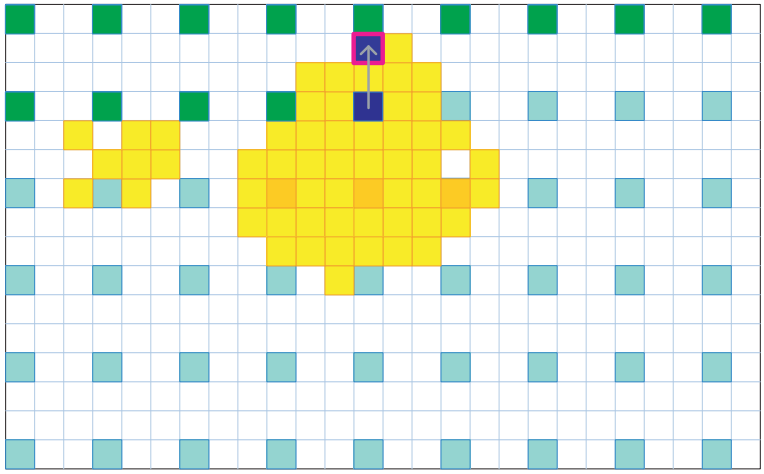

Figure 4.8: Step 2+3: Start point of the contour has been found

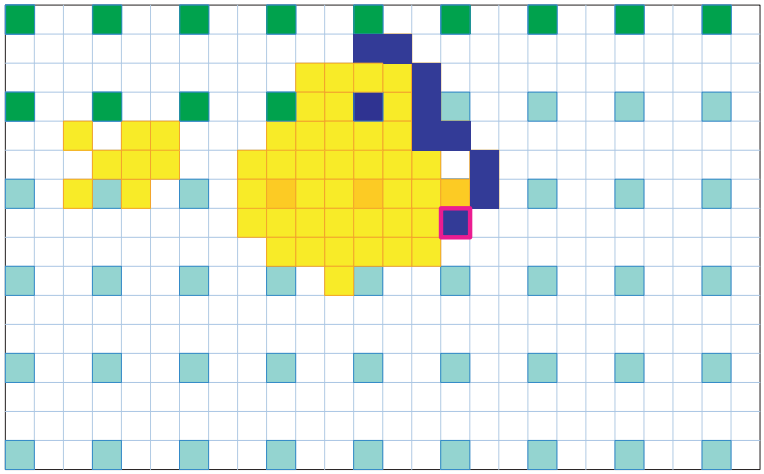

Figure 4.9: Step 4: Contour tracing in progress

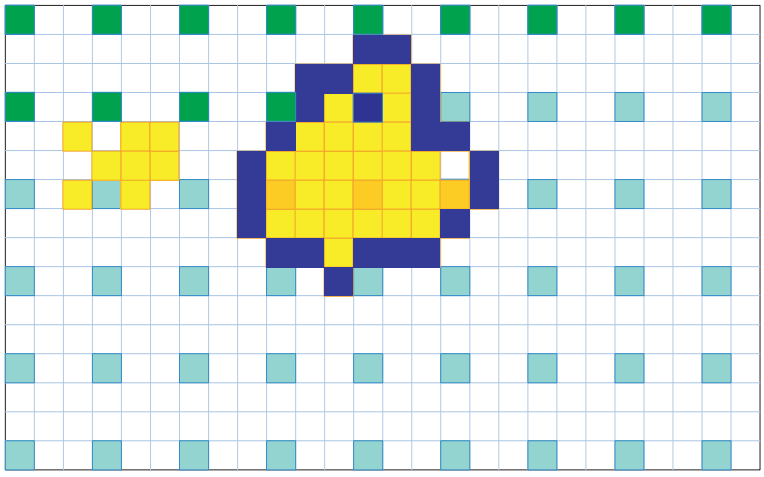

Figure 4.10: Step 4: Finish tracing all pixels of the contour

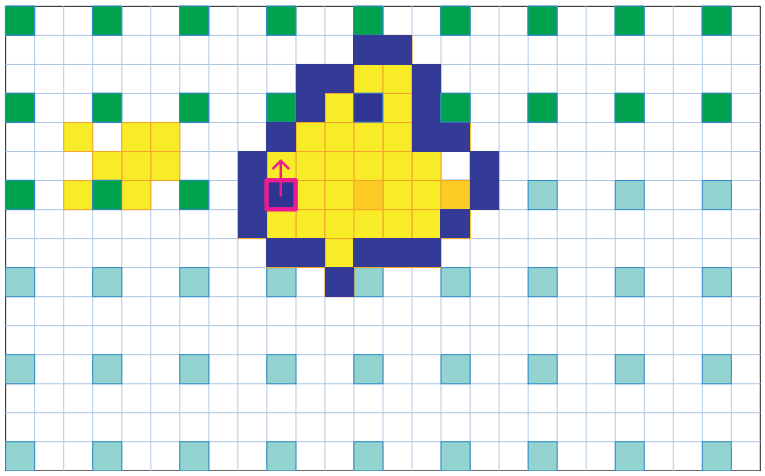

Figure 4.11: Back to step 1: next foreground pixel at coarse resolution level has been found

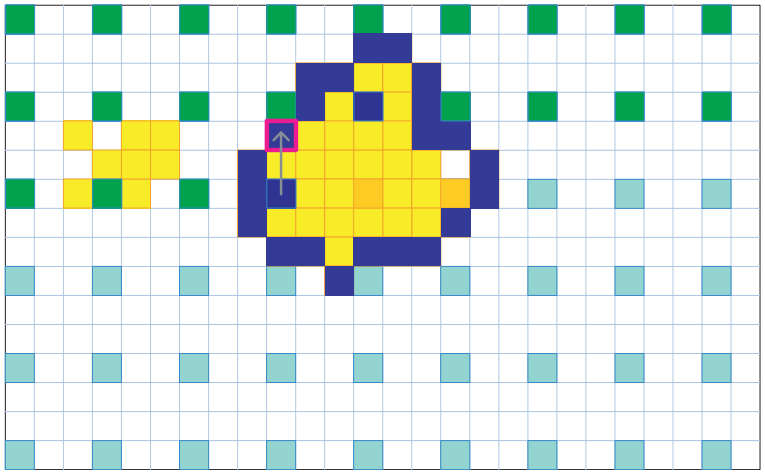

Figure 4.12: Step 2+3: Contour start point has already been in the contour list

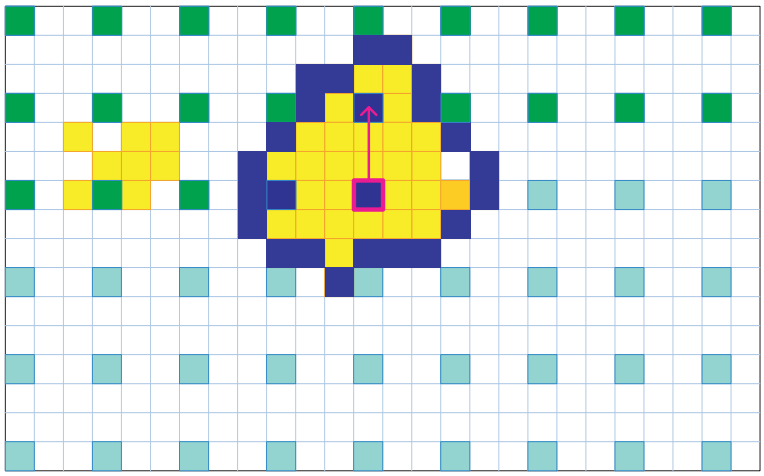

Figure 4.13: Step 1+2: Found next foreground pixel at coarse resolution level, but the contour start point had been processed already when processing the upper foreground pixel at coarse resolution level

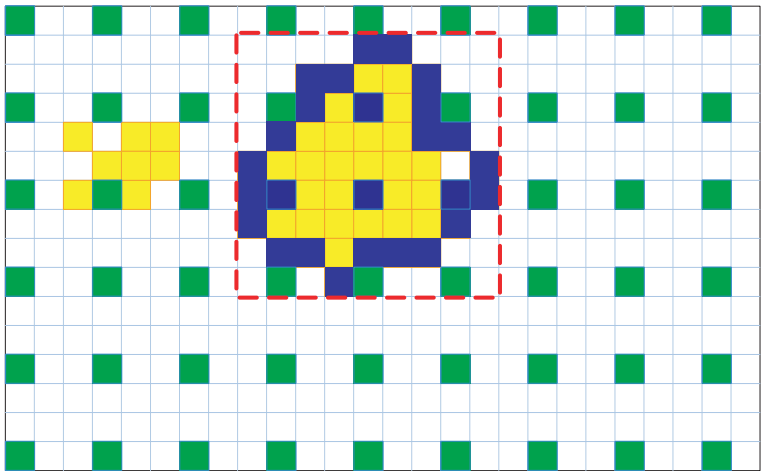

Figure 4.14: Step 5: All pixels at coarse resolution level have been tested. The smallest rectangular has been found.

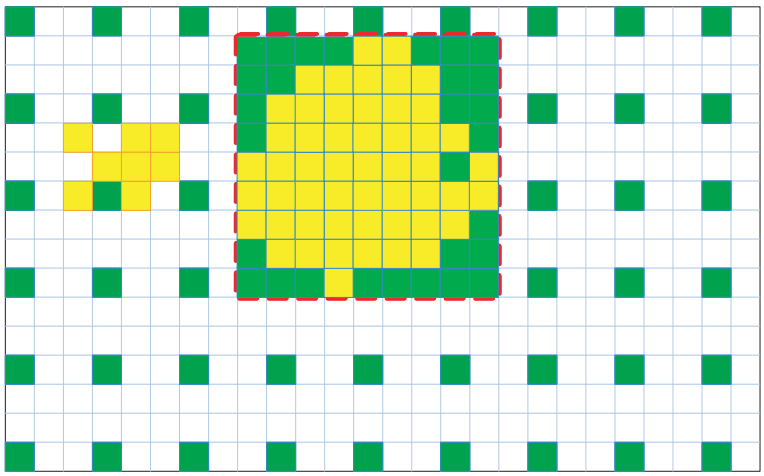

Figure 4.15: Step 5: Classify all pixels in the smallest rectangular

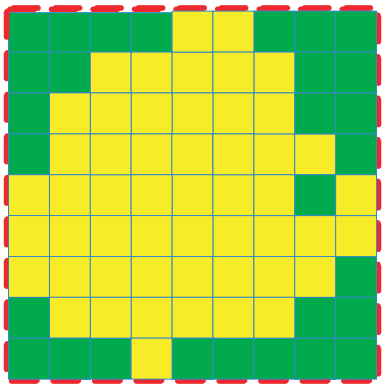

Figure 4.16: Final result: the foreground has been found and the noise was eliminated

Another important requirement to guarantee the good shape of the visual hull is that the background subtraction algorithm must be able to eliminate the shadow caused by the objects.

The proposed framework is fast and general enough to be applied with any pixel level or small region level algorithms. However, to fulfill our needs, we use a modified method based on the scheme of Horpraset [20], which has the good capabilities of distinguishing the highlighted and shadow pixels. However, this algorithm has been modified in our research to reduce the computational intensiveness and optimize for the real time constraints of our system.

The main idea of this method is to learn the statistics of properties of each background pixel over N pre-captured background frames, and obtain the statistical values modeling for the background. The pixel properties to be calculated here are the chromaticity and the brightness which is obtained from a new model of the pixel color. Basing on this, the algorithm can then classify each pixel into *"foreground"*, *"background"*, *"highlighted background"* or *"shadow/shading background"* after getting its new brightness and chromaticity color values. In our application, we only need to distinguish the *"foreground"* type from the rest.

The new color model which separates the brightness from the chromaticity component is summarized in Figure 4.17.

Regarding to Figure 4.17, in the RGB color space , the point $I(i)$ represents the color value of pixel i^{th}, and $E(i)$ represents the expected color value of this pixel, which coordinates $(\mu_R(i), \mu_G(i), \mu_B(i))$ are the mean values of the R, G, B

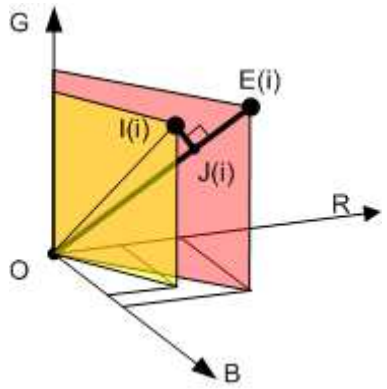

Figure 4.17: Color model

components of this pixel obtained from the learning stage. $J(i)$ is the projection of $I(i)$ on the line $OE(i)$.

The brightness distortion (α_i) and color distortion (CD_i) of this pixel are defined and calculated as [20]:

$$\alpha_i = \frac{J(i)}{E(i)} = argmin_{\alpha_i} \left[\left(\frac{I_R(i) - \alpha_i \mu_R(i)}{\sigma_R(i)} \right)^2 + \left(\frac{I_G(i) - \alpha_i \mu_G(i)}{\sigma_G(i)} \right)^2 + \left(\frac{I_B(i) - \alpha_i \mu_B(i)}{\sigma_B(i)} \right)^2 \right] \tag{4.1}$$

$$CD_i = \sqrt{\left(\frac{I_R(i) - \alpha_i \mu_R(i)}{\sigma_R(i)} \right)^2 \left(\frac{I_G(i) - \alpha_i \mu_G(i)}{\sigma_G(i)} \right)^2 \left(\frac{I_B(i) - \alpha_i \mu_B(i)}{\sigma_B(i)} \right)^2} \tag{4.2}$$

In the above formula, $\sigma_R(i), \sigma_G(i), \sigma_B(i)$ are standard deviations of the i^{th} pixel's red, green, blue values computed in the learning stage. In our version, we assume that the standard deviations are the same for all pixels to make CD_i formula simpler:

$$CD_i = (I_R(i) - \alpha_i \mu_R(i)) (I_G(i) - \alpha_i \mu_G(i)) (I_B(i) - \alpha_i \mu_B(i)) \quad (4.3)$$

Another assumption is that the distributions of α_i and CD_i are the same for all pixel i. With this assumption, we do not need to normalize α_i and CD_i as was being done in the previous work of [20].

These modifications reduce the complexity of the formula and quite drastically decreases the calculation speed from 33ms/frame to 13ms/frame, but produce more small misclassified pixels than the original algorithm. However, these small errors can be easily filtered in the post-processing step.

Actually, the proposed background subtraction framework which finds the foreground object contours also acts as one filtering method. This framework has filtered all small misclassified foreground regions which contour lengths are less than a predefined threshold.

However, the post-processing filtering step is necessary to remove the small misclassified regions inside the smallest rectangular which only contains the foreground objects. There are many filtering methods to process the images after background subtraction. However, although the smallest rectangular region of foreground objects is small enough to guarantee the fast speed, we try to reduce the processing time as much as possible by choosing the simple morphological operators open and close to effectively filter out small misclassified regions.

Figure 4.18: Results of Background subtraction: before and after filtering

4.1.4 Results of the optimized background subtraction algorithms

The quality of the image processing step is shown in the sample results of Figure 4.18. We can see that there are small errors after we subtract the background by our optimized algorithm. In the figure, the small green pixels inside the body is the foreground pixels misclassified as background ones, and the small black pixels outside the body is the background pixels misclassified as foreground ones. However, these errors are completely removed after the filtering step. The speed of this step is only around 15ms/frame. Compared with the non-simplified algorithm, which is 37ms/frame including the filtering step, the optimized algorithm is fast enough for this real time application.

4.2 Data size for real time network constraints

One very important factor is the amount of data to transfer over the network. In order to reach the fastest network speed, the size of data has to be as small as possible. In our system, we try to optimize the data size by using two main following methods:

- Reducing the image size by only storing the smallest rectangular region containing the foreground objects. The implemented background subtraction framework helps to find out the contour of the foreground and bases on this result to calculate the smallest bounding box.

  The size of this smallest rectangular region bounding the foreground objects depends on how close the camera look at the object, and how large the object is. As mentioned in Section 3.2 System setup of Chapter 3, all cameras must be adjusted so that they view the object from a far enough distance to guarantee quality of the visual hull. Consequently, for each camera, the average size of this bounding box of the foreground is normally less than 1/8 the size of the whole image, which is a significant reduction in the data size.

- Using Bayer format [28] with background information encoded to store the images. Instead of using 3 bytes to encode 3 color components Red, Green, Blue for each pixel, we encode the whole image in Bayer format, which costs only 1 byte for each pixel. Moreover, for each pixel, the background information is encoded in the least significant bit of the byte at the position of this

pixel in the Bayer image, value 1 for background pixel and 0 for foreground pixel.

Obviously, this method of storing images leads to some color information lost, however, the lost information is not much. It is first because Bayer format is actually the typical format of the original raw image data received from any CCD-type cameras like Dragonfly. Moreover, the maximum difference between the final background-information-encoded value and the original value at each pixel position in the Bayer image is only 1 unit within the range from 0 to 255. Consequently, the color quality of the output images is still good, as shown in Figure 4.18, and the lost information is trivial, compared with the benefit of reducing much data size, which is at least 3 times smaller than the RGB format with background information encoded.

Chapter 5

3D Live Network Design

3D Live system is a large system with many computers and software modules connected together. Thus, an efficient network design is compulsory to guarantee the completeness of data flow and the fast data transmission speed. Previous version of 3D Live system has a simple network design which uses only TCP protocol to transfer many small data packets in the system. However, TCP has many disadvantages in our situation and makes the system cannot run in real time. Among those disadvantages of using TCP protocol are:

- TCP protocol has a process of optimization of window size data to calculate the optimum window size to transfer data. This process together with its receiver-acknowledgement mechanism makes transferring many small packets of data using TCP protocol consume a lot of time.

- TCP does not support multicast sending. However, for multiple user system, it is preferred to transfer the processed images from Synchronization machine to all Rendering machines at the same time, meanwhile still keep the low network loading.

- It is not necessary that data transferred from Synchronization machine to Rendering machines is guaranteed to be completed. If one frame is lost, the Rendering machines prefer receiving and rendering the next frame than waiting for the lost frame to be re-transmitted which only results in unnecessary longer delay time. Thus, similar to any so-called multimedia streaming system, the reliability of TCP protocol is not necessary here compared with the unreliable but fast UDP protocol.

To transfer image data from Capture Servers machine to Synchronization machine, TCP protocol can be used to guarantee data completeness. However, instead of using TCP protocol to send each frame of each camera as each small packet each time, all data of 3 cameras can be gathered together into a large buffer and the whole buffer is sent at once. This helps to overcome the first disadvantage of using TCP protocol listed above.

For the second and third disadvantages, IP multicast protocol can be the good solution. With IP multicast, data from Synchronization machine can reach all Rendering machines at the same time and the network loading is still optimized. However, IP multicast is only a raw multicast transfer protocol which has no other

advanced services such as packet ordering, sending rate adaptation, data error correction, packet lost recovery, etc. For this reason, RTP and RTCP protocols, which are well-known network protocols for streaming multimedia data (video/audio) over Internet, can be applied.

This chapter will discuss about our designed network architecture for 3D Live system, and the use of IP multicast to transfer data from Synchronization machine to Rendering machines. After that, we will present a design of RTP protocol for 3D Live data together with a simple packet loss recovery mechanism.

5.1 The Network Architecture of Capturing and Rendering

Figure 5.1 illustrates the conceptual components of the distributed capturing and transmission modules.

Each Capture Server machine is connected to 3 cameras each via FireWire (IEEE 1394) cables, which are mounted on the Recording room. The 3 Capturing modules reside on the 3 Capture Server machines and will act upon the instruction of the Synchronization module on Synchronization machine to start capturing.

After all the frames captured from the cameras are processed, they will be gathered together and sent to the Synchronization machine at once following its request. This sending and receiving part using TCP protocol to guarantee the data completeness. After receiving the processed images, the Synchronization module

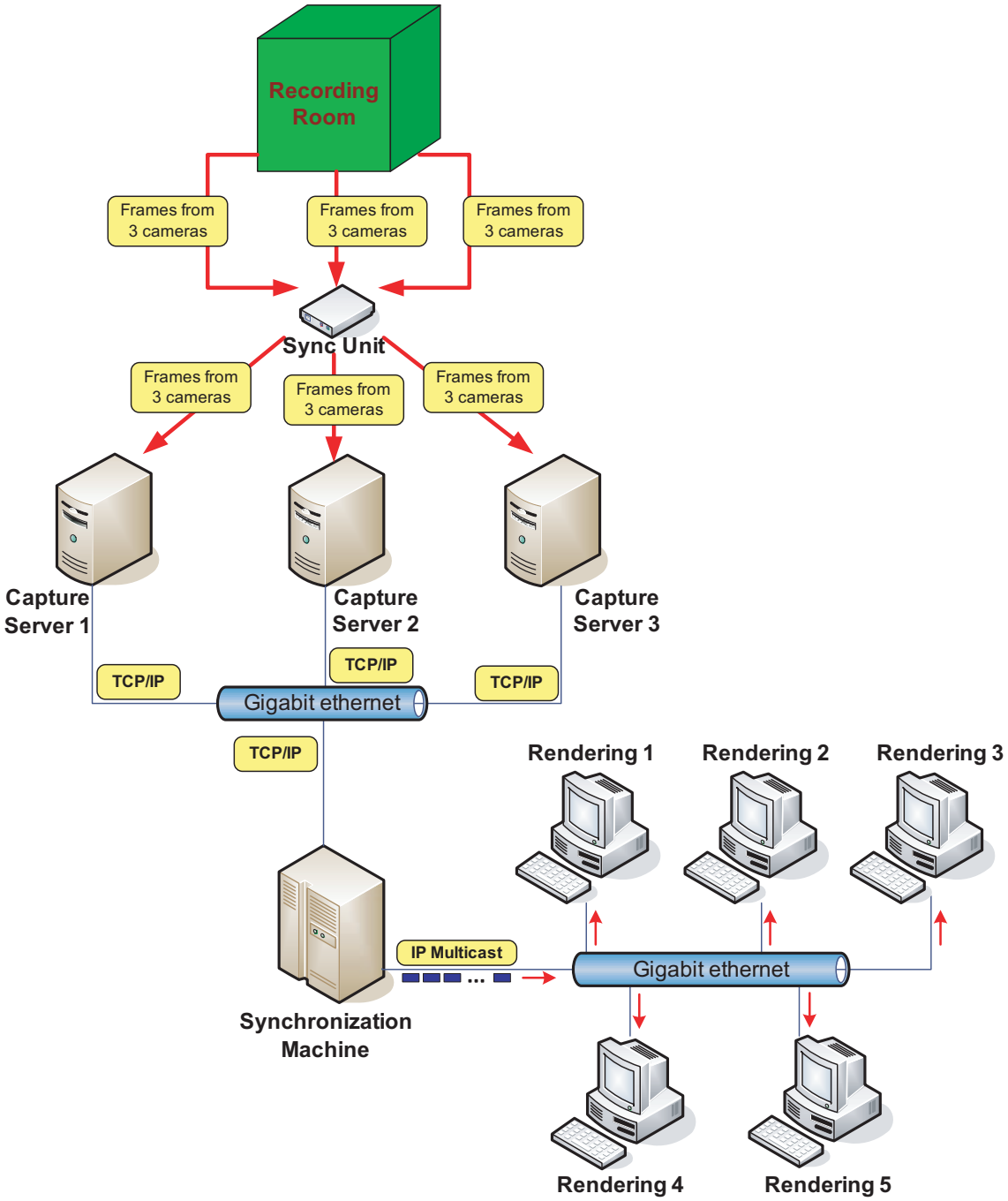

Figure 5.1: 3D Live Network Topology

will synchronize the received frames basing on their timestamp information to make sure that all frames are captured approximately at the same time. Then

these images are transmitted to multiple Rendering computers approximately at the same time using IP multicast or RTP/RTCP protocols specifically designed for 3D Live data.

5.2 Multicasting 3D Live data to Rendering machines

IP multicast is a functionality provided by the transport layer of a network that allows data packets sent by a certain machine to a group of machines to be received by all members of that group simultaneously. This results in a tremendous increase in the efficiency of bandwidth utilization over the conventional unicast.

To *"group"* several machines together, IP multicast a certain non-reserved Class D IP address to specify the group address. These IP multicast addresses are non-reserved addresses ranging from 224.0.0.0 to 239.255.255.255. If some machines want to communicate with others using multicast, they have to join the same group specified by a specific IP multicast address.

To multicast the 3D Live data including nine synchronized foreground images received from the Image Processing and Synchronization modules to all Rendering machines, the large buffer containing the 3D Live data is divided into small 64KB packets and each packet will be sent to the network using multicast. However subsequent tests revealed that if being sent continuously, the packets usually get lost during transmission. The possible reason for this is that IP multicasting is

based on connectionless User Datagram protocol (UDP), which does not guarantee reliable delivery like the connection-oriented protocol TCP, and the packets were lost mainly because of network congestion at the Gigabit switch.

If the ratio of packet lost is small, for example, about 1% to 5% in our experiments, it is still acceptable in our multimedia real time streaming context, as the lost of one rendered frame does not much affect users' visual experiences. However, the network congestion normally causes a very large number of packet lost, which is, for example, around 70% to 80% in our experiments when we send the UDP packets continuously. Consequently, we try to get rid of the network congestion problem by adding the delay time between two consecutive packets sent. In our real implementation with Gigabit switch, this delay time is set at 5ms and there is no packet lost in the normal case. This 5ms value is set by experiment and can be varied from this system configuration to others. This delay causes a small increase of the sending rate, however, as the total delay from capturing to rendering is around 2 seconds, the result is still acceptable.

5.3 RTP protocol for streaming 3D Live data

Real-time Transport Protocol (RTP) is specially designed to deliver real time media data above the unreliable transport layer. It provides a general open framework for delivering any types of real time media data and needs to be modified specifically for each particular media data format before use. Its peer, RTP Control

Protocol (RTCP), is designed to monitor the quality of service like the reception quality report, and to provide other services such as participant identification, or information to synchronize media streams, etc. RTP and RTCP are well specified in RFC 1889.

Due to the lack of advanced services such as packet ordering, data error correction, etc., the IP multicast protocol described above is not suitable for streaming the media data over the network with complex structure like the Internet. In those situations, RTP and RTCP protocols, which are well-known network protocols for streaming multimedia data (video/audio) over Internet, need to be applied. In this section, we will discuss about the modifications of RTP packet format needed to enable streaming of 3D Live data with RTP.

5.3.1 RTP packet format for 3D Live data

The general format of RTP packet for any media types is depicted in Figure 5.2. Some important fields are summarized as follows:

- V (2 bits): specify version of RTP protocol
- P (Padding bit, 1 bit): specify whether Padding is added at the end of the packet. Padding is not part of the payload data.
- M (Marking bit, 1 bit): specify special packet in the packet stream. For example, in case that multiple packets need to be used to deliver 1 frame, the special first packet can be specified using this bit.

- PT (Payload Type, 7 bits): specify format of the payload data. Each specific media data format is assigned a particular PT number so that the receiver can know what type of encoding scheme of the data stream is, and can use the corresponding decoder to play back the stream. For example, value 8 specifies for PCMA Audio stream, 32 specifies for MPEG I/II Video stream, or 26 specifies for Motion JPEG Video stream respectively.

- Sequence Number (16 bits): identify packet. It is used by receiver to restore the packet sequence and detect packet loss. Sequence Number of the first packet of the data stream is initiated randomly.

- Timestamp (32 bits): specify the time when this data was sampled. It is used by the receiver to schedule the playout of the media data. It is also used to synchronize different data streams coming from different source.

- SSRC Identifier (Synchronization Source Identifier, 32 bits): specify source that generated this media stream.

- Payload data: raw data or encoded data of the media stream used to playout the media.

- Padding: additional bits for the completeness of RTP packet, due to the fixed packet size constraint.

Instead of simply dividing 3D Live data into small packets and sending those packets of raw 3D Live data using IP multicast, we can fit that data into RTP

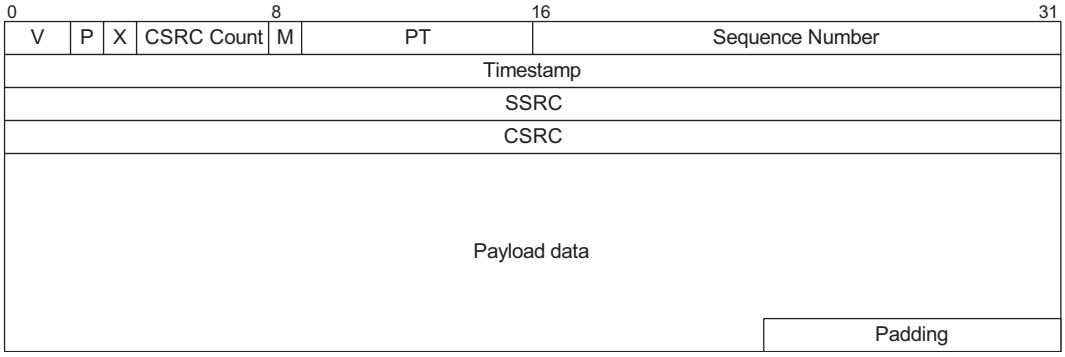

Figure 5.2: Format of general RTP packet

packets before sending them. After processed by Image Processing module, each foreground image of each camera will be divided into small packets with suitable size, and RTP header will be added to form RTP packets as presented in Figure 5.3 and 5.4.

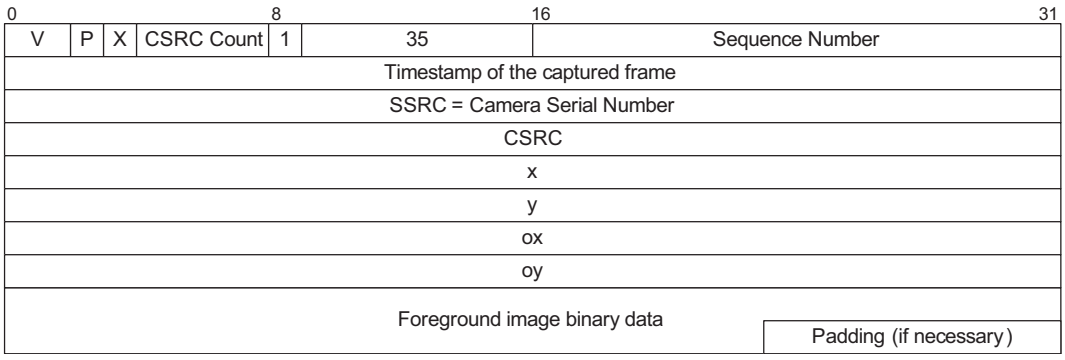

Figure 5.3: Format of RTP packet for 3D Live data - first packet of frame

As can be seen in those Figures, each camera is considered as each media streaming source, and its serial number is used as Synchronization Source identifier (SSRC) field in the packet format. When re-combining those packets into the foreground image, Rendering module can base on the packets' SSRC to know from

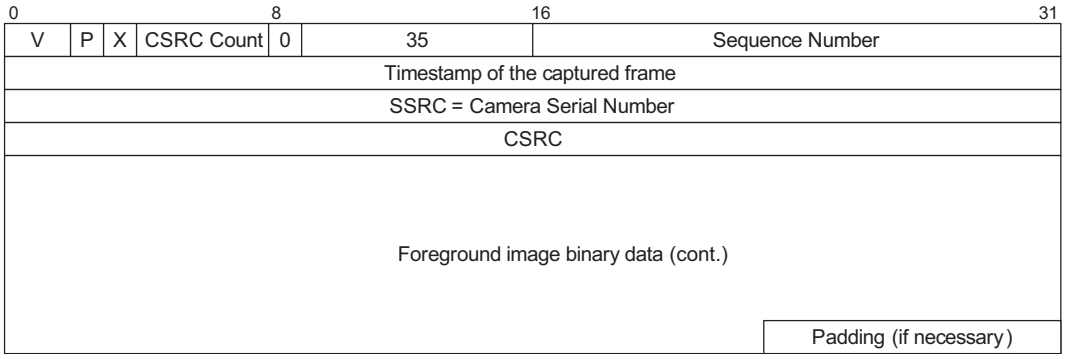

Figure 5.4: Format of RTP packet for 3D Live data - subsequent packets of frame

which camera that the image is captured. Moreover, the Sequence Number field of the RTP packet stream containing foreground images generated from a specific camera is initiated randomly for the first packet corresponding to that camera and increased by 1 for every following packet.

The timestamp of the foreground image, obtained from the first 4 bytes of the captured image as mentioned in 3, is used for the timestamp of the RTP packets delivering that foreground image data. This is exactly the time when this image is captured by the camera. Moreover, due to the synchronization mechanism of Sync Units and Dragonfly cameras, all timestamps of the captured images from all cameras are generated by the same timer system, which means that RTP timestamp of all packets from all images and cameras are generated synchronously by the same clock.

CSRC and CSRC Count fields in RTP packets are Contributing Source Identifiers and Contributing Source Count, which are not necessary in our context. Payload Type (PT) field can be any PT number that has not been assigned for

any previous known video/audio encoding standard. We choose 35, which is an unassigned number, as the value for PT field in our case.

In case that a set of multiple RTP packets needs to be used to deliver one foreground frame, M bit is set to 1 for the first RTP packet (Figure 5.3) and to 0 for all other subsequent packets (Figure 5.4) in this set. For each foreground frame to be sent, the first RTP packet contains the foreground image header in the first 16 bytes of the packet's payload data (Figure 5.3). This header is four 4-byte integer numbers which contain important information to recover the correct foreground image:

- x, y: size of the smallest rectangular containing the foreground part of the original image. They also specify the size of foreground binary data transferred by RTP packets. This information is used by the receiver to calculate the number of RTP packets used to transfer the foreground image: $n = \frac{x \times y}{RTP\_packet\_size - RTP\_packet\_header\_size}$
- ox, oy: offset of the smallest rectangular containing the foreground part relative to the origin of the whole image. They specify coordinate of the top-left corner of the smallest rectangular containing the foreground part.

Normally, the size of the last portion of foreground data to be sent by the last RTP packet transferring that foreground frame does not fit the RTP payload data size. In this case, Padding will be used at the end of the RTP packet, and the Padding bit (P) is set to 1 for this RTP packet. For other RTP packets which use

no Padding, the Padding bit is set to 0.

5.3.2 Receiving RTP packets of 3D Live data and packet lost recovery

With the design of RTP packet format for 3D Live data specified above, the Rendering modules will receive 9 streams of 3D Live RTP packets from 9 cameras. Basing on SSRC field, the receivers can distinguish packets coming from different cameras. For each camera, the first packet of a foreground frame is recognized basing on the marking bit M. After that, number of packets to transfer this foreground frame is calculated basing on the header of the payload data field of this first packet. Received packets are re-ordered basing on their Sequence Number, and after that, the Payload data field of packets with the same Timestamp, which belong to the same foreground frame, are extracted and grouped together to re-form the original foreground image.

After 9 foreground images are re-created successfully from 9 RTP streams, Rendering module can easily use them to generate the 3D Live image. However, in case that some packets are lost due to some bad network condition, which means foreground frames of some cameras cannot be re-generated by the receiver, we will apply a simple packet lost recovery scheme, namely replacement scheme. The last previous successfully re-generated foreground frames will be used to replace for the current lost ones. Difference in the capturing time of the current frame and the replaced frame may cause some error in the final rendered image, however, because

the time difference is not too large, the result is still acceptable.

Chapter 6

Rendering

Our rendering algorithm used in this system is a new development over our previous algorithm which is described in [1]. To improve the speed and quality, we introduce new ways to compute visibility and blend color in generating images for novel viewpoints. In this chapter, the main algorithm will be first briefly described. After that, improvements for speed and quality will be presented.

Our algorithm proceeds entirely on a per-pixel basis. We denote the desired image, the *"virtual camera image"* and its constituent pixels *"virtual pixels"*. The virtual camera can be determined by taking the product of the (head mounted) camera calibration matrix and the estimated transformation matrix. Given this 4 x 4 camera matrix, the center of each pixel of the virtual image is associated with a ray in space that starts at the camera center and extends outward. Any given distance along this ray corresponds to a point in 3D space. We calculate an image based depth representation by seeking the closest point along this ray that is inside

the visual hull. This 3D point is then projected back into each of the real cameras to obtain samples of the color at that location. These samples are then combined to produce the final virtual pixel color. In summary, the algorithm must perform three operations for each virtual pixel:

- Determining the depth of the virtual pixel as seen by the virtual camera.
- Finding corresponding pixels in nearby real images.
- Determining pixel color based on all these measurements.

We briefly describe each of these operations in turn.

6.1 Determining Pixel Depth

The depth of each virtual pixel is determined by an explicit search starting at the virtual camera projection center and proceeding outward along the ray corresponding to the pixel center (see Figure 6.1). Each candidate 3D point along this ray is evaluated for potential occupancy. A candidate point is unoccupied if its projection into any of the silhouettes is marked as background. When a point is found for which all of the silhouettes are marked as foreground, the point is considered occupied, and the search stops.

Using this method, we can generate the visual hull very efficiently. One problem with visual hull is that the geometry it reconstructs is not very accurate. When photographed by only a few cameras, the scene's visual hull is much larger than the

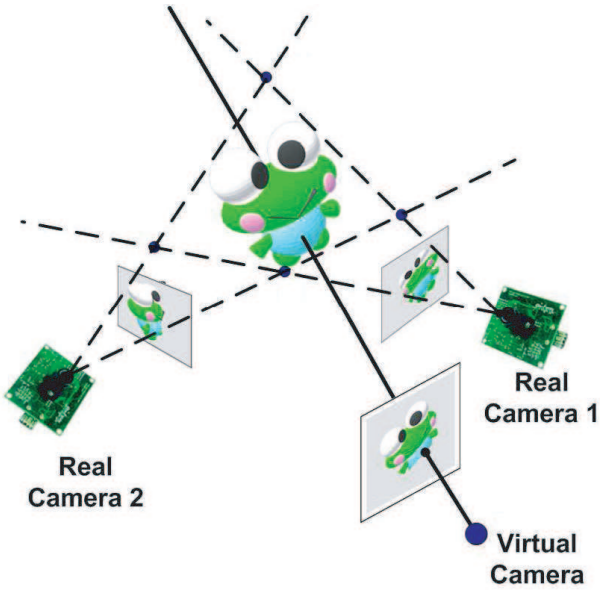

Figure 6.1: Novel View Point is generated by Visual Hull

true scene [29]. One well-known improvement for visual hull which have been discussed in [30], [31], [32], [29], [33], and [34] is to utilize color constraint. Although, using this constraint, we can generate "photo-hull" which is a better approximation than visual-hull, the rendering speed will be decreased significantly and thus not suitable for real-time applications. Alternatively, we reduce the errors of visual hull by using more cameras and a larger recording room.

6.2 Finding Corresponding Pixels in Real Images

The resulting depth is an estimate of the closest point along the ray that is on the surface of the visual hull. However, since the visual hull may not accurately represent the shape of the object, this 3D point may actually lie outside of the object surface. Hence, care needs to be taken in choosing the cameras from which

the pixel colors will be combined. Depth errors will cause incorrect pixels to be chosen from each of the real camera views.

To minimize the visual effect of these errors, it is better to choose incorrect pixels that are physically closest to the simulated pixel. So the optimal camera should be the one minimizing the angle between the rays corresponding to the real and virtual pixels. For a fixed depth error, this minimizes the distance between the chosen pixel and the correct pixel. We rank the cameras proximity once per image, based on the angle between the real and virtual camera axes.

We can now compute where the virtual pixel lies in each candidate cameras image. Unfortunately, the real camera does not necessarily see this point in space - another object may lie between the real camera and the point. If the real pixel is occluded in this way, it cannot contribute its color to the virtual pixel. In the previous versions of this research, we increase the system speed by intermediately accepting points that are geometrically certain not to be occluded. However, this geometrical information does not always provide true occlusion. As we can see in Figure 6.4, in the left image, we still can see false shadows of two hands over the body. These false hand shadows are generated because these parts of the body are occluded from the reference cameras by the two hands, but the geometrical-based method cannot detect it. To achieve better results, in this new version, we introduce a new method to compute occlusion.

6.3 Determining Virtual Pixel Color

After determining the depth of a virtual pixel and which cameras have an unoccluded view, all that remains is to combine the colors of real pixels to produce a color for the virtual pixel. In the previous research, we took a weighted average of the pixels from the closest N cameras, such that the closest camera is given the most weight. This method can avoid producing sharp images that often contain visible borders where adjacent pixels were taken from different cameras. However, there are still some errors along the edge of the silhouette. In next section, we propose a new method to blend color which can overcome this problem.

6.4 New Rendering Algorithm for Speed and Quality

6.4.1 Occlusion Problem

As said above, one of the main issues of this algorithm is the occlusion problem. In order to compute visibility, one basic approach is searching in 3D space. To determine if a point A is visible from one camera, we can simply search point by point from A toward the center O of this camera. If any point in this ray belongs to the visual hull, A is considered to be invisible from this camera (Figure 6.2).

Instead of brute-force searching in 3D space, [29] proposed a more efficient way which only need to step along epipolar lines. However, with this method, we still

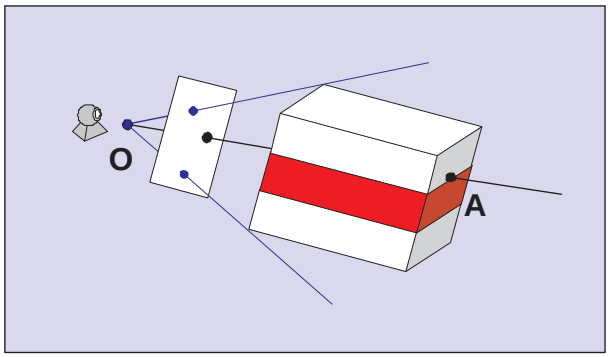

Figure 6.2: Example of Occlusion. In this figure, A is occluded from camera O.

need to search on all captured images. To further increase the speed, we introduce a new method which only requires searching on one captured image.

To compute visibility, Matusik introduced a novel algorithm which can effectively reduce 3D visibility computation to the 2D visibility computation [3]. The main idea of this algorithm can be illustrated in Figure 6.3. In this figure, camera K is chosen so that the projection Q of P on this camera lies on the edge of silhouette. This algorithm bases on the fact that the 3D point P has to be visible from the camera K if on the image plane of one camera K, the 2D point Q is visible from the epipole E (the projection of the center of projection of camera K onto the image plane of camera J). In their paper, they use this algorithm to determine visibility of each face of the visual hull, but we apply it to compute visibility of each point of the image-based visual hull. Our algorithm can be summarized as follows:

To determine if point P is visible from camera K, the three following steps will be processed:

1. Find one camera J where the project Q of P lies on the edge of the silhouette.

2. Find the epipole E of camera K on the image plane of camera J

3. If there is any foreground pixel lying on the line connecting point Q and point E, i.e. Q is occluded from point E, then P will be considered to be occluded from camera K. Otherwise, P will be consider to be visible from camera K.

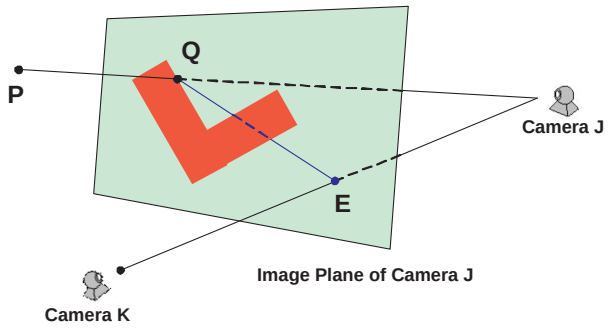

Figure 6.3: Visibility Computation: since the projection Q is occluded from the epipole E, 3D point P is considered to be invisible from camera K

Using this algorithm, we can avoid 3D searching while still able to detect occlusion whenever it happens. However, this algorithm is over-conservative [3]. It never considers a point visible if this point is occluded, but sometimes it considers a point occluded which is in fact visible. As a result, some points in visual hull will be computed to be occluded from all cameras, which leads to holes in the results. To compensate for this, whenever a point is computed to be invisible from all cameras, we do not accept that but use the previous version to recompute visibility. The negative effect of this, for some points, we need to run both methods, but

normally there are only few points like that. Thus, it does not affect the overall speed in any significant way.

Figure 6.4 shows example rendering results. In the left image, we use geometrical information to compute visibility while in the right, we use the above described visibility computing algorithm. As one can see, in the image on the left, there are false shadows of two hands over the body while there is not in the image on the right.

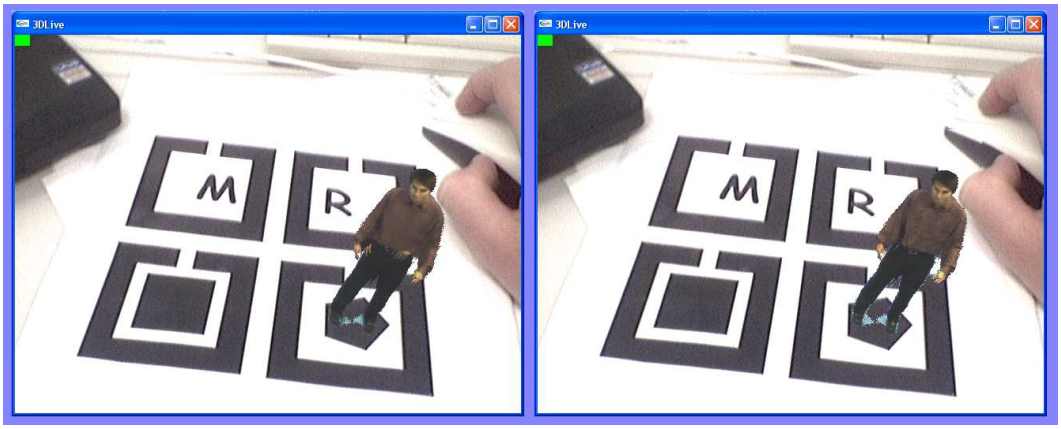

Figure 6.4: Rendering Results: In the left image, we use geometrical information to compute visibility while in the right, we use our new visibility computing algorithm. One can see the false hands appear in the upper image.

Table 6.1 shows the frame rate we can achieve with our algorithm. All three visibility algorithms: 3D searching, geometrical-based and our new algorithm are tested. We also tested with two different resolutions: 320 x 240 and 640 x 480. As we can see, our new method is much faster than 3D searching method. With this new algorithm, we can achieve 23 fps at 320 x 240 and 11 fps at 640 x 480,

Table 6.1: Rendering Speed

Image size	**3D searching**	**New algorithm**	**Geometrical-based**
320 x 240	7 fps	23 fps	27 fps
640 x 480	3 fps	11 fps	13 fps

while with 3D searching, it is only 7 fps and 3 fps respectively. Compared with the geometrical-based method, the new method is a little slower but it provides better results.

6.4.2 New method for blending color

The second improvement is a new method to blend color for visual hull. Most of current shape-from-silhouette algorithms use the angles between the desired view and reference views to decide the weights for blending. However, it can cause errors along the edges of foreground images, because background subtraction usually generates errors in these areas. For example, in Figure 6.5, if we base on the angles of cameras, point A will get color from camera 2 which is closer angle to the novel viewpoint. However, the projection of A to camera 2 is at the edge of the silhouette which usually contains some errors due to the background subtraction.

To address this issue, we utilize a technique from image-mosaicking.In this subject of image-mosaiking, to reduce visible artifacts that is, to hide the edges of the component images, one can usually use a weighted average with pixels near

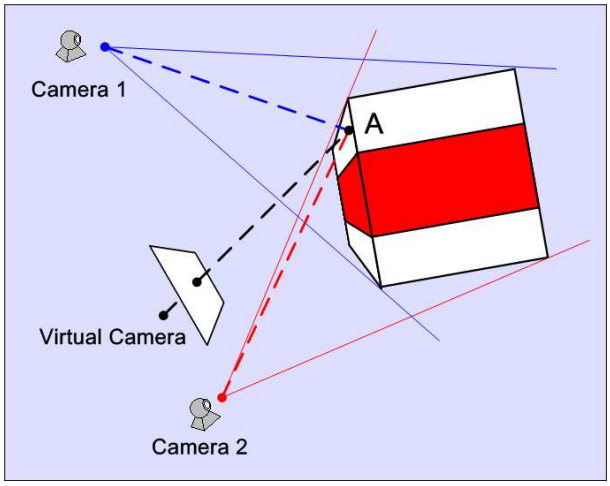

Figure 6.5: Example of Blending Color

the center of each image contributing more to the final composite [34]. Similar to this idea, in our algorithm, to determine the color of the virtual pixel, we take a weighted average with pixels near the center of each silhouette having higher weights. Thus, in Figure 6.5, if we use this blending method, A will get color from camera 1, where the projection of A is closer to the center of silhouette. This new blending method makes the visual hull smoother along the edges of silhouettes.

One problem with this blending method is that it requires more memory and time to store and calculate the weights, as each pixel of each reference images got different weights. To increase the speed, instead of computing these pixel weights during rendering, we calculate them during the image processing process. In such way, we can run this calculation on three different computers, each in charge of images captured from three cameras. This will triple the speed. Thus, for each captured image, the Image Processing module will calculate the weights for each pixel and then pass these weights for the rendering module.

Figure 6.6 shows one set of images from nine cameras and their corresponding pixel weights. The brighter one pixel is, the higher weight it gets. Figure 6.7 shows two rendering results. The left is rendered with camera weights while the right with pixel weights. As we can see, using pixel weights, the result is better and smoother, especially along the edge of silhouettes.

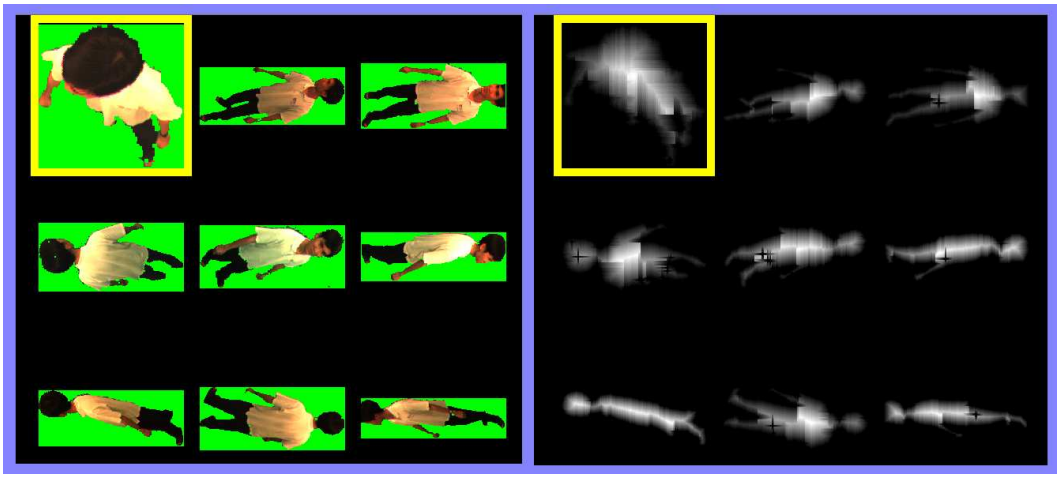

Figure 6.6: Original Images and Their Corresponding Pixel Weights

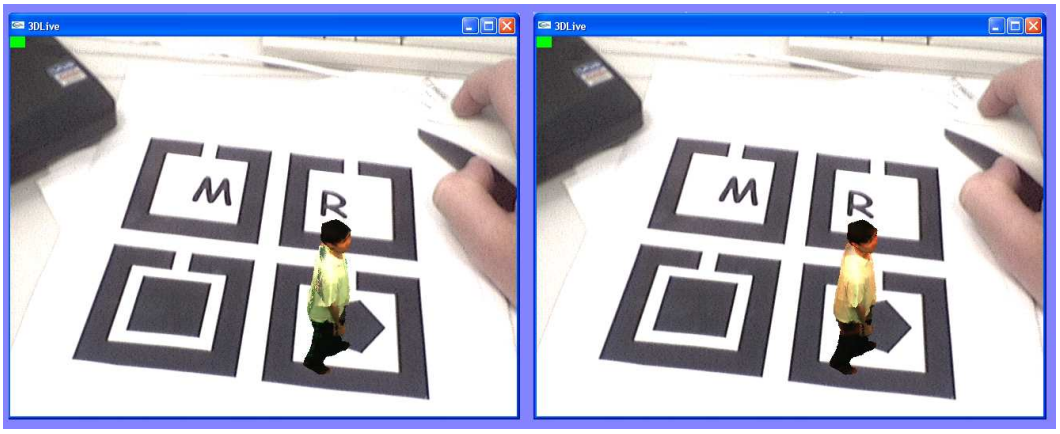

Figure 6.7: Rendering Results: The right is with the pixel weights algorithm while the left is not. The right image shows a much better result especially near the edges of the figure.

Chapter 7

Magic Land - a 3D Live Mixed Reality Application in Art and Entertainment

With the abilities of capturing, sending, regenerating the 3D images of live humans and objects in real time and displaying this objects' 3D images in the augmented reality environment, 3D Live technology has many applications in various fields.

The first obvious application is a three-dimensional video-conferencing and collaboration system, which is much better than the traditional 2D video-conferencing system in term of communication benefits. It is because the 3D images displayed in real environment can fully represent non-verbal communication such as gestures, which the traditional 2D system cannot. Moreover, using the 3D system, users not only can arrange markers representing several collaborators about them to cre-

ate a virtual spatial conferencing space, but also can potentially conference from any location, and thus, the remote collaborators become part of any real world surroundings, potentially increasing the sense of social presence.

Another application of 3D Live system in education and entertainment is an augmented book, in which a different fiducial marker is presented on each page, and associated with each is virtual content consisting both of 3D graphics and a narrator who was captured in our system. Others applications of this system in training, entertainment, computer games, etc. can be seen in [1].

The remaining of this chapter will fully describe a novel application of 3D Live in art and entertainment. This system, named Magic Land, is the cross-section where art and technology meet. In technology viewpoint, it is a combination and demonstration of latest advances in human-computer interaction and human-human communication: mixed reality, tangible interaction, and 3D Live technology. In artistic viewpoint, it aims to introduce tangible approaches of dealing with mixed reality content to artists of any discipline. These approaches, which allow artists to manipulate the mixed reality content intuitively and easily by using cups, was also presented in [35] for a city planning application.

Another main purpose of Magic Land system is to bring to all users a new special kind of human self reflection and human-human interaction. In this system, users can tangibly pick up themselves or their collaborators and watch them in 3D form encountering with other virtual objects. In order to allow users to manipulate their own 3D recorded images in mixed reality environment, this version of Magic Land

does not fully exploit the *"live"* capturing feature of 3D Live, but instead utilizes the fast processing and rendering algorithms for fast 3D Live record and playback features. However, another version of Magic Land, which can be built easily for live capture and live viewing, is discussed further in section **??**. The artistic intention and motivation of the project will also be discussed further in section 7.3.

7.1 System Concept and Hardware Components

Magic Land is a mixed reality environment where 3D Live captured avatars of human and 3D computer generated virtual animations play and interact with each other.

The system includes two main areas: recording room and interactive room. The recording room is where users can have themselves captured into live 3D models which will interact in the mixed reality scene. This room, which has nine Dragonfly cameras mounted inside, is a part of the 3D capture system described above. After the user gets captured inside the system, she can go to the interactive room to play with her own figure.

The interactive room consists of three main components: a Menu Table, a Main Interactive Table, and five playing cups. On top of these tables and cups are different marker patterns. A four cameras system (ceiling tracking system) is put high above the Main Interactive Table to track the relative position of its markers with the markers of the cups currently put on it. The users view the virtual scenes

and/or virtual characters which will be overlaid on these tables and cups via the video-see-through HMDs with the Unibrain cameras mounted in front and looking at the markers. The Main Interactive Table is first overlaid with a digitally created setting, an Asian garden in our case, whereas the cups serve as the containers for the virtual characters and also as tools for users to manipulate them tangibly. There is also a large screen on the wall reflecting the mixed reality view of the first user when he/she uses the HMD. If nobody uses this HMD for 15 seconds, the large screen will change to the virtual reality mode, showing the whole magic land viewed from a very far distant viewpoint.

An example of the tangible interaction on the Main Interactive Table is shown in the Figure 7.1. Here we can see a user using a cup to tangibly move a virtual panda object (left image) and using another cup to trigger the volcano by putting the character physically near the volcano (right image).

The Menu Table is where users can select the virtual characters they want to play with. There are two mechanical push buttons on the table corresponding with two types of characters: the human captured 3D Live models on the right and VRML models on the left. Users can press the button to change the objects showed on the Menu Table, and move the empty cup close to this object to pick it up. To empty a cup (trash), users can move this cup close to the virtual bin placed at the middle of the Menu Table. In the Figure 7.2, in the left image, we can see a user using a cup to pick up a virtual object, at the edge of the table closest to the user are two mechanical buttons. In the right image we can see the augmented

Figure 7.1: **Tangible interaction on the Main Table**: (Left) Tangibly picking up the virtual object from the table. (Right) The trigger of the volcano by placing a cup with virtual boy physically near to the volcano.

view seen by this user. The user had selected a dragon previously which is inside the cup.

After picking up a character, users can bring the cup to the Main Interactive Table to play with it. Consequently, there will be many 3D models moving and interacting in a virtual scene on the table, which forms a beautiful virtual world of those small characters. If two characters are close together, they would interact with each other in the pre-defined way. For example, if the dragon comes near to the 3D Live captured real human, it will blow fire on the human. This gives an exciting feeling of the tangible merging of real humans with the virtual world. As an example of the interaction, in the Figure 7.3, we can see the interaction where the witch which is tangibly moved with the cup turns the 3D Live human character which comes physically close to it into a stone.

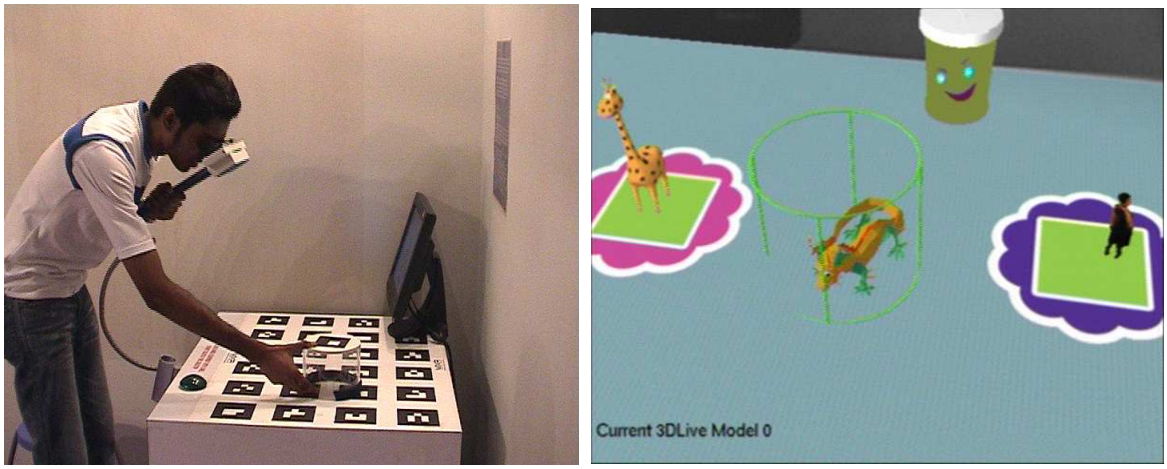

Figure 7.2: **Menu Table**: (Left) A user using a cup to pick up a virtual object. (Right) Augmented View seen by users

Figure 7.3: **Main Table**: The Witch turns the 3D Live human which comes close to it into a stone

7.2 Software Components

As shown in Figure 7.4, the software system of Magic Land consists of 5 main parts: 3D Live Recording, 3D Live Rendering, Main Rendering, Ceiling Camera Tracking, and Game Server. Beside these parts, there is a Sound module that produces audio effects including background music and interactive sounds for the whole system. This section will present the 5 main parts in turn.

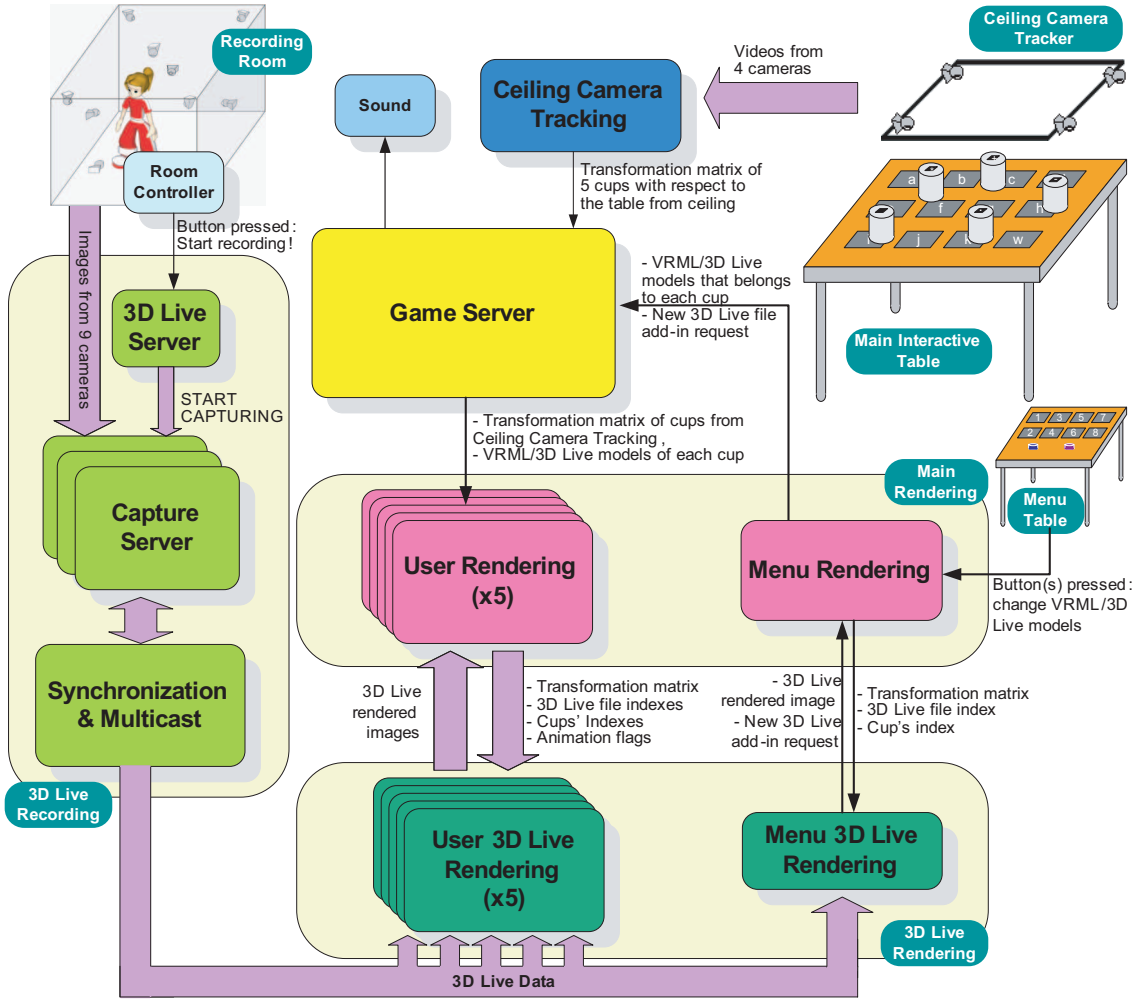

Figure 7.4: System Setup of Magic Land

7.2.1 3D Live Recording and 3D Live Rendering

In this system users can record their live model for playback. After going inside the recording room and pressing a button, the user will be captured for 20 seconds. The captured images are then processed and sent to all 3D Live Rendering modules. The 3D Live Recording and 3D Live Rendering parts are the recording capturing 3D Live system described in the previous parts of this thesis. However, unlike the live version which sends the processed images of nine cameras immediately for each frame, the recorded version sends all the processed images of all the frames captured in 20 seconds at a time. After that, IP multicast is used to send the data to all User 3D Live Rendering and Menu 3D Live Rendering modules of the 3D Live Rendering part. This helps to utilize bandwidth of the network as well as to ensure that all the receivers finish receiving data at the same time.

Capture Server

Three copies of this module are put on 3 capture server machines. As presented in 3D Live system, these modules are responsible for getting images from the cameras, doing background subtraction on these images and sending the results to 3D Live Server module when they receive the corresponding requests. The interaction between Capture Servers and 3D Live Server modules is specified in Figure 7.5.

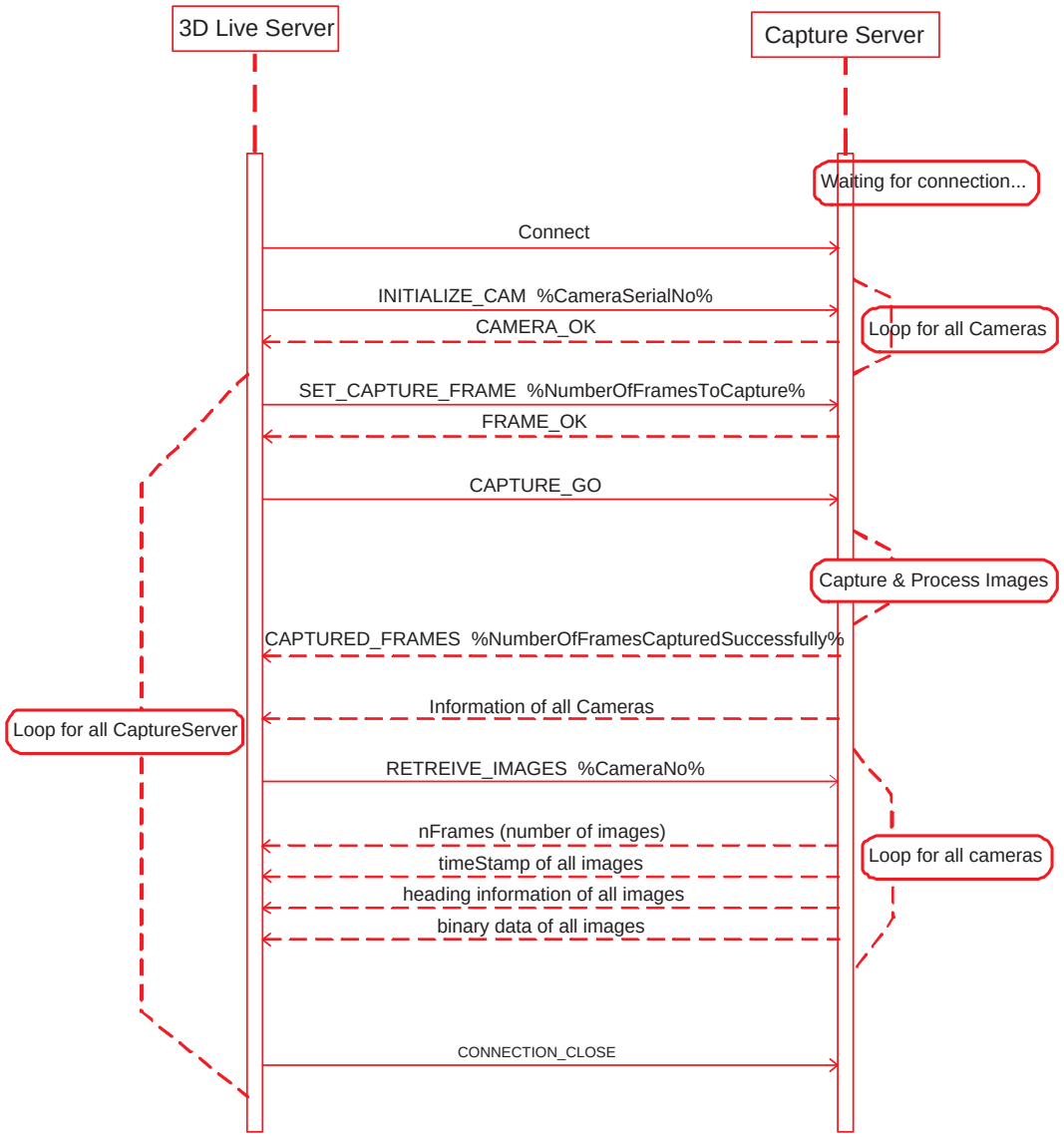

Figure 7.5: The interaction between Capture Server and 3D Live Server modules

For each camera, each Capture Server consists of 2 threads running in parallel. One thread is responsible for capturing and getting images from the cameras and store them into a buffer on RAM memory. The other thread, named Background subtraction, will get images from this buffer, extract the foreground parts, and store them into another buffer. The Background subtraction thread needs to get

the background parameters, which have been learnt and saved to the parameter files for each respective camera. Note that all foreground images of each camera are stored in one buffer corresponding to that camera. After that, when 3D Live Server module requests the foreground images of each camera, instead of sending each image one by one, the whole foreground image buffer corresponding to each camera will be sent at one time. This helps to reduce the communication time significantly, as TCP protocol, for congestion control, has to find the optimal TCP Window size for each time it sends the message. By gathering all images into one big buffer and sending it at once, the number of times for TCP to find the optimum Window size is reduced significantly.

3D Live Server

This module is the intermediary connecting the Room Controller module, 3 Capture Server modules and 3D Live Rendering part. Firstly, it listens to "START-CAPTURE" command from the Room Controller. After the users go inside the recording room and press the button to start capturing, the Room Controller will process all abnormal cases, for example, that the door has not been closed when the button is pressed, or the button is pressed 2 times, etc. In normal cases, Room Controller will send 'START-CAPTURE" command to 3D Live Server. The whole interactions between Room Controller and 3D Live Server is presented in Figure 7.6.

Secondly, after receiving "START-CAPTURE" command from Room Controller, 3D Live Server will start the recording process by sending messages to

Capture Servers. The interaction between Capture Servers and 3D Live Server has been represented in Figure 7.5. After receiving all the foreground images, and combining them to a 3D Live file, 3D Live Server will inform Room Controller that the capturing is successful, and wait for Room Controller sending back the "SUCCESSFUL" or "FAIL" signal to it. The later signal is to specify whether the capturing is failed due to the unintentional opening of the door during the capturing process.

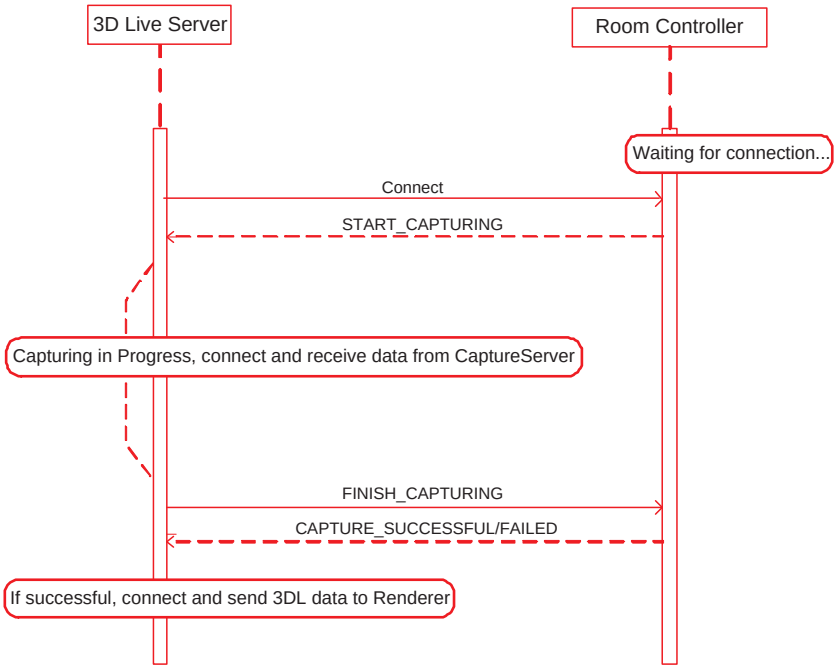

Figure 7.6: Interaction between 3D Live Server and Room Controller modules

If the capturing and processing stages finish successfully, 3D Live Server will send messages to all modules in 3D Live Rendering part, asking them to receive the new 3D Live file. IP multicast protocol is then used to send this file to all 3D Live Rendering modules in order to save the transmission time and guarantee that

all Rendering modules will receive data at the same time.

The interaction between 3D Live Server and 3D Live Rendering modules is specified in Figure 7.7. When combining foreground images received from Capture Servers into one 3D Live file, 3D Live Server stores this file into a large buffer in RAM memory. After receiving signals from all 3D Live Rendering modules notifying that they are ready to receive the new 3D Live file, 3D Live Server will divide the large 3D Live file buffer into many small 64KB packets, and multicast each packets on the Gigabit network. Each 3D Live Rendering module will receive those packets, and then restore the correct 3D Live file.

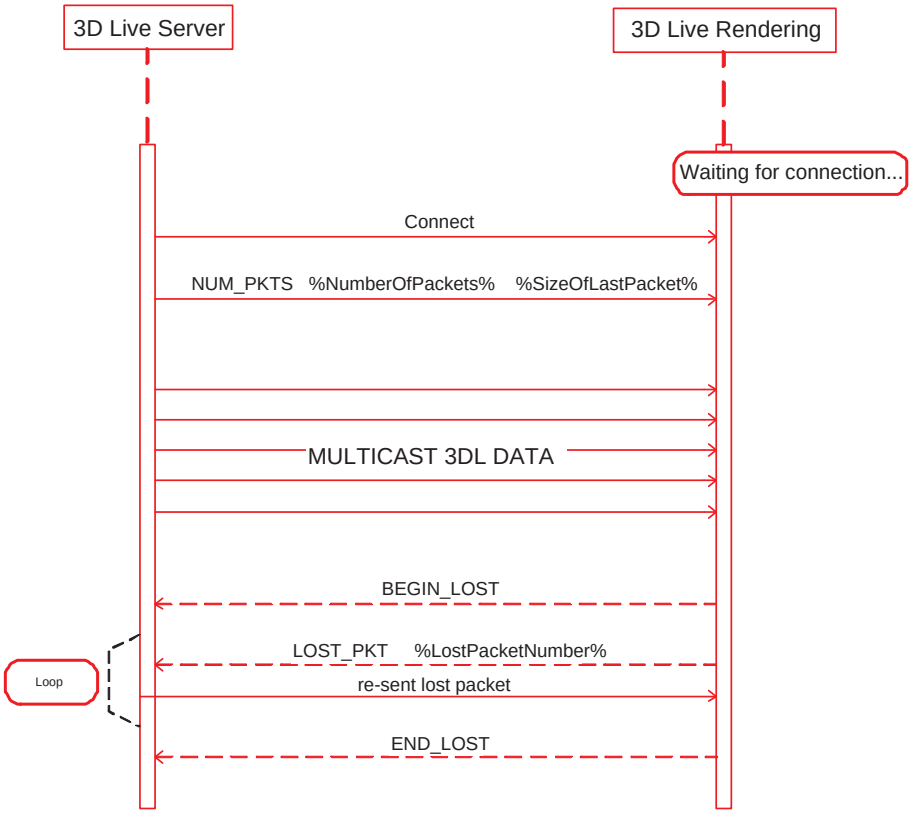

Figure 7.7: Interaction between 3D Live Server and 3D Live Rendering modules

3D Live Rendering

In each 3D Live Rendering module, there are two threads running simultaneously. The first thread, Multicast Receiver, is responsible for receiving multicast packets from 3D Live Server and combining them together to make a complete 3D Live file. The second thread, 3D Live Renderer, is in charge of generating the novel view point of the 3D Live characters for each transformation matrix received from its corresponding Rendering module of the Main Rendering part.

7.2.2 Main Rendering

The Main Rendering part includes a Menu Rendering module and five User Rendering modules, corresponding with the Menu 3D Live Rendering and five User 3D Live Rendering modules of 3D Live Rendering part. These Rendering modules track the users' viewpoints, and render the corresponding images to the users. First, they obtain images from the Unibrain cameras mounted on the users' HMDs, track the marker patterns and calculate the transformation matrix relating the coordinates of these markers with the coordinate of the camera. After that, basing on the transformation matrix, each module will render the image and output the result to the corresponding HMD. All virtual characters and their animations are also rendered basing on the positions of the markers on their corresponding cups. If a cup contains a 3D Live character, the Rendering module will send the transformation matrix, the cup's index and the 3D Live character's index to its corresponding 3D Live Rendering module. After that the Rendering module will receive back the

rendered 3D Live image from its corresponding 3D Live Rendering module.

Especially, the Menu Rendering module also handles the users' inputs when they press the buttons on the Menu Table, or when they use the cups to select and remove virtual characters. All augmented reality rendering, virtual character interactions and user tangible interactions are developed basing on ARToolkit [12], and the similar tangible interaction techniques using cups presented in [35].

7.2.3 Ceiling Camera Tracking

The Ceiling Camera Tracking module, developed using ARToolKit [12], receives images from four cameras of the Ceiling Camera Tracker system put above the Main Interactive Table. It tracks the markers of the table and cups, and calculates the transformation matrices of the cups relative to the table from top view. After that, it sends these matrices to the Game Server.

The Ceiling Camera Tracker system is essential as it sends the cups' positions to the Game Server and the Game Server will use this information to determine if there should be interactions between the models on the cups. Another purpose of this system is that when the user's camera on the HMD is not able to detect the marker on the cup due to the orientation of the cup, the model on the cup can still be rendered basing on the position tracked by these four cameras above.

7.2.4 Game Server

Last but not least, Game Server is the heart of the system, which links all the modules together. It receives and forwards information from the Ceiling Camera Tracking, Menu Rendering and User Rendering modules. This Game Server coordinates and synchronizes what every user has in their cup in terms of type of the character and its animation, position and orientation. First of all, it receives the camera tracking data from the Ceiling Camera Tracking module and determines the interaction between the characters inside the cups, basing on the distances between cups. After that, it forwards this interaction information to the User Rendering and Sound modules so that these modules can render the respective animations and produce the corresponding interactive sound. The ceiling camera tracking data is also forwarded to the User Rendering modules for usage in the case that the users's camera lost the tracking of their cups' marker. When the users select a new character, the Game Server also receives the new pair of cup-character indexes from the Menu Rendering and forwards to all the User Rendering modules to update the change.

7.3 Artistic Intention

Magic Land demonstrates novel ways for users in real space to interact with virtual objects and virtual collaborators. Using the tangible interaction and the 3D Live human capture system, our system allows users to manipulate the captured 3D

humans in a novel manner, such as picking them up and placing them on a desktop, and being able to "drop" a person into a virtual world using users' own hands. This offers a new form of human interaction where one's hands can be used to interact with other players captured in 3D Live models.

The artistic aspect of this installation introduces to artists easy, tangible and intuitive approaches in dealing with mixed reality content. The main challenge of the project is to create a new medium located somewhere between theater, movie and installation. The outcome of the project is an infrastructure that gives artists new opportunities to transport audiovisual information and encourage artists of any discipline to deal with those new approaches.

We can perceive Magic land as an experimental laboratory that can be filled by a wide range of artistic content, which is only limited by the imagination of the creators. To watch the scene from above with the possibility of tangible manipulation of elements creates a new form of art creation and art reception that generates an intimate situation between the artist and audience.

The project itself brings together the processes of creation, acting and reception in one environment. These processes are optimized to the visitors experience in order to better understand the media and lead to a special kind of self reflection. The recording area plays the role of the interface between human being and computer. It is also a special experience for the users to watch themselves acting in 3D on the interactive table from the external point of view like the "Bird in the sky". In Figure 7.8 are two bird's eye views of this system.

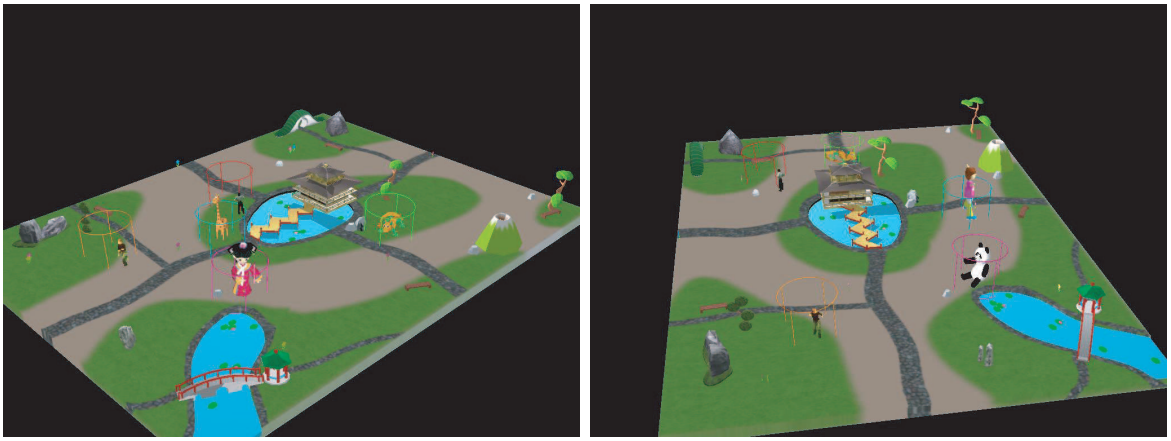

Figure 7.8: **Main Table**: The bird's eye views of the Magic Land. One can see live captured humans together with VRML objects

7.4 Magic Land's Relationship with Mixed Reality Games

Nowadays, computer games have become a dominating form of entertainment due to their higher level of attractiveness to game players. There are some superior advantages which make computer games more popular than traditional games. Firstly, it attracts people by creating the illusion of being immersed into imaginative virtual world with computer graphics and sound [36]. Secondly, the goals of computer games are typically more interactive than that of traditional games, which brings players stronger desire to win the game. Thirdly, usually designed with the optimal level of information complexity, computer games can easily provoke players' curiosity. Consequently, computer games intrinsically motivate players by bringing them more fantasy, challenge and curiosity, which are the three main elements con-

tributing the fun in games [37]. Moreover, compared with many traditional games, computer games are also easier to play at any individual's preferred location and time.

However, the development of computer games has often decreased their physical activities and social interactions. Addressing this problem, growing trends of nowadays game, especially mixed reality games, are trying to fill in this gap by bringing more physical movements and social interactions into games while still utilizing the benefit of computing and graphical systems.

A typical VR game CAVE Quake [38] increases the player's sense of 3D space by surrounding them with life-sized 3D virtual world, instead of constraining them within a limited 2D screen. However, CAVE Quake players still in lack of physical movement, tangible interactions and social communications.

AR2 Hockey [39], an air-hockey AR game in which users use real mallet to play with a virtual puck on a real table, enhances physical interactions and social communication, but does not utilize the graphical power of computer systems.

AquaGauntlet [40] is another AR game in which several players gather in a small place with some physical egg-shape objects to shoot computer-generated creatures superimposed onto the real scene as if they came from these egg-shape objects. This game enhances physical interactions and social communication, and also utilizes the graphical power of computer system. However, players of AquaGauntlet, as well as AR2 Hockey, still have limited movement and little interaction with the physical space (as they must stand in a fairly constant location).

Another embodied computing based mixed reality game which also enhances physical interactions and social communication is Touch-Space [41]. This game is carried out in the physical world with a room-size space where two players will collaboratively finish some tasks and then rescue a princess in castle controlled by a witch. This game provides different levels of interaction in different environments: physical environment, augmented reality, and virtual reality. However, all these interactions are limited in a room-size space and only for two users.

Pirates! [42] and Human Pacman [43] are two typical outdoor mixed reality games aiming for enhancing physical activities and social interactions as much extent as possible. Pirates! uses handheld computers and proximity-sensing technology to make real world properties, such as locations or objects, important elements of game mechanics. Meanwhile, in Human Pacman, the player who acts as "Pacman" wearing a wearable computer and an HMD goes around the physical game space to collect cookies, where as other player acting as 'ghost' will find and touch to kill the Pacman. There are two other players acting as Pacman's and Ghost's helpers sitting inside offices, using computer's graphical information to search their enemy's locations in order to help their partners. These games are very successful in term of enhancing physical interactions and social communications, however, they have not utilized fully graphical power of computing system to create an appealing imaginative virtual world. Pirates is played on a PDA screen which does not allow a 3D mixed reality experience. Human pacman requires quite heavy and bulky wearable computers and equipment.

Compared with the above typical AR/VR games, as an indoor mixed reality and tangible interaction game, Magic Land exploits physical tangible interaction, social interaction and also utilizes 3D graphics rendering to create an attractive imaginative virtual world. Moreover, the act of putting 3D images of real human beings in to that inventive world and making them new characters of the game story is unique in game context. Most importantly, Magic Land is a kind of "free play" game [44], in which players are free to use their imagination and creativity to design the game story and rules. Thus, as mentioned before, the game story and rules is not fixed but depends on players' imagination and decision.

Table 7.1: Comparison of Magic Land with other mixed reality games

Games	**A**dvantages	**D**isadvantages
CAVE Quake	Significantly increase players' sense of 3D space by fully immersing them into a 3D virtual world. Provide beautiful graphics and interesting game story.	Very limited physical movement. No tangible interaction and social communication.
AR2 Hockey	Provide 3D mixed reality experience and tangible interaction with virtual object. Enhance social communication.	Limited physical movement and tangible interaction. No attractive 3D graphics of virtual world.
AquaGauntlet	Provide 3D mixed reality experience, tangible interaction and nice 3D graphics of virtual characters. Enhance social communication.	Limited physical movement and tangible interaction.
Touch Space	Tangible interaction with virtual object, enhance social communication, nice graphical virtual characters in mixed reality world. Different levels of interaction in different environments: physical environment, augmented reality, and virtual reality.	Limited physical movement and number of players.
Pirates!	Provide physical movement and social interaction to great extent.	Limited tangible interaction and graphical virtual characters. No 3D mixed reality experience.
Human Pacman	Provide physical movement and social communication to large extent. Enhance 3D mixed reality experience and tangible interaction.	Physical movement is slightly limited due to wearable computer and HMD.
Magic Land	Provide varied tangible interaction with virtual objects, beautiful 3D mixed reality virtual scene and characters, and social interactions among players. *Players can be captured and become new characters encountering with other virtual characters in mixed reality world.*	Not fully provide physical movement like outdoor games such as Pirates! and Human Pacman.

Chapter 8

Conclusion and Future Work

8.1 3D Live

To summarize, a complete novel system for real-time capturing and rendering 3D images of live subjects in a mixed reality environment has been described in this thesis. We believe that this is a significant step towards the goal of perfect "telepresence" for remote collaborations in the near future.

The hardware and software issues, together with the new and novel improving algorithms and methods to speed up the system and obtain good quality have been discussed in detail. Chapter 3 has presented the hardware and software architecture of the whole system thoroughly and discussed roughly about its new developments compared with the previous system prototype. Those new developments in image processing, networking and 3D live rendering are detailed in the subsequent chapters. In Chapter 4, we have presented many techniques to decrease

the image processing speed and optimize the data size for real time network constraints significantly. Especially, a novel general framework which can be applied for any pixel level or small region level background subtraction has been proposed in Chapter 4 to reduce the image processing time as much as possible. Chapter 5 has discussed about our new network design with the application of IP multicast and RTP/RTCP protocols so as to optimize the data transferring speed and the network loading but still guarantee the good quality of service at the same time. After that, Chapter 6 has described our improvements in the real time rendering algorithm to make it faster and generate the better final results. All the system developments presented in this thesis have really made the system become real time, which has not happened before with the previous system prototype.

However, the quality and speed of 3D Live system can be improved more in the future. Firstly, the speed of image processing stage can be further optimized by parallelizing the background subtraction algorithm, making it run in systems with multiple CPUs to increase the speed more significantly. Secondly, the networking stage can be further optimized by implementing a more efficient error recovery scheme in case of packet lost. Finally, the quality and speed of the rendering stage can be improved also by parallelizing the rendering algorithm so that it can run faster and produce higher resolution results in multiple CPU platforms. We also aim to develop a new real time 3D re-construction algorithm to re-construct the whole 3D model of the captured object instead of just generating its instant image at the current virtual viewpoint. Such an algorithm will be very useful in some

other applications of 3D Live technology which needs to detect the collision of the captured object with other 3D objects in the mixed or virtual reality environment.

8.2 Magic Land

3D Live system has many applications in various fields, such as communication, remote collaborations, education, military, art and entertainment. Focusing on interactive media, in Chapter 7, this thesis has presented Magic Land system, which is a novel application of 3D Live in mixed reality art and entertainment. This system integrates and demonstrates many technologies in human computer interaction: mixed reality, tangible interaction and 3D live collaboration and communication. It plays an important role in the development of mixed reality art and entertainment, as it opens not only a new form of art creation and reception but also a new type of mixed reality games which enable players' participant and submergence totally in the virtual/mixed reality world. This type of games also allows players to control the game story freely themselves using their creativity and imagination. Especially, results of the survey on Magic Land's users presented in Chapter A have revealed some important issues and emphasized the effectiveness of 3D Live, mixed reality, and tangible interaction in human computer interaction.

In the future, a new version of Magic Land can be developed by exploiting the "real time" capability of 3D Live technology, in which, outside players can see on the Main Interactive Table the 3D images of the one who is being captured inside

the room in real time. Instead of sending all the processed images of all the frames captured in 20 seconds at a time, this version uses RTP [45] and IP multicast to stream the processed images to all User 3D Live Rendering modules immediately for each frame. To guarantee continuous rendering, User 3D Live Rendering modules will buffer these images for a number of received frames before generate the 3D images inside one of the special cup on the Main Interactive Table. Moreover, inside the recording room, the captured player wears an HMD to view the virtual environment in front of her at the viewpoint corresponding to the position of the cup on the table. The HMD is connected to a computer outside by a small cable going through the ceiling of the recording room. The cable is painted the same color with the room and its width is small enough to be eliminated by the filter step of 3D Live background subtraction and image processing modules.

In this context, the captured player can actively interact with other virtual objects in virtual reality environment when seeing them on the HMD, and the outside players will have fun seeing her reaction the in mixed reality environment. In our further future work, we want to explore the problem of whether the cup which represent for the 3D Live object can automatically move when the captured player moves inside the room. Such a system will give the captured person more freedom exploring the whole virtual world herself. Technologies in Touchy Internet [46] can be applied to automatically move the special cup around the table. Touchy Internet uses special sensors and wireless system to track the movement of a pet at home backyard and control the doll's movement placed at the office corresponding

to the pet's movement.

The future version of Magic Land will open a new trend for mixed reality games, in which players can actively play a role of a main character in the game story, be submerged totally in the virtual environment, and explore the virtual world themselves, while at the same time in mixed reality environment, other players can view and construct the virtual scene and new virtual characters to challenge the main character. Consequently, the game story is not fixed but will depend on the players' creativity and imagination, and follow their reactions when they travel around the virtual word.

Appendix A

User Study of Magic Land System

A.1 Aim of this User Study

We conducted this user study to obtain the feedback from the users regarding their perception to our Magic Land system. For example, their feeling on interacting with virtual objects, being captured in 3D in a special recording room, etc. This survey also helps to assess how much this system promotes social interaction and remote 3D collaboration. The improvements that may be continuously made in the future work are also expected to obtain from this user study.

A.2 Design and Procedures

Thirty subjects (13 Females and 17 Males) were invited to participate in this study. The age group of the subjects ranges from 15 years old to 54 years old, with the

average age of 25.4 years old. All of them reported clear-vision and normal hearing abilities.

During the user study, each subject will go into the 3D Live recording room first, and follow the system voice instructions to record herself. After the recording is finished, the system will ask her to leave the recording room and go to the Menu Table and wait for her 3D data to be transferred over. Once her captured 3D Live data being sent to the Menu Table, the subject then can use the green button on the Menu Table to find herself amongst the various recorded human characters. Once she has found herself, she can then use one of the empty cups to pick herself up, and put herself onto the main interaction table. She can then go and pick some more captured human 3D Live characters and virtual 3D VRML characters and add to the interaction table, and try the interactions among them. Subjects were also encouraged to play this system together with their friends at the same time (social collaboration).

After the subjects tried all the functions of this system, they were asked to fill in a questionnaire paper with 13 questions as shown in Table A.1 and A.2.

A.3 Results of this User Study

Question 1 and 2 are used to assess the overall feelings of the subjects to the Magic Land system. The two main features here are merging the user into the virtual world, and interacting with other virtual objects. From the feedback, we found that

Table A.1: Questions in the user study

Questions	**A**	**B**	**C**	**D**
1. Overall, how do you rank the Magic Land as a concept of merging yourself into the virtual world?	Very exciting.	Exciting.	Moderate.	Boring.
2. Overall, how do you rank the Magic Land as a concept of interacting with virtual objects?	Very exciting.	Exciting.	Moderate.	Boring.
3. In your view, how much does this technology promote social interaction?	Very helpful in promoting social interaction.	Some help in promoting social interaction.	Only little help in promoting social interaction.	No help at all in promoting social interaction.
4. How do you feel about being captured in 3D in the special recording room, and then shown in the virtual world on the menu and interactive table?	Comfortable and good	Moderate, same as taking normal photos.	Uncomfortable.	Nervous, and feel uneasy.
5. Would you like to have such 3D Live system for remote 3D collaboration in the future? Here, collaboration means you can see someone remotely in 3D (different from traditional 2D video conference), and work and play with him/her together.	Yes, looking forward to trying it.	Yes, it might be a good idea.	I don't really care.	No, I don't think it will work well.
6. How collaborative is this system if we implement it for the remote 3D collaboration? Here, collaboration means you can see someone remotely in 3D (different from traditional 2D video conference), and work and play with him/her together.	Very collaborative. Everyone is working closely together to achieve the goal.	There is some level of collaboration here.	Only little collaboration here.	No collaboration at all, it's basically a single player system.
7. How entertaining is this system to you?	Very fun! I really enjoyed it.	It is a nice game. Good for playing occasionally.	It is about the same as the other games. Not much difference.	I don't like this game. It is not entertaining at all.

Table A.2: Questions in the user study (cont.)

Questions	**A**	**B**	**C**	**D**
8. Compare to current 2D communications such as web cam, do you think this system is useful for communication in telepresence? Telepresence is to use computer technology to give the appearance of an individual being present at a location other than the actual location of that individual.	Very useful. It will make telepresence communication very exciting.	It may be useful for the telepresence communications.	Telepresence communication may not be very different from the current 2D communications.	I don't think this system can help in communication in telepresence.
9. Do you like the idea of using a physical cup to pick up the virtual objects or 3D Live characters, comparing to using mouse and keyboard as in traditional computer games	It is good. The cup is easy to use for picking up objects.	I think there is no difference between using a cup and using normal mouse and keyboard for the control.	I prefer to use mouse and keyboard instead.	
10. How do you feel about the control of interaction between objects by moving the cups around on the interactive table?	Very interesting. I like this way of control of interactions.	It is ok. But the cup is not so easy to move around on the table.	This kind of control is fine. But I also like to use traditional mouse and keyboard to control.	I don't like to cups.
11. In your view, how much do the physical cups promote the social collaboration to make interactions on the table, comparing to the traditional way of using keyboard and mouse?	Very helpful in promoting social interaction.	Some help in promoting social interaction.	Only little help in promoting social interaction.	No help at all in promoting social interaction.
12. How do you feel about the deleting of the objects in the cup by using a virtual trash can?	Very easy to use. It is a good idea.	It is fine.	I don't feel special, neither do I like it.	I don't like this way of deleting.
13. Would you like to try this kind of system again in the future?	Yes. I am looking forward to it.	Yes, maybe I will try.	I don't want to try any more.	

25 subjects out of 30 felt *Very Exciting* about the concept of merging themselves into the virtual world; and 20 subjects felt *Very Exciting* about the concept of interacting with virtual object. From this results, we can see that this technology is indeed very attractive to the general public.

Question 3 is concerning about how much this technology can help to promote the social interaction. The feedback was quite positive. In total, 20 subjects felt that this system can help to promote social interaction, and 6 of them felt it is *Very helpful*. Question 4, 5 and 6 are concerning about the 3D Live recording room. 18 subjects felt that 3D Live recording process is *Comfortable and good*, and 9 felt *Moderate*. Only 3 subjects felt that the recording processing is *Uncomfortable* or make them *Nervous and feel uneasy*. 73.4% of the testing subjects felt this system can be used for remote 3D collaboration in the future, and 63.4% of the testing subjects believed such system would be collaborative. It shows that this 3D Live capturing process can be accepted by most of the population. The feedback of question 8 shows that nearly two third of the testing subjects think this system is useful in telepresence comparing to the current 2D video teleconferences.

Another important part of this Magic Land system is that we are using physical cups to pick up and move the virtual objects or 3D Live characters. From the answer to question 9, 10, and 11, we can see that most testing subjects like the way of using physical cups comparing to using mouses and keyboards as in traditional computer games. Comparing to mouses and keyboards, 17 subjects felt that the cups were easier for picking up virtual objects, and 18 subjects felt it is easy for

them to move the objects around using the cups. Also 18 subjects felt that using cups is helpful in promoting social interaction.

As a multimedia system, we also value how entertaining this system is through question 7. As a result, 10 subjects enjoyed the game a lot, and 11 said *It is a nice game. Good for playing occasionally.* This result is quite encouraging to apply this technology in further digital entertainment development. To check how friendly the user interface is, we put question 12 to see how the users feel about the way of deleting virtual object inside the cup. It shows that 70% of the subjects like our idea of using the virtual trash can. And from question 13, we can see that more than 90% of the subjects liked to try this kind of system again in the future.

A.4 Conclusion of the User Study

Overall, from the user study we can conclude that our Magic Land system is testified to have produced a tangible, natural and novel interaction interface to the users. The diagram of the results of all these 13 questions can be viewed in Figure A.1.

Most testing subjects claimed that this system is very attractive, and they were excited to see themselves being captured in 3D, and then being put into the interaction table together with the other 3D objects. Although few people complained that the capturing process makes them feel uncomfortable or nervous, most of them felt comfortable or natural with the system. So, we can say that this

system is acceptable by the general public, and maybe minor modifications can be made to make it more user friendly.

From these results, we can see that most testing subjects felt that mixed reality technology helps to promote the social interaction among participants. More than half of the participants think this technology will be useful for the remote 3D collaboration system in the future. But still a few of them think there are a little collaboration in this system or no collaboration at all. The reason for this should be that all the 3D Live characters we used now are captured separately without any relationship among them. But when the technology be used in the remote 3D collaboration in the future, the captured characters must be related, users should have different feeling.

Using physical cups instead of traditional mouses and keyboards is also proved to be a more natural way of controlling virtual objects from the results of the user study. Most participants felt it easy to use, and helpful in promoting the social interaction. Also we can see that most users think it is a good idea to use the virtual trash can to delete the objects. This result shows that mixed reality technology provides a natural user interface.

Additionally, further improvements of this system may be made in increasing the gaming complexity and hardware refinement. There were still 30% of the testing subjects felt that this system is not so entertaining. We can improve that by adding more meaningful interactions, 3D sound effects, better computer graphics, etc.

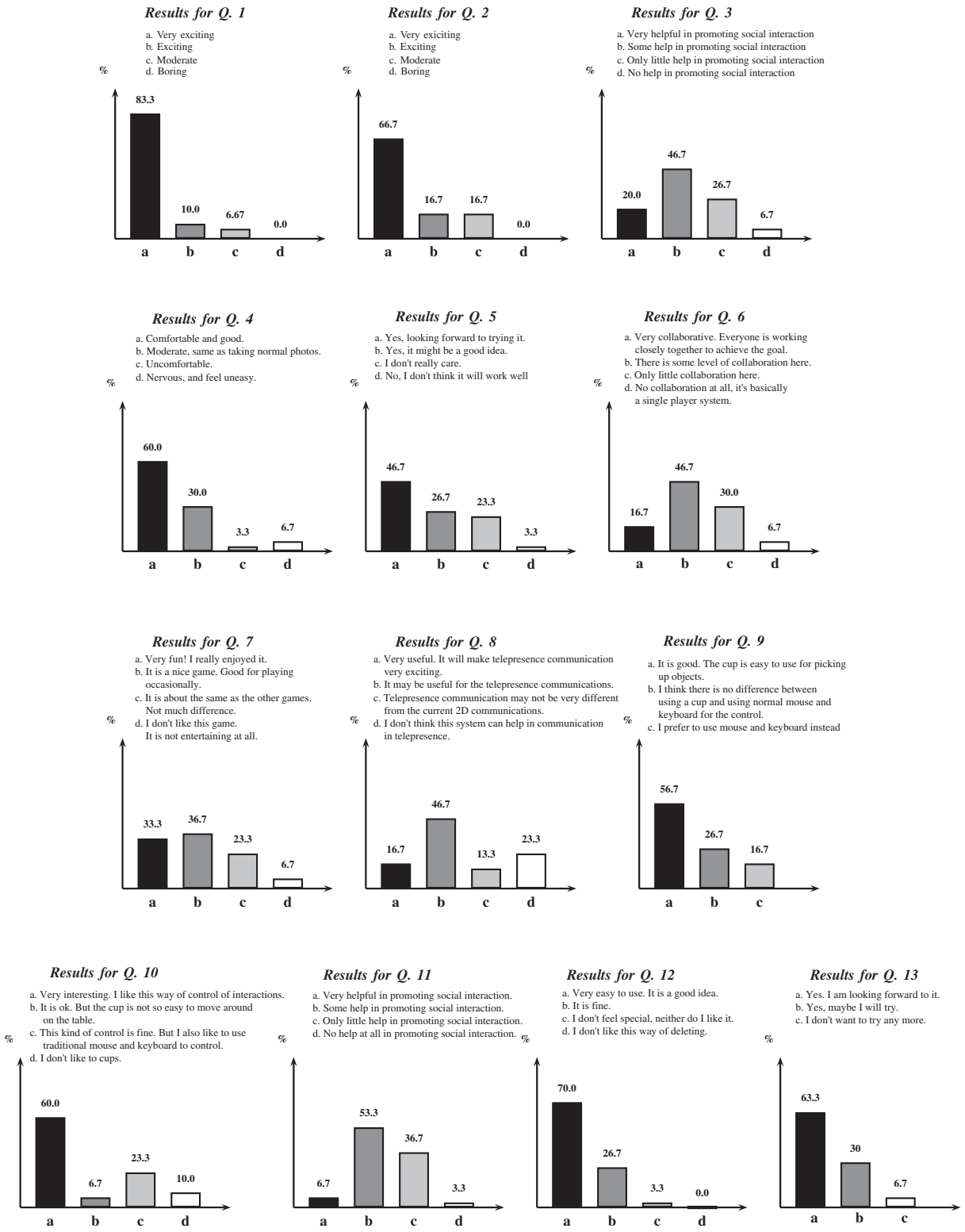

Figure A.1: Graph results for multiple choice questions

Appendix B

List of Publications and Demonstrations

- **Journal paper:**

  Ta Huynh Duy Nguyen, Tran Cong Thien Qui, Ke Xu, Adrian David Cheok, Sze Lee Teo, ZhiYing Zhou, Mallawaarachchi Asitha, Shang Ping Lee, Wei Liu, Hui Siang Teo, Le Nam Thang, Yu Li, Hirokazu Kato, "Real Time 3D Human Capture System for Mixed-Reality Art and Entertainment", *IEEE Transaction On Visualization And Computer Graphics (TVCG)*, November / December Issue of 2005.

- **Conference demonstration paper:**

  Tran Cong Thien Qui, Ta Huynh Duy Nguyen, Asitha Mallawaarachchi, Ke Xu, Wei Liu, Shang Ping Lee, ZhiYing Zhou, Sze Lee Teo, Hui Siang Teo, Le

Nam Thang, Yu Li, Adrian David Cheok, Hirokazu Kato, "Magic Land: Live 3d Human Capture Mixed Reality Interactive System", *Interactivity Venue at SIGCHI 2005*, Portland USA, 6-8 April 2005.

- **Conference proceeding & key note papers:**

  Tran Cong Thien Qui, Ta Huynh Duy Nguyen, Adrian David Cheok, Sze Lee Teo, Ke Xu, ZhiYing Zhou, Asitha Mallawaarachchi, Shang Ping Lee, Wei Liu, Hui Siang Teo, Le Nam Thang, Yu Li, Hirokazu Kato, "Magic Land: Live 3D Human Capture Mixed Reality Interactive System", In *Proc. of International Workshop on Re-Thinking Technology in Museums: Towards a new understanding of visitors experiences in museums*, June 2005.

  Adrian David Cheok, Ta Huynh Duy Nguyen, Tran Cong Thien Qui, Sze Lee Teo, Hui Siang Teo, "Future Interactive Entertainment Systems Using Tangible Mixed Reality", *International Animation Festival*, Hangzhou, China, 2005.

Demonstrations have been shown for academic research community in SIGCHI 2005 conference at Portland, USA, and for public at Singapore Science Center in September 2004 at Planet Games exhibition which has attracted more than 100,000 visitors, especially young children. Magic Land is currently a permanent exhibition for public at Singapore Science Center. Especially, in June 2005, Magic Land was shown for around 30,000 attendees at WIRED NextFest exhibition at Chicago USA. This is a huge exhibition of around 120 projects which has been selected through a

Figure B.1: Magic Land demonstration at WIRED Nextfest exhibition, Chicago, 2005.

worldwide search for cutting-edge prototypes, installations, proof-of-concepts and other emerging technologies.

Figure B.2: Magic Land demonstration at SIGCHI Conference, Portland, 2005.

Bibliography

[1] S. J. D. Prince, A. D. Cheok, F. Farbiz, T. Williamson, N. Johnson, M. Billinghurst, and H. Kato. 3d live: real time captured content for mixed reality. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2002.

[2] S. J. D. Prince, A. D. Cheok, F. Farbiz, T. Williamson, N. Johnson, M. Billinghurst, and H. Kato. Live 3-dimensional content for augmented reality. *IEEE Transactions on Multimedia (submitted).*

[3] W. Matusik, C. Buehler, and L. McMillan. Polyhedral visual hulls for real-time rendering. In *Proc. of the 12th Eurographics Workshop on Rendering Techniques*, 2001.

[4] Tran Cong Thien Qui. Interactive mixed reality media with real time 3d human capture. Thesis for Master of Engineering degree in National University of Singapore, 2005.

[5] M. Billinghurst and H. Kato. Real world teleconferencing. In *Proc. of the conference on HFCS (CHI 99)*, 1999.

[6] Milgram, P. and Kishino, F. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, E77-D(12):1321–1329, 1994.

[7] M. Billinghurst, A.D. Cheok, S.J.D. Prince, and H. Kato. Projects in VR - Real World Teleconferencing. In *IEEE Computer Graphics and Applications, Volume 22*, 2003.

[8] G. Markus, W. Stephan, N. Martin, L. Edouard, S. Christian, K. Andreas, K. M. Esther, S. Tomas, V. G. Luc, L. Silke, S. Kai, V. M. Andrew, and S. Oliver. blue-c: A spatially immersive display and 3d video portal for telepresence. In *Proc. of ACM SIGGRAPH*, 2003.

[9] J. M. Hasenfratz, M. Lapierre, and F. Sillion. A real-time system for full body interaction. *Virtual Environments*, pages 147–156, 2004.

[10] T. Koyama, I. Kitahara, and Y. Ohta. Live mixed-reality 3d video in soccer stadium. In *ISMAR*, pages 178–187, 2003.

[11] Point Grey Research Inc. [Online]. Available at: http://www.ptgrey.com.

[12] ARToolKit. [Online]. Available at: http://www.hitl.washington.edu/artoolkit/.

[13] MXRToolkit. [Online]. Available at: http://sourceforge.net/projects/mxrtoolkit/.

[14] OpenCV. [Online]. Available at: http://sourceforge.net/projects/opencvlibrary/.

[15] S. Y. Lee, I. J. Kim, S. C. Ahn, H. Ko, M. T. Lim, and H. G. Kim. Real time 3d avatar for interactive mixed reality. In *Proc. of the 2004 ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry*, 2004.

[16] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *Proc. 13th Conf. Uncertainty in Artificial Intelligence (UAI)*, 1997.

[17] A. Dempster, N. Laird and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39 (Series B):1–38, 1977.

[18] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. Computer Vision and Pattern Recognition 1999 (CVPR '99)*, 1999.

[19] M. Harville, G. Gordon, and J. Woodfill. Foreground segmentation using adaptive mixture models in color and depth. In *Proc. of IEEE Workshop on Detection and Recognition of Events in Video*, 2001.

[20] T. Horprasert, D. Harwood, and L.S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *Proc. of IEEE ICCV99 FRAME-RATE Workshop*, 1999.

[21] Y. Ivanov, A. Bobick and J. Liu. Fast lighting independent background subtraction. *International Jounal of Computer Vision*, 37(2):199–207, 2000.

[22] M. Seki, H. Fujiwara, and K. Sumi. A robust background subtraction method for changing background. In *Proc. of IEEE Workshop on Applications of Computer Vision*, pages 207–213, 2000.

[23] M. Seki, T. Wada, H. Fujiwara, and K. Sumi. Background subtraction based on cooccurrence of image variations. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages II: 65–72, 2003.

[24] M. Lamarre and J.J. Clark. Background subtraction using competing models in the block-dct domain. In *Proc. of International Conference on Pattern Recognition*, pages I: 299–302, 2002.

[25] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *Proc. of IEEE International Conference on Computer Vision*, pages 255–261, 1999.

[26] O. Javed, K. Shafique, and M. Shah. A hierarchical approach to robust background subtraction using color and gradient information. In *Proc. of IEEE Workshop on Motion and Video Computing*, pages 22–27, 2002.

[27] F. Chang, C.J. Chen, and C.J. Lu. A linear-time component-labeling algorithm using contour tracing technique. *Computer Vision and Image Understanding*, 93(2):206–220, February 2004.

[28] RGB "Bayer" Color and MicroLenses. [Online]. http://www.siliconimaging.com/RGB Bayer.htm.

[29] G. Slabaugh, R. Schafer, and M. Hans. Image-based photo hulls. In *Proc. of the 1st International Symposium on 3D Data Processing, Visualization, and Transmission*, 2002.

[30] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. In *International Journal of Computer Vision, Vol.38, No. 3*, 2000.

[31] S. Seitz and C. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, June 1997.

[32] G. Slabaugh, W. B. Culbertson, T. Malzbender, and R. Schafer. A survey of volumetric scene reconstruction methods from photographs. In *Proc. International Workshop on Volume Graphics*, 2001.

[33] G. G. Slabaugh, W. B. Culbertson, T. Malzbender, M. R. Stevens, and R. W. Schafer. Methods for volumetric reconstruction of visual scenes. In *The International Journal of Computer Vision*, 2004.

[34] R. Szeliski. Video mosaics for virtual environments. In *IEEE Computer Graphics and Applications*, March 1996.

[35] H. Kato, K. Tachibana, M. Tanabe, T. Nakajima, and Y. Fukuda. Magiccup: A tangible interface for virtual objects manipulation in table-top augmented

reality. *Proc. of Augmented Reality Toolkit Workshop (ART03)*, pages 85–86, 2003.

[36] A. Amory, K. Naicker, J. Vincent, and C. Adams. The use of computer games as an educational tool: identification of appropriate game types and elements. pages 311–321, 1999.

[37] T. W. Malone. Toward a theory of intrinsically motivating instruction. 5:333–369, 1981.

[38] CAVE Quake II. [Online]. Available at: http://brighton.ncsa.uiuc.edu/ prajlich/caveQuake/.

[39] T. Oshima, K. Satoh, H. Yamamoto, and H. Tamura. Ar2 hockey system: A collaboration mixed reality system. 3(2):55–60, 1998.

[40] H. Tamura, H. Yamamoto, and A. Katayama. Mixed reality: Future dreams seen at the border between real and virtual worlds. 21(6):64–70, 2001.

[41] A. D. Cheok, X. Yang, Z. Zhou, M. Billinghurst, and H. Kato. Touch-space: Mixed reality game space based on ubiquitous, tangible, and social computing. *Journal of Personal and Ubiquitous Computing*, pages 430–442, 2002.

[42] S. Bjork, J. Falk, R. Hansson, and P. Ljungstrand. Pirates! - using the physical world as a game board. In *Interact 2001, IFIP TC. 13 Conference on Human- Computer Interaction*, Tokyo, Japan, 2001.

[43] A. D. Cheok, S. W. Fong, K. H. Goh, X. Yang, W. Liu, and F. Farzbiz. Human pacman: A sensing-based mobile entertainment system with ubiquitous computing and tangible interaction. 2003.

[44] R. L. Mandryk and K. M. Inkpen. Supporting free play in ubiquitous computer games. 2001.

[45] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. Rtp: A transport protocol for real-time applications. *Internet Engineering Task Force, Audio-Video Transport Working Group*, 1996.

[46] S. P. Lee, F. Farbiz, and A. D. Cheok. Touchy internet: A cybernetic system for human-pet interaction through the internet. *SIGGRAPH 2003, Sketches and Application*, 2003.