國立交通大學

資訊科學與工程研究所

碩士論文

透過 KINECT 影像做視訊監控應用上的立體環境建模與監視

3D Environment Modeling and Monitoring via KINECT Images for Video Surveillance Applications

研究生：馬秉辰

指導教授：蔡文祥　教授

中華民國一百零二年六月

透過 KINECT 影像做視訊監控應用上的立體環境建模與監視

3D Environment Modeling and Monitoring via KINECT Images for Video Surveillance Applications

研究生：馬秉辰	Student：Bing-Chen Ma
指導教授：蔡文祥	Advisor：Wen-Hsiang Tsai

國立交通大學
資訊科學與工程研究所
碩士論文

A Thesis
Submitted to Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in
Computer Science

June 2013
Hsinchu, Taiwan, Republic of China

中華民國一百零二年六月

3D Environment Modeling and Monitoring via KINECT Images for Video Surveillance Applications

Student: Bing-Chen Ma Advisor: Wen-Hsiang Tsai

Institute of Computer Science and Engineering
National Chiao Tung University

ABSTRACT

In this study, several methods and strategies are proposed for 3D environment modeling and monitoring using an octagonal-shaped 9-KINECT imaging device for video surveillance.

Firstly, an environment modeling method is proposed which, based on the pinhole camera model, converts KINECT images into 3D images. In the method, at first a new technique is employed to correct geometrically the bending phenomenon existing in constructed 3D images. The technique is based on the use of an MMSE paraboloid approximation scheme and a data interpolation scheme. Also, a technique is proposed to calibrate spatial relations between KINECT devices by the ICP algorithm. Finally, a technique using the calibration result and the constructed 3D images as inputs is proposed to construct the indoor environment model.

Secondly, a human tracking method is proposed, by which human activities can be detected and tracked using the 9-KINECT imaging device. Specifically, a human detection process is conducted first, which includes the operations of background subtraction, mathematical morphology, and region growing. Then, during the human tracking process, the tilting devices of the KINECTs are used dynamically to track a human. The problem of handoff between KINECT devices, which occurs during the human tracking process, is also solved in this study.

Finally, to extract the features of tracked humans for use in security monitoring, a human modeling method is proposed, in which sequences of 3D images constructed from KINECT images are integrated, using the distance-weighted correlation (DWC) measure and the K-d tree structure, to form a human model. From the model, human features like body height, width, and thickness may be extracted for use in security monitoring and off-line video search.

Good experimental results are also shown, which prove the feasibility of the proposed methods for real applications.

透過 KINECT 影像做視訊監控應用上的立體環境建模與監視

研究生：馬秉辰　　　　指導教授：蔡文祥 博士

國立交通大學資訊科學與工程研究所

摘要

本研究設計一新的八角形 9-KINECT 視訊裝置，並提出一系列相關策略和方法，進行視訊監控上立體環境之建模及人物活動之追蹤。

首先，對於環境模型之建立，使用針孔成像原理將 KINECT 影像轉換成立體影像，進而使用幾何修正的方式，利用最小平方差橢圓曲面內差近似法，去修正立體影像的彎曲現象。接著，使用遞迴最近點(iterative closest point, ICP)演算法校正 KINECT 裝置間之空間相對關係。最後，使用校正出來的結果和立體影像，建立出室內環境的模型。

在使用八角形 9-KINECT 視訊裝置做人物追蹤方面，本研究首先進行人物偵測，使用的方法包括深度影像背景相減法、數學形態學操作和區域增長等技術。偵測到人物之後、進行人物活動追蹤時，會動態去改變 KINECT 裝置的仰角及處理 KINECT 之間的換手問題。

最後是建立人物之模型並擷取人物之特徵，應用於安全監視。對此，本研究藉由立體影像之序列，搭配距離權重相關係數(distance-weighted correlation, DWC)以及 k 維樹(k-d tree)之結構，建構出單一人物之立體模型，並從模型中擷取出人物特徵，如身高、體寬以及身體厚度，做為安全監視和事後觀看之用。

上述諸方法的實驗結果良好，證明在實際應用上該等方法確實可行。

ACKNOWLEDGEMENTS

The author is in hearty appreciation of the continuous guidance, discussions, and support from his advisor, Dr. Wen-Hsiang Tsai, not only in the development of this thesis, but also in every aspect of his personal growth.

Appreciation is also given to the colleagues of the Computer Vision Laboratory in the Institute of Computer Science and Engineering at National Chiao Tung University for their suggestions and help during his thesis study.

Finally, the author also extends his profound thanks to his dear mom and dad for their lasting love, care, and encouragement.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

1.1 Background and Motivation

With the advance of technology, there are more and more vision-based devices in our daily life for different applications. Some of them are used to monitor events in environments or track objects dynamically. Others are used as event recorders, and the recorded data are used for event analysis or other applications.

In recent years, Microsoft releases a new type of sensing device — KINECT. It can capture not only RGB color images and audio data but also depth information as well as the data of the human skeleton at the same time. With the depth information, we can translate it into 3D information. It is beneficial to researches of 3D object detection and modeling.

So, in this study it is desired to design a 3D video surveillance system using multiple KINECT devices and implement some applications described as follows.

1. Monitoring an indoor environment and displaying the captured images of the environment in 3D manners for users to inspect the recorded environment data from different viewpoints.
2. Using the depth information provided by KINECT devices to detect and track human activities and providing changes of viewpoints from different KINECT devices.
3. Creating human models when users browse the records acquired by the KINECT devices, and providing the features of the humans such as height, body width, body thickness, etc.

1.2 Review of Related Works

In this section, we conduct a survey of related works about the construction of the 3D video surveillance system, including 3D environment modeling and motion object detection and tracking.

Many modeling techniques have been proposed for object or environment modeling using data acquired with the KINECT device. Zollhöfer [1] proposed a simple algorithm which uses robust non-rigid registration and merging of the deformation face model to simulate a high-quality virtual interactive 3D face. The data used in the modeling work were captured with the KINECT device. This technique can be applied to computer animation. Shahram [2] proposed a technique, called KinectFusion, which uses the depth information acquired by moving the KINECT device to build up a high-quality and geometrically-precise 3D model quickly. In the operation of his system, a user takes a KINECT device and moves around the indoor environment, and the system will scan and model the entire environment in a short time. The precision of the model can also be adjusted by changing the distance from the target to the KINECT device.

Henry [3] proposed a 3D mapping system which uses visual features and a shape-joint optimization algorithm with RGB color images and depth information acquired with KINECT devices as inputs. In a cooperative project conducted by the MIT, the University of Washington, and Intel Lab. [4], the researchers put the KINECT device on a small airplane to acquire data and built a full view of the 3D environment using the features extracted from the acquired data and an RGBD-SLAM algorithm.

Many algorithms were proposed for motion detection and tracking. Chaiyawatana [5] constructed an automatic system for vehicle detection by a frame subtraction technique. The algorithm adopts a suitable threshold and subtracts each frame from its previous one. The results are analyzed by some process units to detect motion objects using the threshold value. Tian [6] used pixels from continuous video frames and a Gaussian distribution to build up and adjust a background model. By this way, noise coming from light changing, leaf swaying from the background, and so on, in each frame can be avoided when the human detection work is conducted by background subtraction. Xia [7] used the depth information from the KINECT device to do 2D chamfer matching and adopted some human features to figure out human shapes to conduct human activity tracking. Meltem [8] proposed a standard video tracking and person classification system. When a human is tracked under multiple video devices, the system puts the faces and the soft biometric features into the feature domain to develop an algorithm of feature extraction. This algorithm can acquire the features of the human sex, the human race, and other soft biometric features in the low-resolution video or in the unknown illumination video. It also solves the handoff problem between multiple video devices. Pantrigo [9] considered, in a video processing system, the descriptions of human activities under different situations such as sport technique analysis and video surveillance. A highly efficient system was proposed for multiple-object tracking, which can not only merge particle filtering and the memetic algorithm correctly but also track multiple targets precisely in image sequences and classify the extracted human beings if needed.

1.3 Overview of Proposed Methods

To reach the goal of this study mentioned above, at first we should construct a device for use as a 3D video surveillance system. The device is constructed in this study by use of multiple (nine) KINECT devices and all of them are placed in the device at fixed positions. The device will be called an *octagonal 9-KINECT imaging device* in the sequel of this thesis. More than one octagonal 9-KINECT imaging device have been produced in this study. A picture of one of such devices is shown Fig. 1.1. Each of them is then fixed on the ceiling of our experimental environment with a suitable height, and the KINECT devices in it may be used to acquire images of the environment around from top to bottom from the full view of 360^{o}. More details about such devices, such as the design idea and the inside structure, will be presented in Chapter 3.

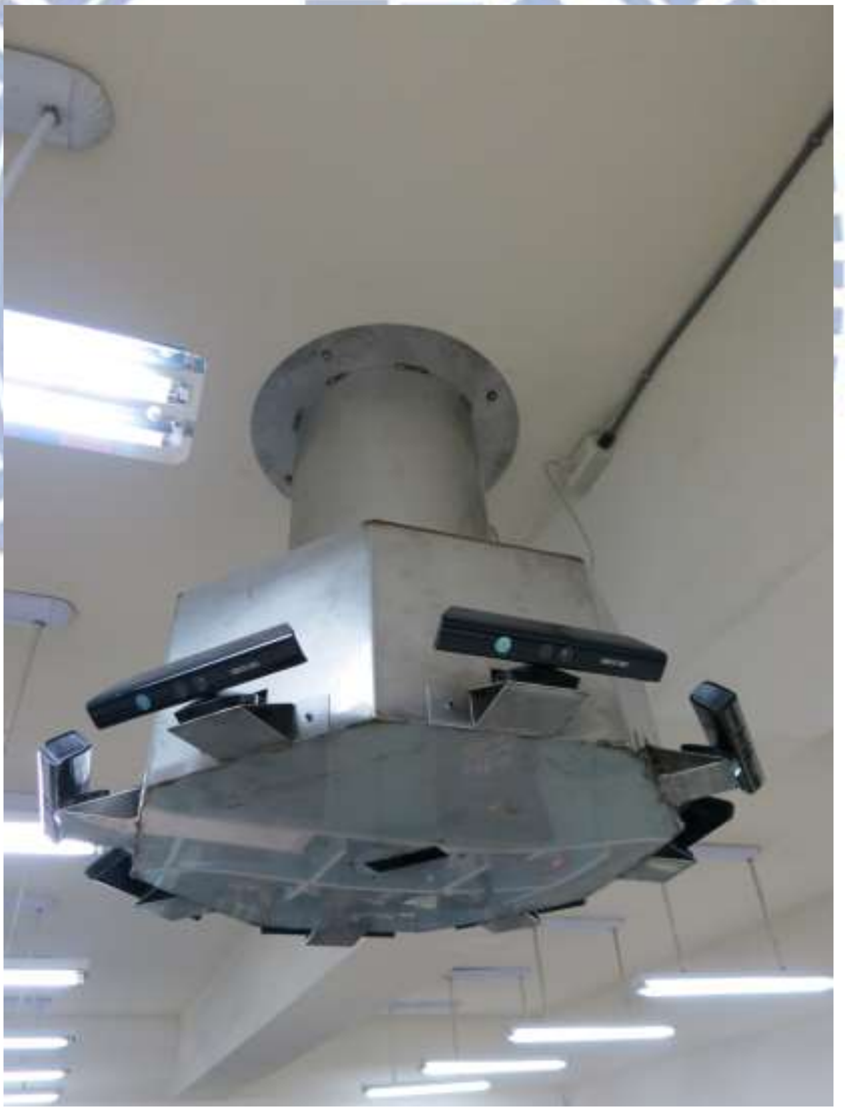


Figure 1.1 A picture of the proposed octagonal 9-KINECT imaging device.

The next major task in the proposed system is integration of the data acquired with the octagonal 9-KINECT imaging device. Because the depth information acquired with each KINECT device is not 3D in nature, we convert the depth information acquired by the device into 3D data form. The detail about the data structure and the proposed method for such conversions will be introduced in Chapter 4. Some definitions will also be given in that chapter.

With the 3D data, we can calibrate the spatial relations between the KINECT devices before modeling the indoor environment. Because all of the KINECT devices are placed at fixed positions, it is easier to use a calibration target to conduct the calibration work. The details of the proposed calibration technique will be explained in Chapter 5.

After getting the results from the calibration process, we use them next to construct the indoor environment model. More specifically, we shift the 3D data for each KINECT device to a proper position for registering the data acquired by the neighboring device so that we can build up a complete indoor environment model. The details of the proposed shifting method and the modeling construction process will be described in Chapter 5.

After building up the environment model, we start to detect and track humans and associated activities. For this, we make two assumptions as follows.

1. The indoor environment is unchangeable all the time.
2. The detected motion objects are humans for security surveillance.

These assumptions are helpful for designing schemes to detect human activities, which will be presented in Chapter 6.

As for the purpose of tracking human activities, because the KINECT device can be tilted in space, we use this function to track human activities dynamically. Besides, because we use multiple KINECT devices, device handoff problems will occur in our system, which will affect the ways of displaying the recorded data. We will explain our human tracking strategy and the proposed solution to the handoff problem in Chapter 6 as well.

Finally, when users browse the records of indoor monitoring, the proposed system will access the saved 3D data which have been recorded by the 9-KINECT imaging devices, and converted them to model any detected human. From the model, the system will also extract the features of the human, such as his/her height, body width, body thickness, etc. The details of the modeling algorithm and the feature extraction process are introduced in Chapter 7.

1.4 Contributions

Some contributions of this study are listed in the following.

1. Designing a 3D video surveillance system using multiple KINECT devices and integrating all the data acquired by different KINECT devices.
2. Displaying in 3D ways of the monitored environment after integrating all the view images acquired by the KINECT devices and providing different viewpoints for convenient browsing by the user.
3. Fully using the capabilities of KINECT devices by tilting the devices to track human activities dynamically.
4. Extracting more features from the human model than from 2D images such as height, body width and body thickness.

1.5 Thesis Organization

The remainder of this thesis is organized as follows. In Chapter 2, we introduce the configuration of the proposed system and the system process in detail. In Chapter 3, we introduce the design of the hardware device of the 3D video surveillance system in detail, and analyze its performance. In Chapter 4, we describe the proposed schemes for conversion of KINECT data into 3D image data, and correction of the conversion result. In Chapter 5, we describe the proposed methods to calibrate the KINECT devices and to model the indoor environment. In Chapter 6, we introduce the proposed human detection and tracking method. In Chapter 7, we introduce the proposed human modeling method and the 3D way we use for displaying the result. In Chapter 8, we will show some experimental results of the entire system process. At last, conclusions and some suggestions for future works are given in Chapter 9.

Chapter 2
Ideas of Proposed Methods and System Design

2.1 Ideas of System Design

To complete the construction of the proposed 3D video surveillance system, it is important to design an appropriate structure of the video acquisition device for the system. The field of view of a single KINECT device is not wide enough, so we construct an *octagonal* 9-KINECT imaging device using multiple KINECT devices to extend the view of field. It not only can monitor an indoor environment which is large enough as a whole, but also can fully use the tilting mechanism in the KINECT device for dynamic human activity tracking. The detail of the octagonal 9-KINECT image device will be introduced in Chapter 3.

After constructing the octagonal 9-KINECT imaging device, we affix it on the ceiling of our experimental environment at a suitable height, and the KINECT devices in it are used to acquire image data of the around environment by tilting them from top to bottom for a full view of 360^{o}. Because the KINECT devices in the octagonal 9-KINECT imaging device work individually and the computer controller acquires images sequentially, we set an image acquisition order for the KINECT devices. When acquiring the data from KINECT devices, we will sort the data by this order of KINECT devices.

Finally, we design several software process units to analyze the data acquired from the KINECT devices and display the result. More details about the hardware devices which we use in this study and the software for processing image data and displaying the processing result will be described in Section 2.2. The system processes are introduced in Section 2.3.

2.2 System Configuration

In this section, we introduce the configuration of the proposed 3D video surveillance system. The hardware of the proposed system includes the KINECT devices we use in this study widely and the necessary devices for acquiring data from multiple KINECT devices. It will be introduced in detail in Section 2.2.1. In Section 2.2.2, we will describe the software development environment for processing data and displaying results.

2.2.1 Hardware Configuration

The sensor we use in this study widely is the KINECT device which is made by Microsoft. It consists of one RGB camera, a couple of 3D depth sensors, a set of multi-array microphone, and one motorized tilt. Its appearance is shown in Figure 2.1. Its vertical and horizontal viewing angles are 43° and 57°, respectively. Its vertical tilt angles range from –27° to 27°. Its sensing distances for the color image, the depth image, or the skeleton tracking ranges from 1.2 meters to 3.6 meters, but the actual sensing distance used in this study will be larger and will be discussed in Section 2.2.2. The maximum resolution of the color image and the depth image captured from the KINECT device is up to 1280×960 pixels with a lower frame rate. For performance efficiency, we usually use the resolution of 640×480 pixels and 320×240 pixels in our system, and the frame rate is kept 30 fps. Its audio format is 16-kHz and 24-bit mono pulse code modulation (PCM). Its audio unit has a four-microphone array with a 24-bit analog-to-digital converter (ADC), and a Kinect-resident signal processing unit with the functions of acoustic echo cancellation and noise suppression. In this study, we won’t use the audio device and the skeleton tracking function.

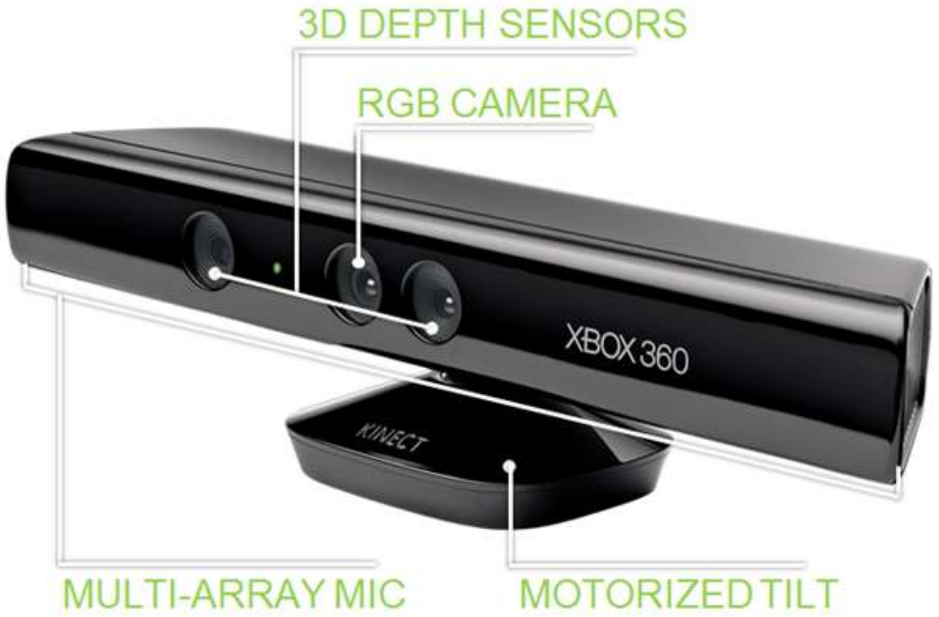


Figure 2.1 The KINECT device used in this study.

A single KINECT device uses a USB to deliver its data to the data-processing device (a computer), so the data-processing device should prepare more USB ports for multiple KINECT devices. Furthermore, the data volume delivered by a single KINECT device is too huge, so we can’t use a general USB port extension without adding a USB controller to the data-processing device. In this case, the KINECT device relies on more USB controllers than USB ports, so we should prepare more USB controllers instead of more USB ports for the data-processing device. As previously mentioned, we install the Aguila SU16T Base and the Aguila SU16T Expansion to our data-processing device to extend USB ports and controllers. The Aguila SU16T Base and the Aguila SU16T Expansion are shown in Figure 2.2. The

Aguila SU16T Base is installed on the mother board by PCI Express with 16 ports, and the Aguila SU16T Expansion is installed on the Aguila SU16T Base. The Aguila SU16T Base and the Aguila SU16T Expansion provide 8 USB controllers and each USB controller has 2 USB ports.

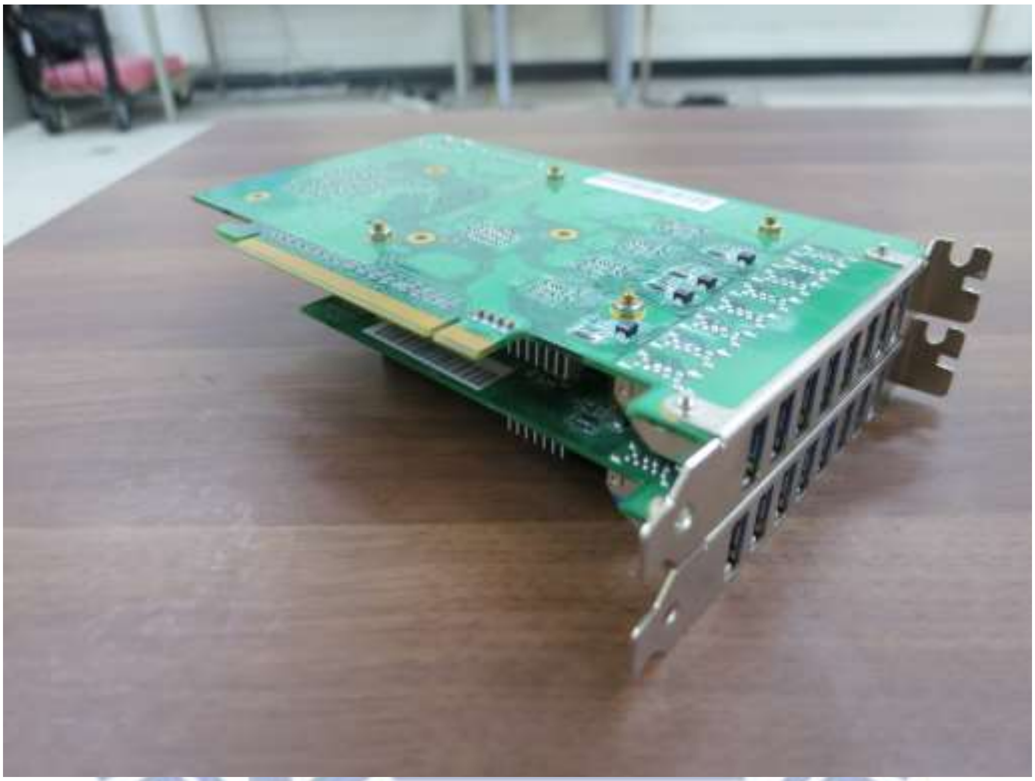


Figure 2.2The Aguila SU16T Base is on the top of PCI Express x16 and the Aguila SU16T Expansion is at the bottom.

2.2.2 Software Configuration

After the hardware of the 3D video surveillance system is constructed, we build up a data-processing system to implement the desired functions of the 3D video surveillance system. The system is written in the C++ programming language using the Microsoft Visual Studio 2010 development environment, and run under the Windows 7 operating system. The system initializes the KINECT device and acquires the image data from the KINECT devices through the Kinect-for-Windows SDK, which is provided by Microsoft. By the way, the maximum sensing distance is 4 meters by using the Kinect-for-Windows SDK, because Microsoft considers that distances smaller than 4 meters is more precise than those larger than 4 meters. The system also uses open sources such as the Open Source Computer Vision (OpenCV) and the Open Graphics Library (OpenGL) to assist data processing. By using the OpenCV application programming interface (API), the system can process the image data easily, and display the result in 3D manners by the OpenGL API.

2.3 System Processes

With the hardware and software configuration completed, we will introduce the whole process of the proposed processing system in detail in this section. For this, we separate the system process into four parts.

The first part is a data conversion process. Because the depth information acquired from the KINECT devices is not 3D in nature, we should convert it to 3D data and the converted data can also be used for other processes. The detail of the conversion scheme will be described in Chapter 4.

The second part is a model construction process of the indoor environment. First, we use the 3D data, which are obtained from the data conversion process just mentioned, from each KINECT devices to calibrate the spatial relation between KINECT devices. Afterwards, we use the calibration result to merge the 3D data and construct an indoor environment model. Finally, we show the model with color images in 3D manners. The flow of the process is shown in Figure 2.3, and the details of the calibration strategy, the merging algorithm, and the model display scheme will be introduced in Chapter 5.

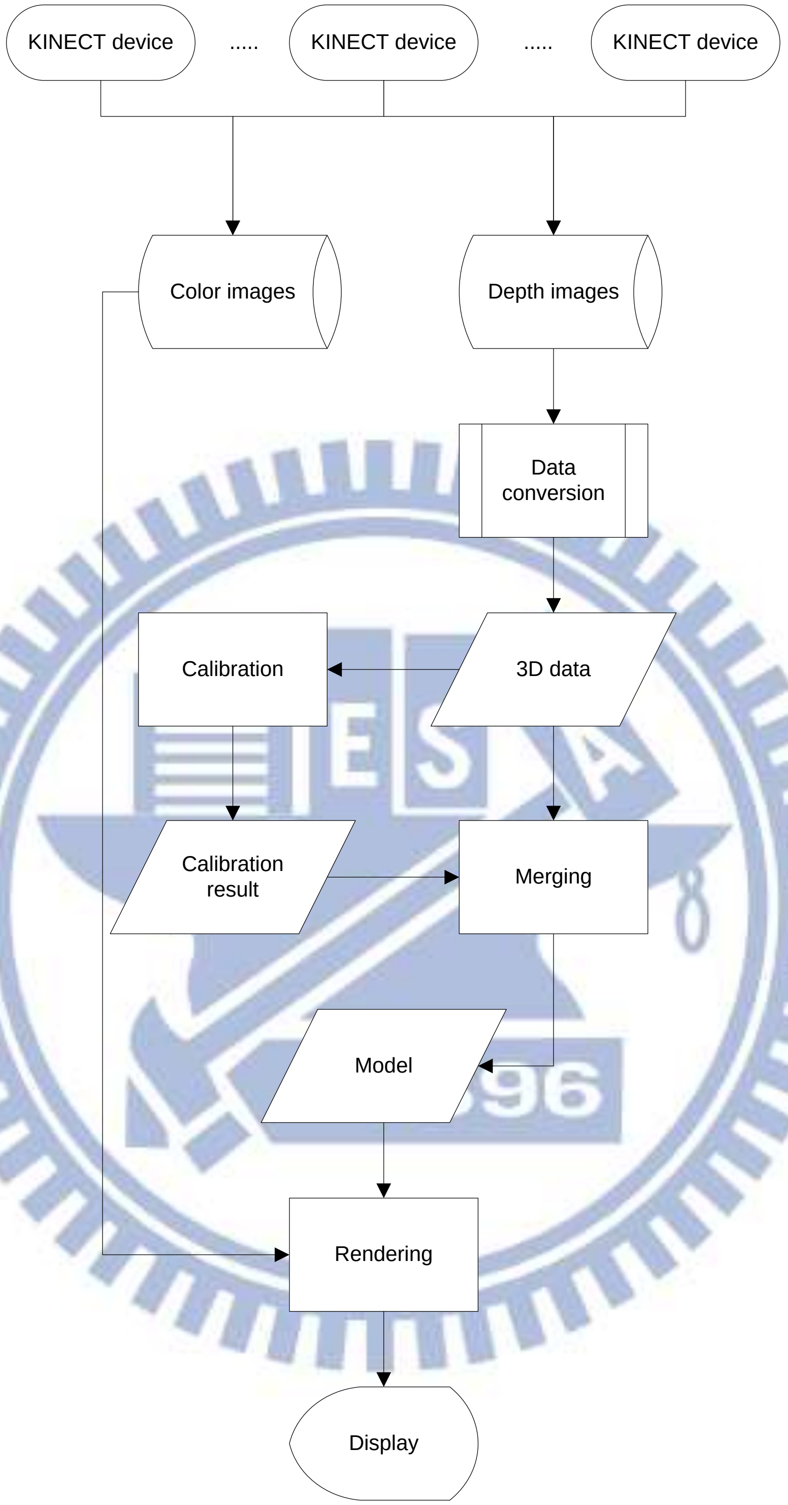


Figure 2.3The model construction process of the indoor environment

The third part is a process of human activity tracking. First, we use depth images to detect human activities. By the detection strategy used in this study, we conduct background learning and noise elimination. The detail of human detection will be described in Section 6.2. Next, we use the result of detection to track human activities. When tracking human activities, we will adjust the tilter of the KINECT device dynamically. Furthermore, we will also change the viewpoint by the in-time handoff between KINECT devices and display the result with color images in 3D manners. The flow of the whole process is shown in Figure 2.4. The details of the tracking algorithm will be introduced in Section 6.3, and some experimental results will be shown in Section 6.4.

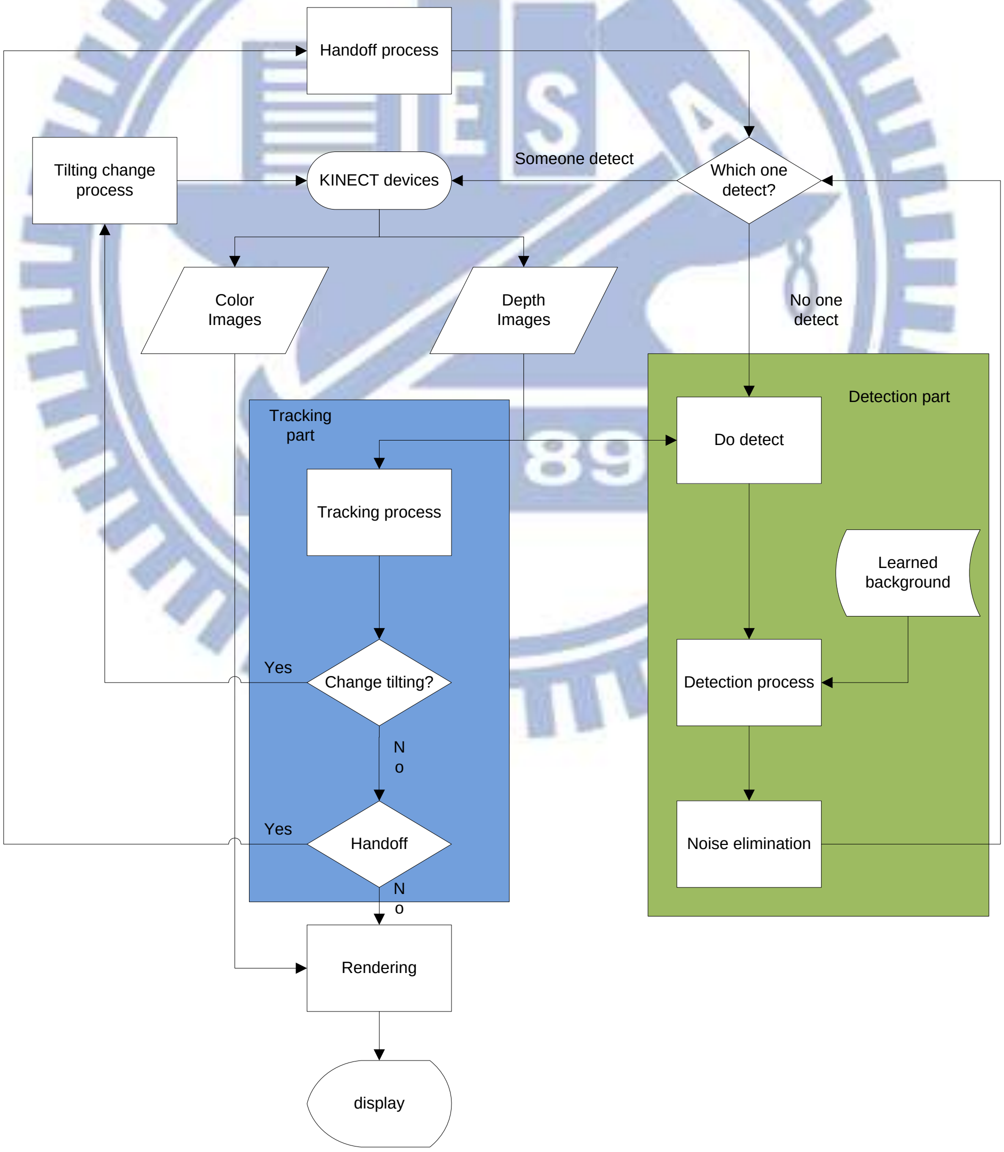


Figure 2.4 The process of tracking human activities.

The forth part is a process of human model construction and human activity display. We will convert the 3D data, which are recorded by the KINECT devices, by a data conversion process proposed in this study to build up the human model. For this, at first we segment the human activity in each frame out from individual KINECT devices by using the detection method described in Section 6.2. Next, we merge the 3D data obtained for the individual KINECT devices. Then, we use the merging results of individual KINECT devices to merge again to build up a finer human model. Finally, we display the human model and show the human features extracted from the model. The whole process is shown in Figure 2.5, and the detail of the process will be introduced in Chapter 7.

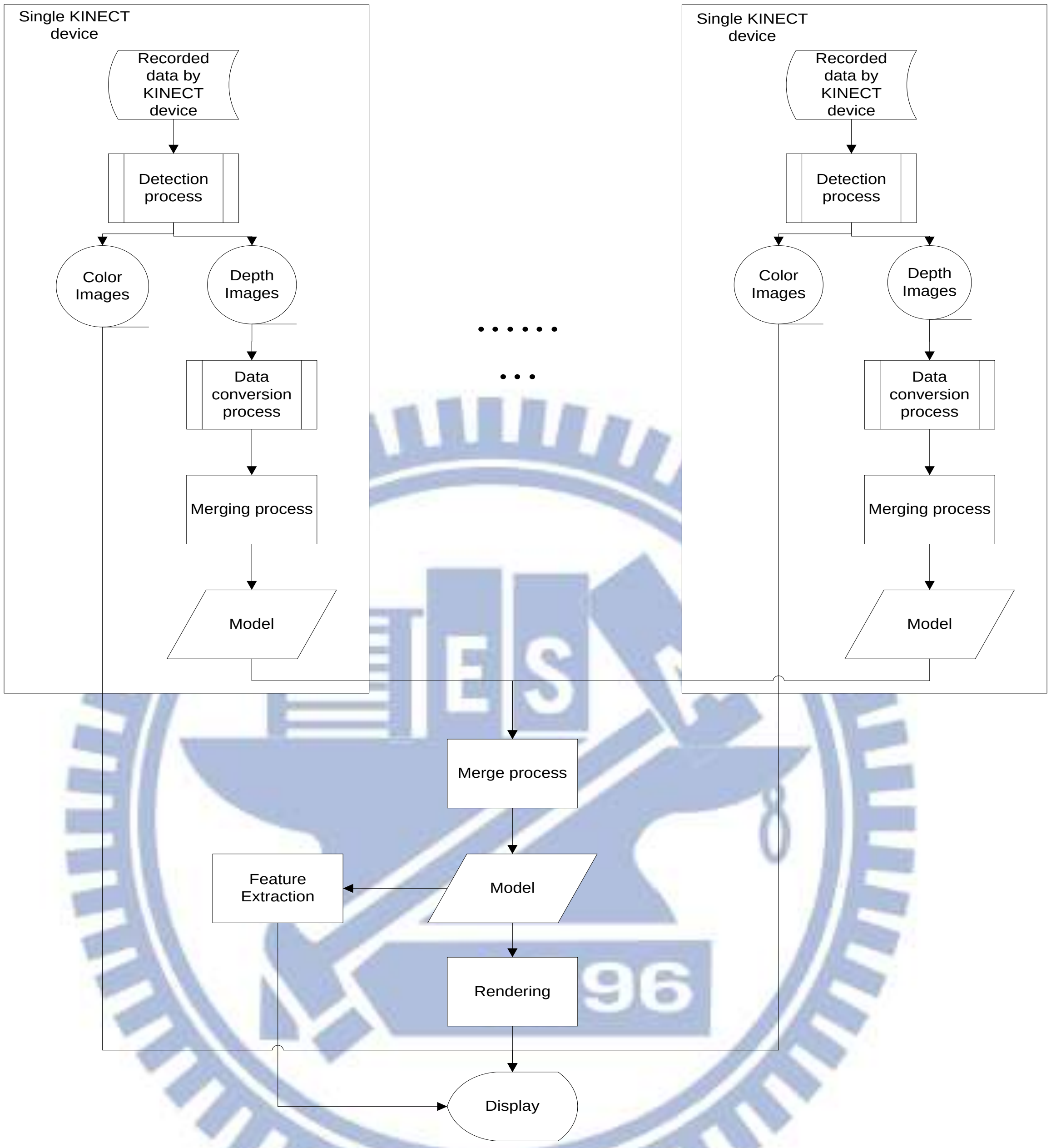


Figure 2.5 The process of constructing human model and displaying human activities.

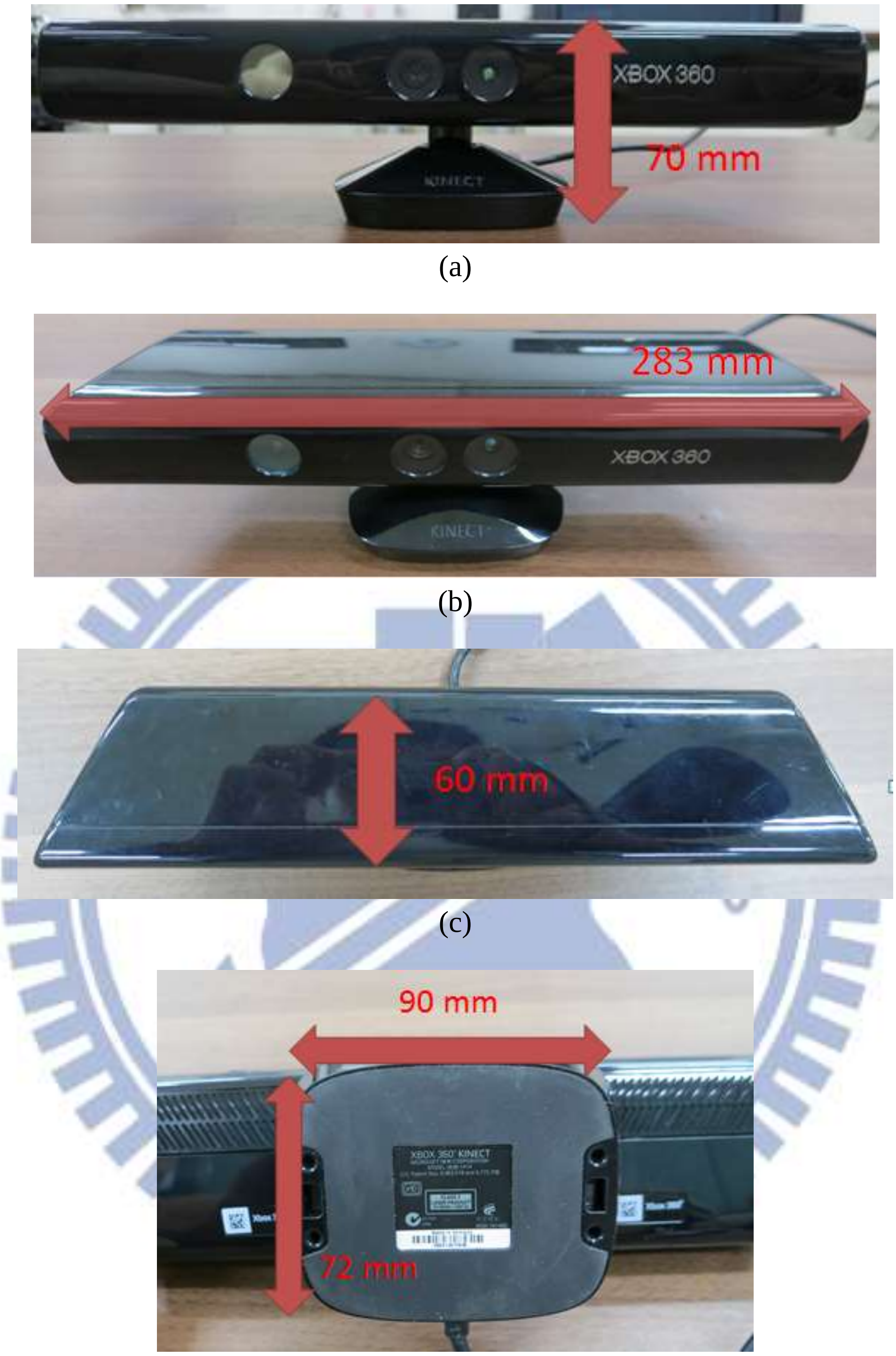
Chapter 3
Design of Proposed Octagonal 9-KINECT Imaging Device

3.1 Introduction to KINECT Device

In this study, we have designed an octagonal 9-KINECT imaging device for environment monitoring. About the basic unit of the imaging device, namely, the KINECT device, we have presented some of its basic specifications in Section 2.2.1, but we would like to introduce the structure of the KINECT device in detail in this section.

The height of the whole KINECT device is 70 millimeters, the width of the main part of the KINECT device is 283 millimeters, and the thickness of the main part of the KINECT device is 60 millimeters. The area of the basement of the KINECT device is 90×72 square millimeters. The structure specifications are shown in Figure 3.1.

The KINECT device can also change its panning angle by manual adjustment, but we won’t use the panning angle in this study because the constructed 9-KINECT imaging device is hung high up on the ceiling for monitoring the environment from a higher position. The KINECT device contains a gravity sensor which can detect the tilting angle between the device and the ground. We will use this tilting function to monitor wider areas of the environment.



(a)

(b)

(c)

(d)

Figure 3.1 The Structure specifications for each part of the KINECT device. (a) The height of the KINECT device. (b) The width of the main part of the KINECT device. (c) The thickness of the main part of the KINECT device. (d) The area of the basement of the KINECT device.

3.2 Ideas of Proposed Design

In this study, we want to use multiple KINECT devices for the proposed 3D video surveillance system, but we can't directly use multiple KINECT devices without being organized. So we propose the octagonal 9-KINECT imaging device to organize multiple KINECT devices. The idea of the design of this system is described in this section.

Firstly, we have to know how many KINECT devices we should use. As we mention in the previous sections, the horizontal viewing angles of a single KINECT device is 57^o, so we should use at least 7 KINECT devices for a full view of 360^o. In our design, we would like to use 8 KINECT devices to cover the full view with a certain degree of overlapping. But when we use the 8 KINECT devices to sense outward for a full view of 360^o, there is a missing field of view which appears in the combination of the 8 views given by the 8 KINECT devices, namely, the middle part. So, we add an additional downward-looking KINECT device to make up the missing field of view. So, totally 9 KINECT devices are used to establish the system. The basic placement idea of the 9 KINECT devices is illustrated in Figure 3.2.

With the basic placement idea, we can make a container for the 9 KINECT devices as shown in Figure 3.3 which is a copy of Figure 1.1. We also consider the utility of the tilting device within each KINECT device, so we place the 8 KINECT devices, which are sensing outward for a full view of 360^o, on their individual bases outside the container as shown in Figure 3.3.

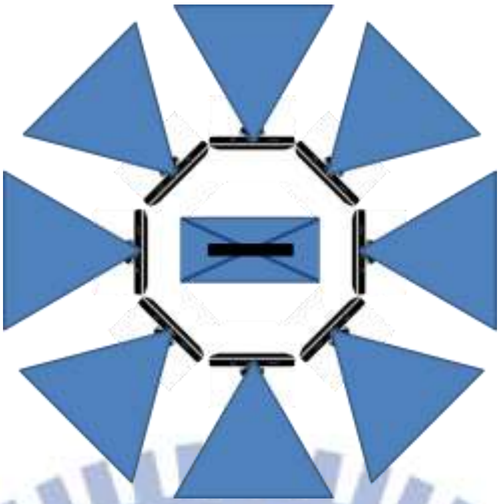


Figure 3.2 The basic placement idea of the proposed octagonal 9-KINECT imaging device. The central KINECT device looks downward and the others senses outward.

3.3 Details of Design

With the design idea as described above, we will now introduce the design specification of the proposed octagonal 9-KINECT imaging device in detail. We will separate the design specification into three main parts: interchangeable bases for KINECT devices, the container, and the top part. The whole appearance of the octagonal 9-KINECT imaging device is already shown in Figure 3.3.

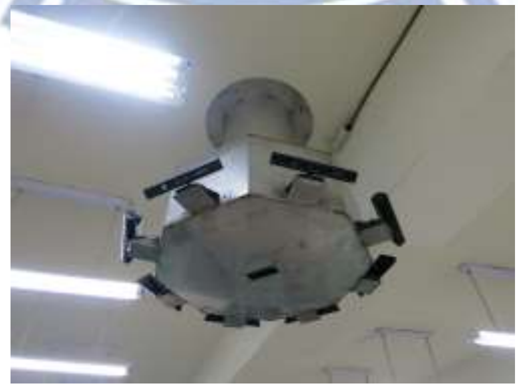


Figure 3.3 The octagonal 9-KINECT imaging device.

3.3.1 Interchangeable Bases for KINECT Devices

The first part is interchangeable bases for the outer 8 KINECT devices. We want to use the outer 8 KINECT devices to sense more information above the ground when the outer 8 KINECT devices are placed on the bases with a suitable height. Therefore, we designed an incline for every base. The tilt angle of the incline is 30^{o}. Because the area of the basement of the KINECT device is 90×72 square millimeters, we design the incline to have the area of 100×100 square millimeters to fit the basement. We also make two screw holes to fix the whole base. The base is shown in Figure 3.4.

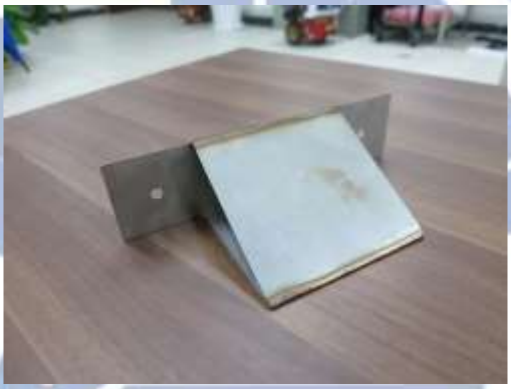


Figure 3.4 The interchangeable base.

3.3.2 Container

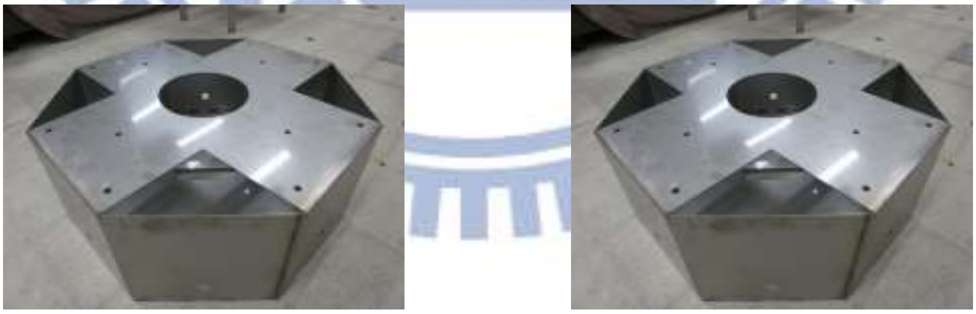
The second part is the container. All the lines of the KINECT devices are put in the container. We design the container in an octagon shape for the outer 8 KINECT devices. Because the width of the main part of the KINECT device is 283 millimeters and we don’t want to make collisions when changing the tilting angles of the KINECT devices, we designed each of the edges of the octagon to be 320 millimeters. The height of the octagonal container is 300 millimeters.

Then, on each side of the octagonal container, we make one square hole and two screw holes. The size of the square hole on the side of the octagonal container is

25×25 square millimeters. For the each KINECT devices on the interchangeable base outside the octagonal container, we can put the transmission line and power line of the KINECT device into the container through the square hole. We also used the two screw holes to fix the interchangeable base.

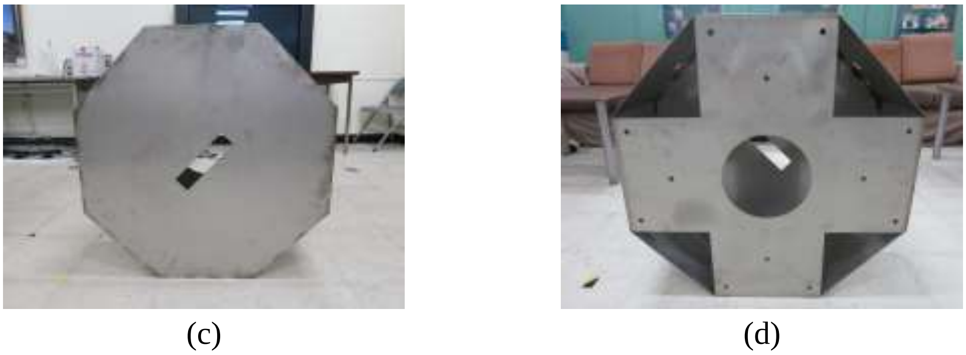
Furthermore, we made a rectangular hole whose size is 70×150 square millimeters on the center of the bottom of the octagonal container. The inner KINECT device can look downward through the rectangular hole.

Finally, the cap of the octagonal container is a cross-shaped plate. We used the crossed plate as a plate to connect with the top part. The width edge of the cross-shaped plate is 320 millimeters and the length of it is 775 millimeters. We made one circular hole whose diameter is 230 millimeters and twelve screw holes on the cross-shape plate. We can put the plugs of the 9 KINECT devices into the top part and arrange all lines of the KINECT devices through the circular hole. We use the twelve screw holes to connect the octagonal container with the top part. The octagonal container is shown in Figure 3.5.



(a) (b)

Figure 3.5 The octagonal container. (a) The whole appearance of the octagonal container. (b) The side of the octagonal container. (c) The bottom of the octagonal container. (d) The cap of the octagonal container.



(c) (d)

Figure 3.5 The octagonal container. (a) The whole appearance of the octagonal container. (b) The side of the octagonal container. (c) The bottom of the octagonal container. (d) The cap of the octagonal container (cont'd).

3.3.3 Top Part

The third part is the top part. We separate the top in three parts. The first part of the top part is a circular plate. The diameter of the circular plate is 600 millimeters. There are four screw holes on the plate. We use the four screw holes to fix the whole octagonal 9-KINECT imaging device on the ceiling.

The second part of the top part is a hollow cylinder. We set two sockets of power extension cords in the hollow cylinder. The two sockets of power extension cords are used to extend the power lines of the 9 KINECT devices. The diameter of the hollow cylinder is 400 millimeters and its height is 650 millimeters. We make one square hole and one rectangular hole on the surface of the hollow cylinder. The size of the square hole is 100×100 square millimeters. We put two plugs of the socket of the power extension cords into the outer socket through the square hole. The size of the rectangle hole is 400×150 square millimeters. A user can put their hands into the octagonal 9-KINECT imaging device through the rectangular hole.

The third part of the top part is another cross-shaped plate. The design specification is the same as the cross-shaped plate of the octagonal container. A user can arrange all lines of the 9 KINECT devices through the circular hole. We connect the top part and the octagonal container with the twelve screw holes. Finally, we welded the three parts of the top together. The top part is shown in Figure 3.6.

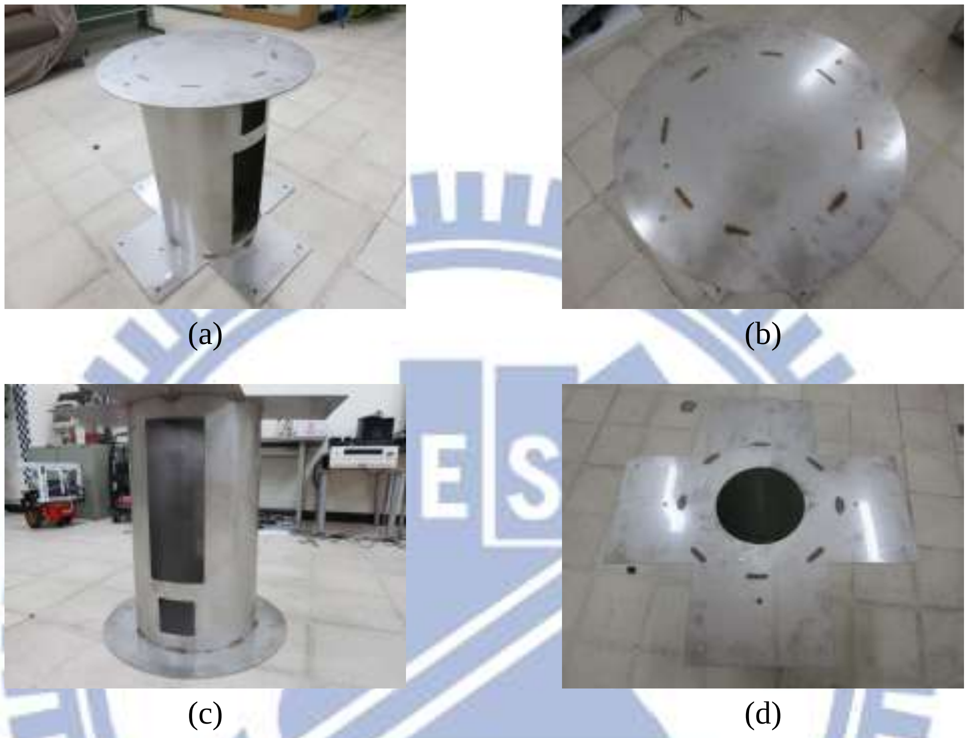

(a) (b) (c) (d)

Figure 3.6 The top part. (a) The whole appearance of the top part. (b) The circular plate of the top part. (c) The hollow cylinder of the top part. (d) The crossed plate of the top part.

3.4 Analysis of Device Performance

In this study, we think the suitable height from the bottom of the octagonal 9-KINECT imaging device to the ground is 3,000 millimeters. If the suitable height is not 3,000 millimeters, we can change the hollow cylinder of the top part. The vertical tilt angle of the outer 8 KINECT devices on the interchangeable bases ranges from -3^{o} to -57^{o}. We can change the range of the vertical tilt angle by changing the interchangeable base with the different tilt angle of the incline. But it should be noticed that the tilting device of the KINECT device won't work, when the vertical tilt angle of the KINECT device is smaller than –60°, because of the gravity sensor on the KINECT device. We would like to use the range of the vertical tilt angle from –25° to –55°.

3.4.1 Coverage of Views

With the height from the bottom of the octagonal 9-KINECT imaging device to the ground and the range of the vertical tilt angle, we can analyze the coverage of views of the octagonal 9-KINECT imaging device. We separate the analysis of the coverage of views into the color image side and the depth image side.

On the color image side, we use a single KINECT device to analyze the maximum and minimum sensing ranges of the field of view. The maximum sensing range is approximate 45,000 millimeters with a –25° vertical tilt angle of the KINECT device. A diagram illustrating this case is shown in Figure 3.7. The minimum sensing range is approximate 2,350 millimeters with the –55° vertical tilt angle of the KINECT device and an illustration diagram is shown in Figure 3.8.

We now analyze the coverage of views when all of the 9 KINECT devices are used. Because we want to have more overlapping views between the 9 KINECT devices to facilitate human model construction, we use the minimum sensing range. Also, we can use a circle whose diameter is approximate 6,730 millimeters to represent the coverage of views of the 9 KINECT devices from the top view, as can be figured out from the illustration diagram shown in Figure 3.9, in which the blue region is the view of the outer 8 KINECT devices, the red is the view of the inner KINECT device, and the yellow circle represents the coverage of views of the 9 KINECT devices.

On the depth images side, as we mentioned in the previous sections, the maximum sensing distance is 4 meters which is decided by the Kinect-for-Windows SDK provided by Microsoft. So the sensing range of the depth images is smaller than that of the color image, and an illustrative diagram is shown in Figure 3.10.

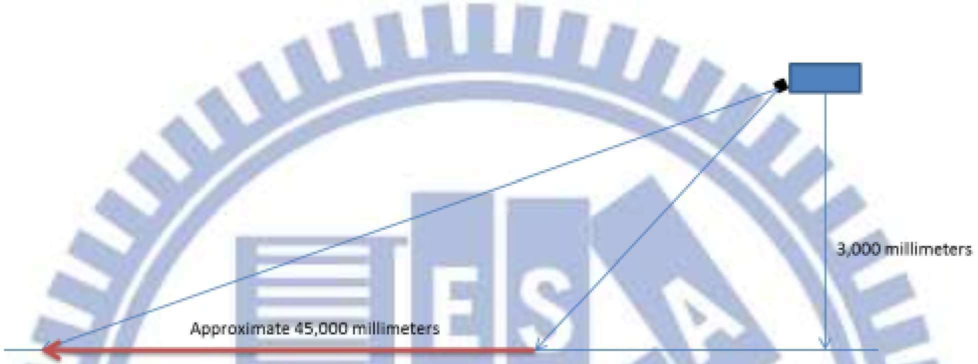


Figure 3.7 The maximum sensing range from the side view.

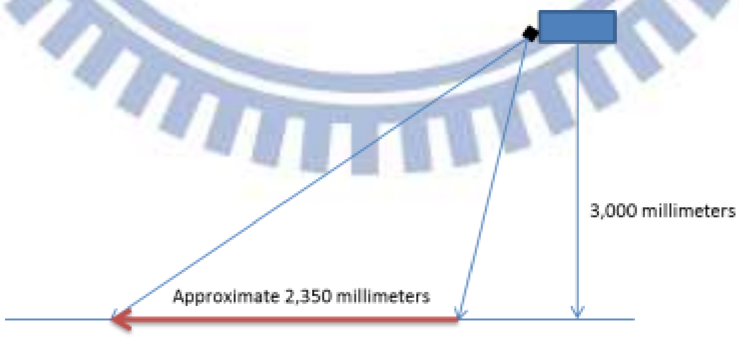


Figure 3.8 The maximum sensing range from the side view.

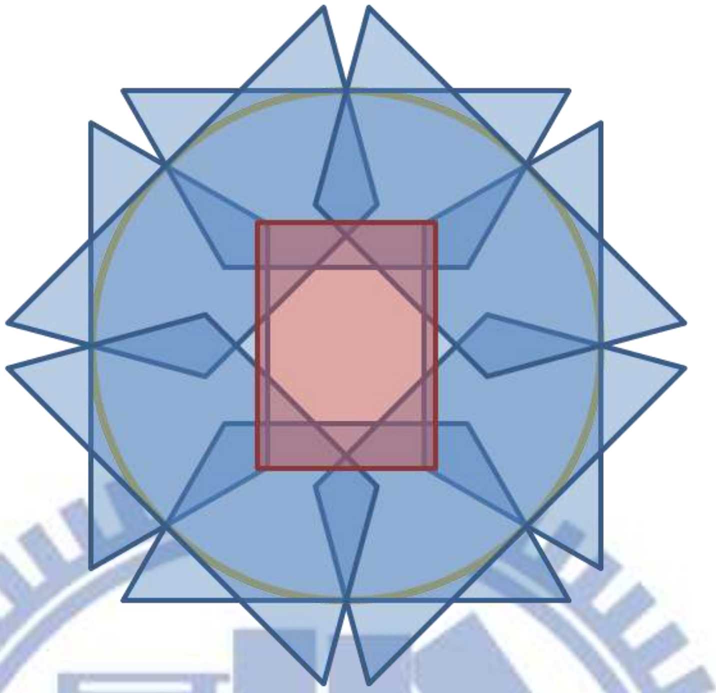


Figure 3.9 The coverage of views of 9 KINECT devices from top view. The blue region is the view of the outer 8 KINECT devices. The red is the view of the inner KINECT device. The yellow circle represents the coverage of views of the 9 KINECT devices.

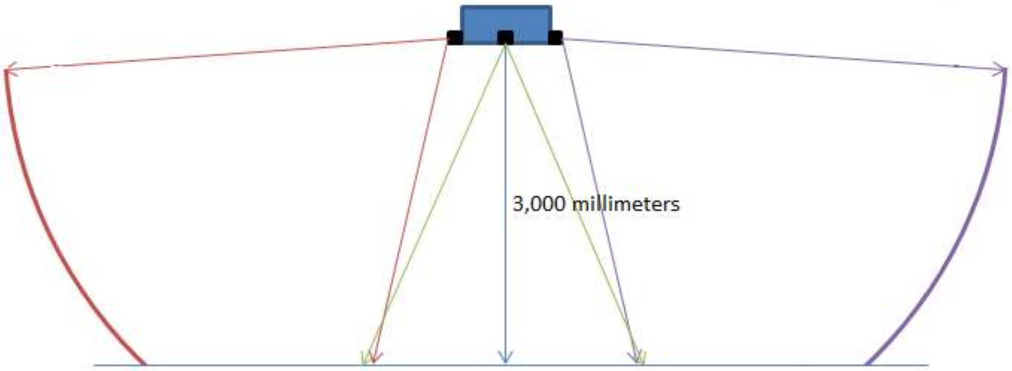


Figure 3.10 The coverage of the views by the depth image from side view.

3.4.2 Imaging Sequence and Speed

As we mentioned in the previous sections, we acquire the data from the 9 KINECT devices sequentially. When we take a frame consisting of a color image and a depth image from a single KINECT device, the frame rate of the device is 30 fps. In other words, we take a frame from a single KINECT device in 33 milliseconds. Then, when we use the 9 KINECT devices to take 9 frames sequentially, on the whole the imaging speed is $33 \times 9 = 297$ milliseconds, so the fps is $1/297 \approx 3.37$. But we assume that the monitored object or human moves not too fast, so it will not be a problem to our processing work.

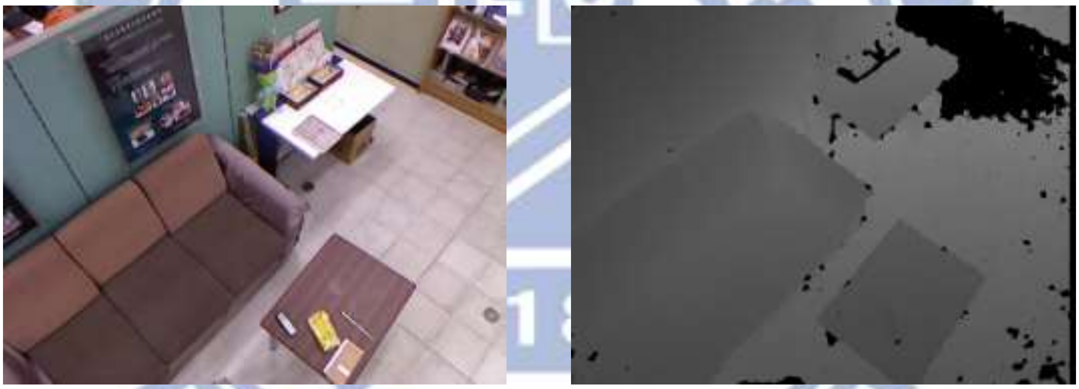
Chapter 4
Construction of 3D Images from KINECT Images

4.1 Introduction

The data acquired from a KINECT device each time consists of a color image and a depth image. We call them *KINECT images*. The KINECT images are not 3D in nature and so inconvenient for processing for 3D video surveillance applications. So, we want to construct a corresponding *3D image* from each pair of KINECT images. The 3D image contains three kinds of data. One is color data which come from the color image directly. Another is the 3D data which are obtained by converting the depth image into a 3D version. The third is a *mapping array*, which is obtained by using the Kinect-for-Windows SDK provided by Microsoft and is used as a tool for combining the former two parts, the 3D data and the color data. With the 3D image, we not only can conduct appropriate processing works required by a 3D video surveillance system more conveniently, but also can display results in 3D manners more easily.

4.2 Review of KINECT Image Structures

In this section, we will introduce the structure of the KINECT images in detail. As we mentioned in the last section, the KINECT images include a color image and a depth image. We use the KINECT device, which yields images with the resolution of 640×480 pixels, together with the Kinect-for-Windows SDK to get KINECT images. Each pixel in the color image has four bytes. The first three bytes are used to show the color and the last one is used to show the skeleton information. We can directly display the color image as a picture. An example of such color images is shown in Figure 4.1(a). Each pixel in the depth image has a value of an unsigned short integer. In other words, each pixel in the depth image has sixteen bits. The first thirteen bits are used to represent depth information and the last three bits are used to specify the skeleton information. We can display a depth image as a gray level image. An example of such depth images is shown in Figure 4.1(b).



(a) (b)

Figure 4.1 An example of a KINECT image pair. (a) The color image. (b) The depth image.

4.3 Construction of 3D Images

4.3.1 Review of Pinhole Camera Model

The *pinhole camera* [10] is a simple camera model. Its structure is an opaque box with an aperture of only the pinhole size on one side. The light reflected from the object and passing through the pinhole produces a projection of the scene in front of the pinhole. In the projection, right and left, and up and down are both reversed. An illustration is shown in Figure 4.2.

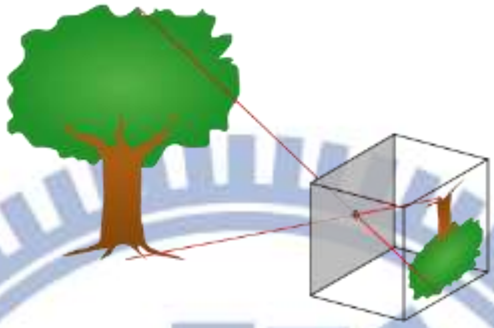


Figure 4.2 An illustration of the pinhole camera model.

The pinhole camera model describes the mathematical relationship between the coordinates of a 3D point and its projection on the image plane of the pinhole camera. An example of the geometry of the pinhole camera model is shown in Figure 4.3.

More specifically, in Figure 4.3(a), there is a 3D orthogonal coordinate system with its origin at *O*. The origin *O* is also the location of the camera aperture. The three axes of the 3D orthogonal coordinate system are referred to as X_1, X_2 and X_3. The X_3-axis is pointing in the viewing direction of the camera and is referred to as the *optical axis*. There is also a 2D coordinate system on the image plane with its origin at *R*. The origin *R* is at the intersection of the optical axis and is referred to as the *image center*. The two axes of the 2D coordinate system in the image plane are referred to as Y_1 and Y_2 which are parallel to the axes of X_1 and X_2, respectively. The distance from point *O* to *R* is *f*. The distance *f* is referred to as the *focal length* of the pinhole camera.

With the basic definitions given above, we can find out the relation between the point *P* with the 3D coordinates (x_1, x_2, x_3) and the projection point *Q* with the 2D coordinates (y_1, y_2). When we look in the negative direction of the X_2-axis from Figure 4.3(a), we get Figure 4.3(b). From the two similar triangles appearing in Figure 4.3(b), we can derive the following equation according to the similar-triangle principle:

$$\frac{-y_1}{f} = \frac{x_1}{x_3}. \tag{4.1}$$

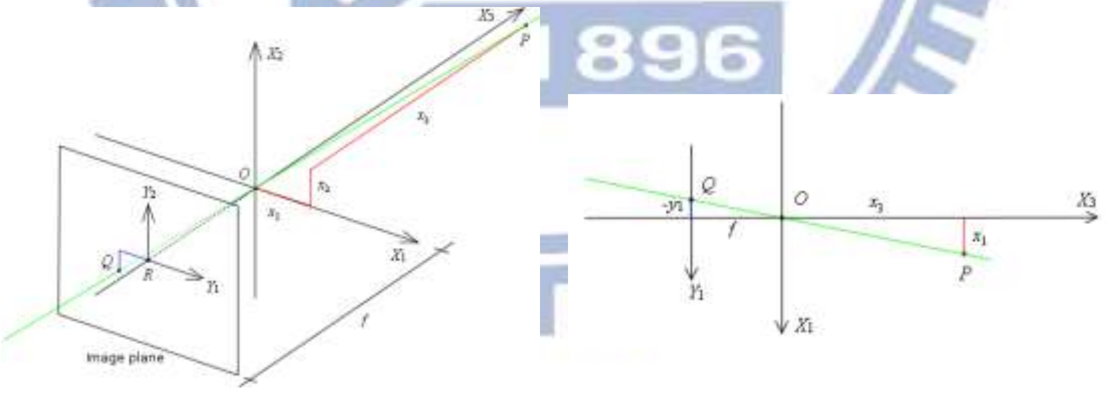
When we look in the negative direction of the X_1-axis, the following equation can be derived similarly:

$$\frac{-y_2}{f} = \frac{x_2}{x_3}. \tag{4.2}$$

Summarizing these two equations, we get the following vector equation:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = -\frac{f}{x_3}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{4.3}$$

which describes the relation between point P with the 3D coordinates (x_1, x_2, x_3) and the projection point Q with 2D coordinates (y_1, y_2).



(a) (b)

Figure 4.3 The geometry of a pinhole camera model. (a) Seen from a 3D point. (b) Seen from the X_2-axis.

4.3.2 Idea of 3D Image Construction and Coordinate Conversion

With the concept of the pinhole camera model, we can construct the 3D image from the depth image of the KINECT image by coordinate conversion. We will use Figure 4.3 to help us to explain the conversion process. From Equation 4.3, we get:

$$x_1 = -\frac{x_3}{f} \times y_1 ; \tag{4.4}$$

$$x_2 = -\frac{x_3}{f} \times y_2 ; \tag{4.5}$$

$$x_3 = \frac{x_3}{f} \times f \ . \tag{4.6}$$

and from Figure 4.3(a) and by the similar-triangle principle again, we have the equation:

$$\frac{x_3}{f} = \frac{\sqrt{x_1{}^2 + x_2{}^2 + x_3{}^2}}{\sqrt{(-y_1)^2 + (-y_2)^2 + f^2}} \tag{4.7}$$

where $\sqrt{(-y_1)^2 + (-y_2)^2 + f^2}$ is the length of the line segment $\overline{OQ}$, and $\sqrt{x_1{}^2 + x_2{}^2 + x_3{}^2}$ is the length of the line segment $\overline{OP}$ which is the depth captured by the KINECT device and is denoted as *d* in the sequel. Let *R* represent the center of the depth image, which is located at coordinates (320, 240) in a depth image of resolution 640×480 acquired by the KINECT device. And let *Q* be a pixel located at image coordinates (x_p, y_p), and let y_1 and y_2 represent the distance to the center *R* in the vertical and horizontal directions, respectively. The focal length *f* of the KINECT device is 600. The equations (4.7), (4.4), (4.5) and (4.6) can be rewritten according to the mentioned parameter values to be:

$$\frac{x_3}{f} = \frac{d}{\sqrt{(x_p - 320)^2 + (y_p - 240)^2 + 600^2}} ; \tag{4.8}$$

$$x_1 = \frac{d}{\sqrt{(x_p - 320)^2 + (y_p - 240)^2 + 600^2}} \times (x_p - 320); \tag{4.9}$$

$$x_2 = \frac{d}{\sqrt{(x_p - 320)^2 + (y_p - 240)^2 + 600^2}} \times (y_p - 240); \tag{4.10}$$

$$x_3 = \frac{d}{\sqrt{(x_p - 320)^2 + (y_p - 240)^2 + 600^2}} \times 600. \tag{4.11}$$

The unit of x_p and y_p is pixel and that of x_1, x_2 and x_3 is millimeter. With the above equations, we can convert the depth image of the KINECT image into a 3D image. The color data of the 3D image uses the color image acquired from the KINECT device directly. The mapping array can be produced by using the Kinect-for-Windows SDK, provided by Microsoft, with the depth image of the KINECT image.

4.3.3 Construction Algorithm

With the required data for constructing the 3D image ready, we can use the construction algorithm to construct the 3D image. A flowchart of the 3D image construction algorithm is shown in Figure 4.4. The detail of the construction algorithm is as follows.

Algorithm 4.1: 3D image construction.

Input: a depth image I_d and a color image I_c acquired from the KINECT device.

Output: a 3D image I_3D formed from converted I_d and original I_c combined with a mapping array A.

Steps:

Step 1. Convert the coordinates of the depth image I_d into 3D data by the coordinate conversion process described in Section 4.3.2.

Step 2. Use the Kinect-for-Windows SDK provided by Microsoft with the converted depth image Id as input to get a mapping array A.

Step 3. Display the 3D image I3D by drawing 3D data in the 3D space with the corresponding color, which is produced by the color image Ic and the mapping array A, by the OpenGL.

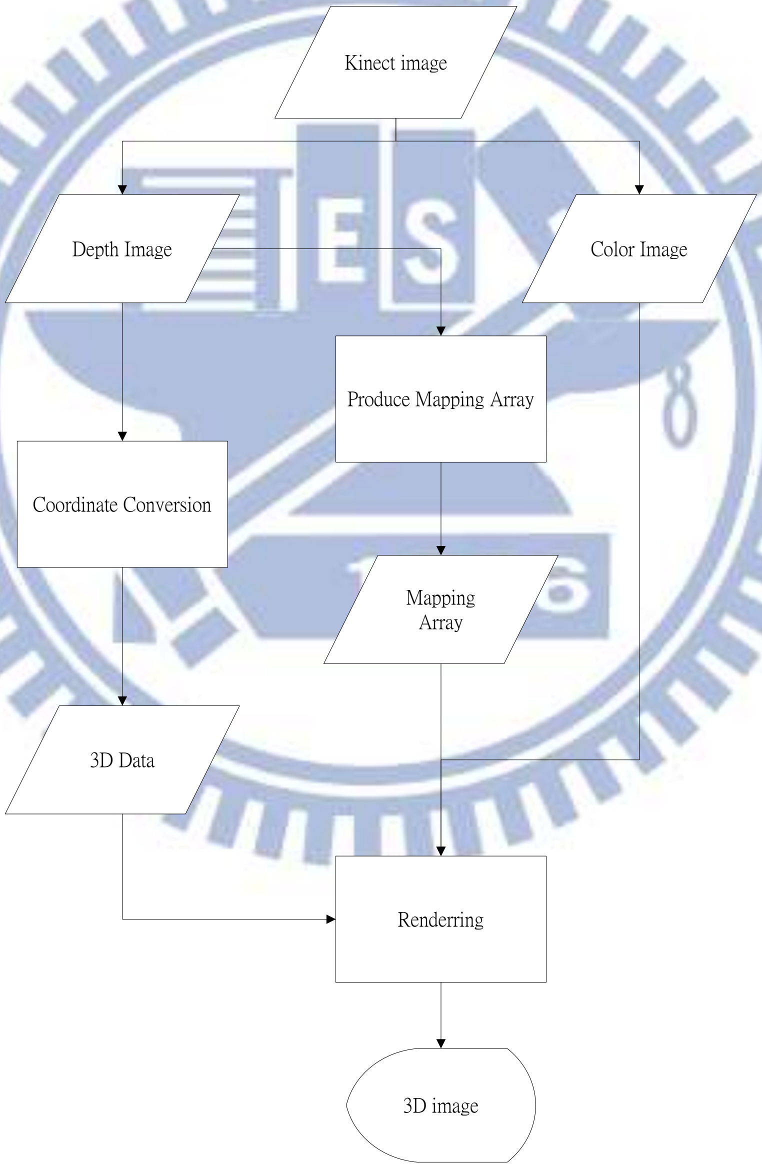


Figure 4.4 The flowchart of the 3D image construction algorithm.

4.4 Geometric Correction of 3D Images

4.4.1 Need of Correction

In our experiments of this study, when we displayed a plane in the 3D image, we discovered that the plane becomes a curved surface rather than a flat one. An example of such a phenomenon is shown in Figure 4.5. The reason why this problem arises is that the infrared light rays sent out by the KINECT device for depth sensing do not go in parallel. It affects the accuracy of the depth because the depth is not a vertical distance anymore. In order to solve this problem, we propose a method for the geometric correction of 3D images.

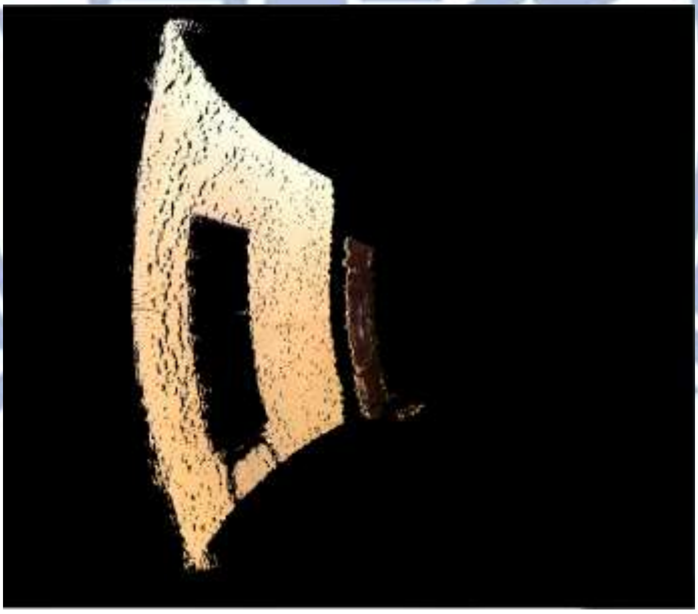


Figure 4.5 The 3D image of a plane (a wall).

4.4.2 Proposed Correction Technique by Interpolation

The idea of the proposed correction technique is to use a paraboloid to approximate the curved surface formed by the 3D data of the 3D image. An illustration is shown in Figure 4.6. When the paraboloid equation is found, we can substitute the values of the coordinates *x* and *y* of each pixel of the 3D image into the paraboloid equation to compute the correction distance. Then, we subtract the correction distance from the value of the coordinate *z* of the 3D data of the 3D image, and we get the correction result.

But we discovered that when we use the KINECT device to sense planes with the different distances from the KINECT device, the degrees of the curvature for the curved surfaces formed by the 3D data of the 3D image are also different. So we try to find several paraboloid equations with different sensing distances from the KINECT device. And we use these paraboloid equations according to the value of the coordinate *z* of the 3D data of the 3D image to find suitable correction distances by the interpolation.

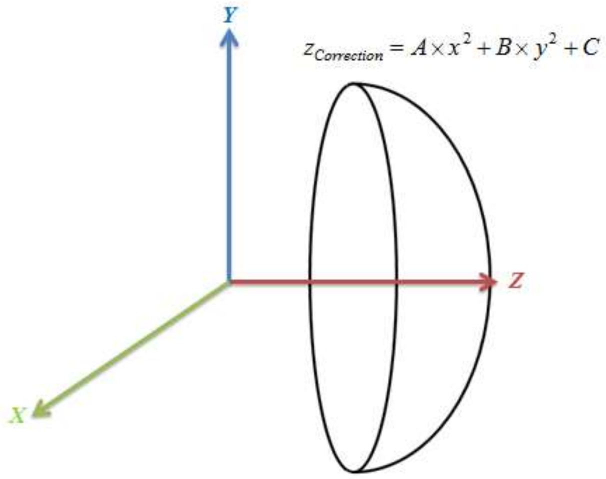


Figure 4.6 The paraboloid equation.

4.4.3 Correction Algorithm

In this section, we will describe the method for getting the paraboloid equation and the process of the interpolation mentioned previously. The criterion of minimum sum of squared errors (MSSE) is used to decide the parameters of the approximating shape. That is, we will use the MSSE criterion to approximate the paraboloid. The detail of the process is as follows.

First, let the paraboloid equation be described by:

$$z_{Correction} = A \times x^2 + B \times y^2 + C \tag{4.12}$$

where C is the distance between the KINECT device and the apex of the paraboloid, as shown in Figure 4.6. The equation for computing the value SSE of the SSE is:

$$SSE = \sum_{i=0}^{640\times480} \left[z_i - \left(A \times x_i^2 + B \times y_i^2 + C \right) \right]^2 \tag{4.13}$$

where x_i, y_i and z_i are the coordinates of a set of sample 3D data of the curved surface. To find the coefficients A, B, and C which minimize the SSE value, we compute the partial derivatives of Equation (4.13) with respect to variables A, B, and C, respectively, to produce the following equations:

$$2 \times \sum_{i=0}^{640\times480} \left[z_i - \left(A \times x_i^2 + B \times y_i^2 + C \right) \right] \times \left(-x_i^2 \right) = 0\,; \tag{4.14}$$

$$2 \times \sum_{i=0}^{640\times480} \left[z_i - \left(A \times x_i^2 + B \times y_i^2 + C \right) \right] \times \left(-y_i^2 \right) = 0; \tag{4.15}$$

$$2 \times \sum_{i=0}^{640\times480} \left[z_i - \left(A \times x_i^2 + B \times y_i^2 + C \right) \right] \times \left(-1 \right) = 0\,. \tag{4.16}$$

The values of A, B, and C are computed by solving the simultaneous equations (4.14), (4.15) and (4.16). For this, we substitute all the known values of x_i, y_i and z_i into the simultaneous equations (4.14), (4.15) and (4.16) to get three three-variable linear equations. We use the substitution and elimination method to solve three three-variable linear equations. After solving these three independent equations, we can get the values of A, B, and C.

But, as mentioned we need more solutions of the values A, B, and C by using different sets of 3D data of the 3D image with different distances between the

KINECT device and the planes, and the results are shown in Table 4.1.

Table 4.1 Results of paraboloid coefficient estimations using different sets of 3D data of the 3D image with the different distances between the KINECT device and planes.

coefficient distance	A	B	C
1003.62 (mm)	0.000495949	0.000499793	-1003.62
1535.92 (mm)	0.000323624	0.000380348	-1535.92
2120.88 (mm)	0.000232773	0.000281706	-2120.88
2560.30 (mm)	0.000188799	0.000248159	-2560.30
3111.78 (mm)	0.000155055	0.000205877	-3111.78

With Table 4.1, we can use it to decide which equations we will use to do interpolation by the value of the coordinate z of the 3D data of the 3D image. When the equations are found, we subtract the value of C of the equations themself from the equations to get correction equations. Then, we substitute the values of the coordinates x and y of the 3D data of the 3D image into the correction equations to get correction distances. We use the correction distances and the values of the coordinates z of the 3D data of the 3D image to do the interpolation by the proration principle and get the result of the interpolation. Finally, we subtract the result of the interpolation from the value of the coordinate z of the 3D data of the 3D image to get the correction result. A flowchart of the correction algorithm is shown in Figure 4.7 and the detail of the correction algorithm is as follows.

Algorithm 4.2: correction algorithm.

Input: the values of the coordinates x, y, and z of the 3D data of the 3D image.

Output: the correction value $z_{Corrected}$.

Steps:

Step 1. Use the values of the coordinate z to find the paraboloid equations *PE*s.

Step 2. Subtract the values of C from the paraboloid equations PEs themselves and get correction equations CEs.

Step 3. Substitute the values of the coordinates x and y into the correction equations CEs to get the solutions $z_{Correction}$s.

Step 4. Use the solutions $z_{Correction}$s and the values of the coordinate z to do interpolation and get the result $z_{Interpolaton}$.

Step 5. Subtract the result $z_{Interpolaton}$ from the values of the coordinate z and get final corrected value $z_{Corrected}$.

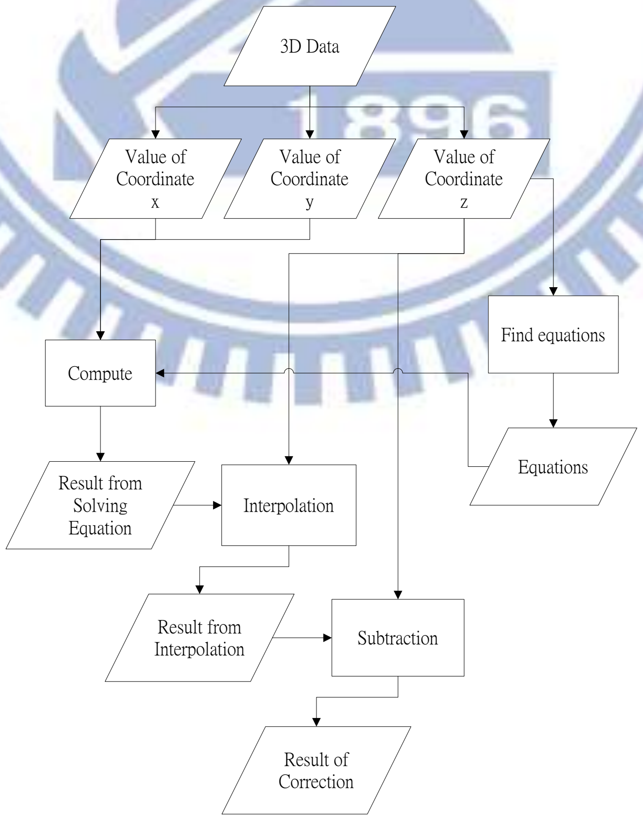


Figure 4.7 A flowchart of the correction algorithm.

4.5 Experimental Results

4.5.1 Results of 3D Image Construction

We use the KINECT image to construct 3D images and display them by the OpenGL. An example of the results of 3D image construction is shown in Figure 4.8.

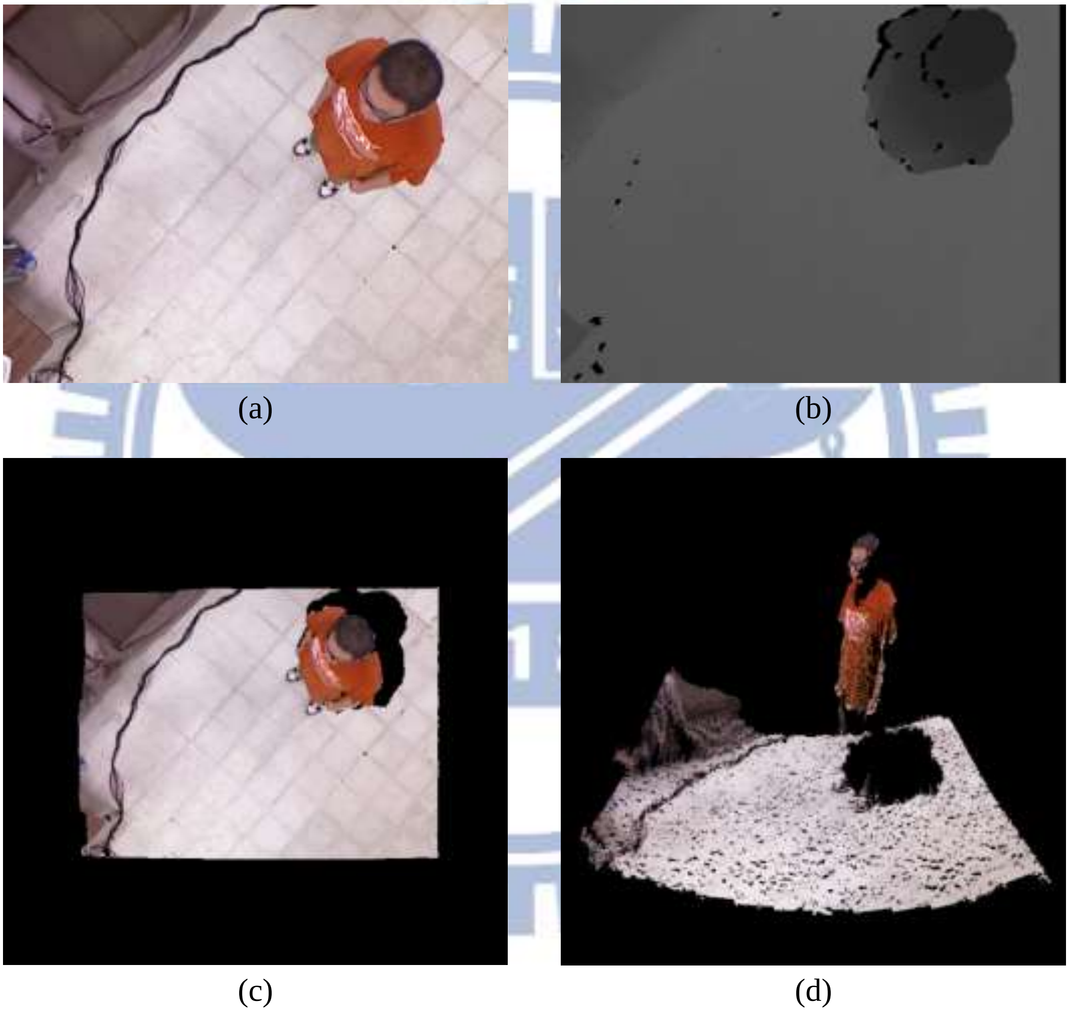
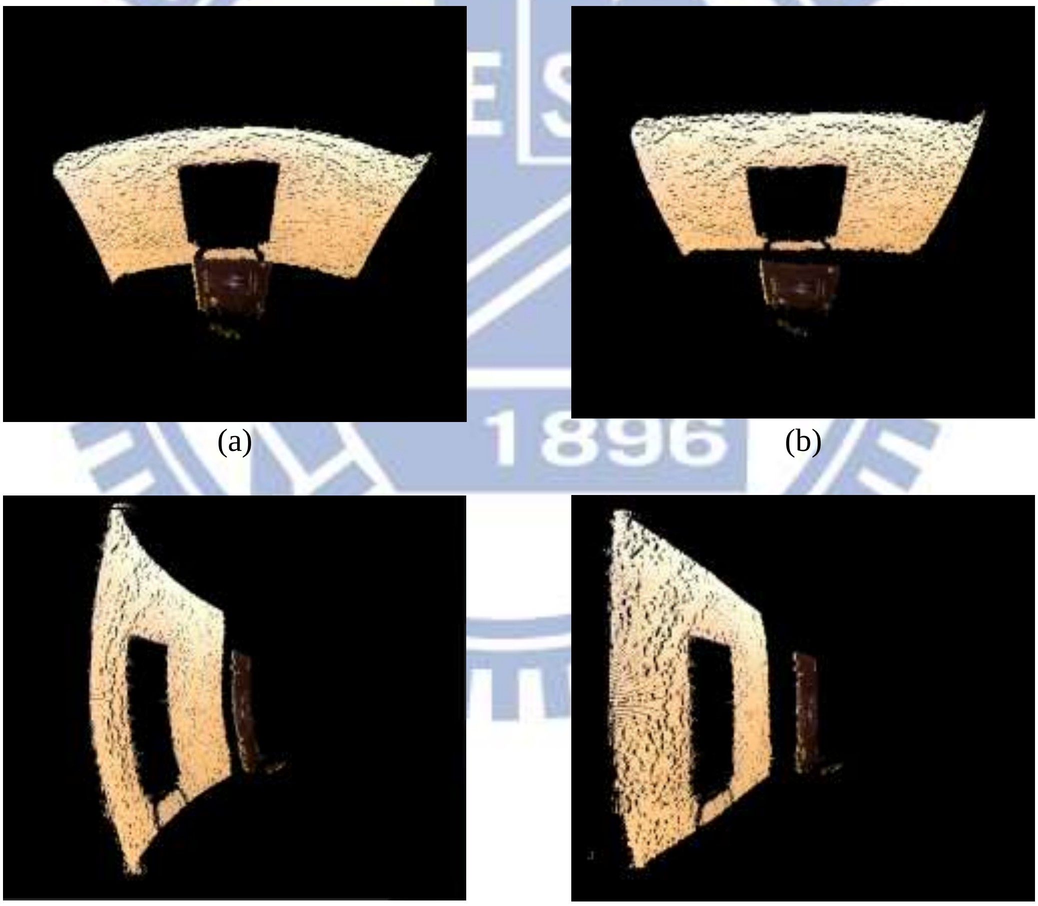


Figure 4.8 An example of construction of 3D images. (a) The color image of the KINECT image. (b) The depth image of the KINECT image. (c) The 3D image seen from the top. (d) The 3D image seen from a side.

4.5.2 Results of 3D Image Correction

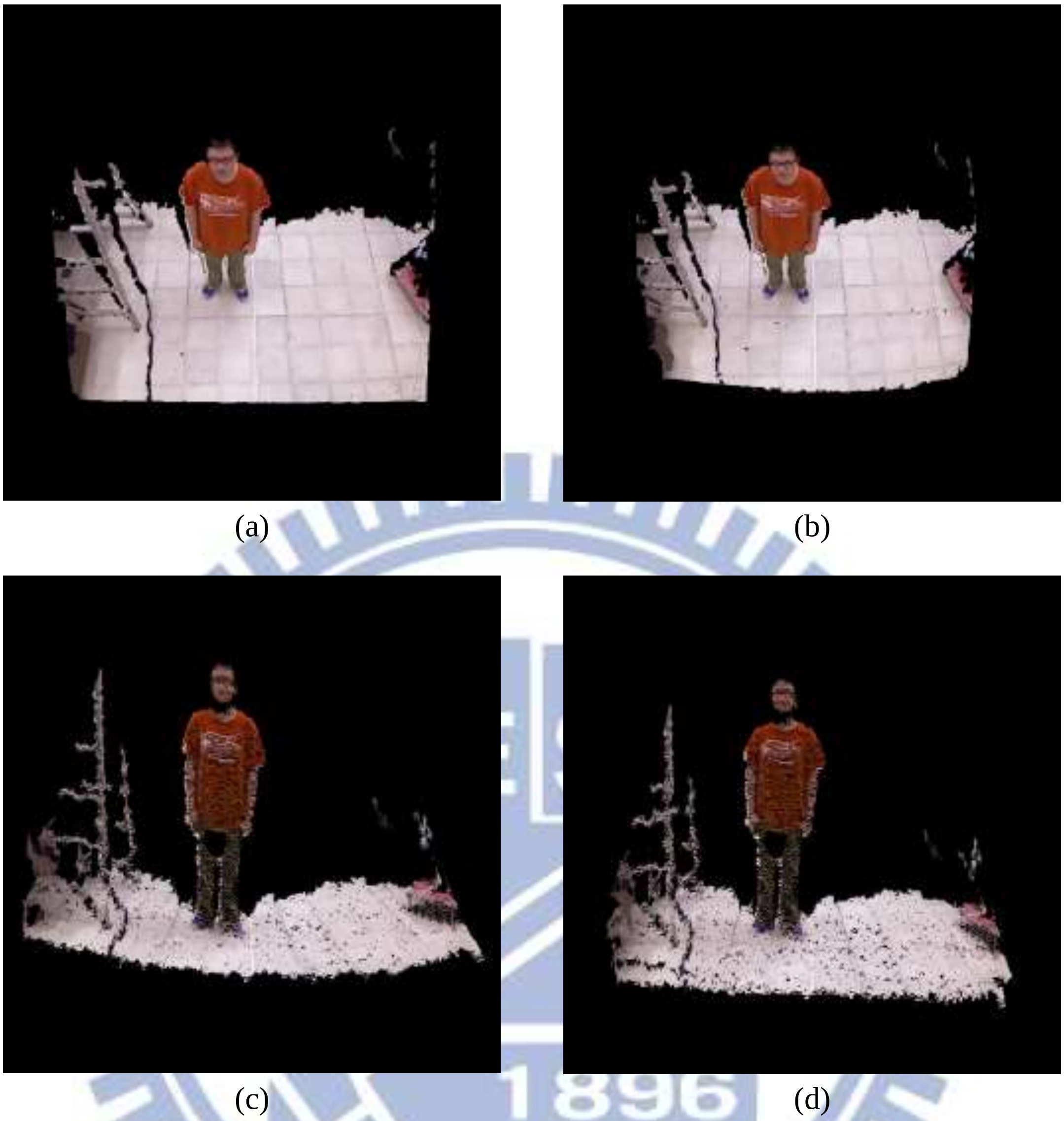
We use the correction algorithm to correct the 3D data of the constructed 3D image and display the result by the OpenGL. But there is still a problem. That is, the corners of the 3D image are still curved irregularly. For this, on solution is to avoid the use of the 3D data of the corners of the 3D image. An example of the results of such geometric corrections for planes is shown in Figure 4.9. Another example of the results of such geometric corrections for an indoor environment is shown in Figure 4.10.



(a) (b)

(c) (d)

Figure 4.9 Results of geometric correction. (a) Original data seen from the top before correction. (b) Data seen from the top after correction. (c) Original data seen from the side before correction. (d) Data seen from the side after correction.



(a) (b)

(c) (d)

Figure 4.10 Results of geometric correction. (a) Original data seen from the top before correction. (b) Data seen from the top after correction. (c) Original data seen from the front before correction. (d) Data seen from the front after correction.

Chapter 5
Construction of 3D Indoor Environment Model from Multiple KINECT Images

5.1 Introduction

In this chapter, we describe how we construct the indoor environment model for 3D video surveillance using images acquired by the octagonal 9-KINECT imaging device. More specifically, we use the nine KINECT devices to get nine sets of KINECT images and convert them into nine 3D images individually. Then, we merge the nine 3D images to build up an indoor environment model. But, before doing so, we should calibrate the spatial relation between the nine KINECT devices in advance. The detail of the proposed calibration technique will be described in Section 5.2. After the calibration work, we use the results to merge the nine 3D images by shifting, rotating, and merging them to build up the indoor environment model. Finally, we display the model in 3D manners. The details of data merging and model displaying will be shown in Sections 5.3 and 5.4, respectively.

5.2 Calibration of KINECT Devices

5.2.1 Review of Iterative Closest Point (ICP) Algorithm

The iterative closest point (ICP) algorithm [11] can be employed to minimize the difference between two groups of points. It is often used to match objects, which are constructed by many points, to compute their similarity. It is useful for constructing 2D or 3D images from different views, because object registration or stitching requires shape matching.

The concept of the algorithm is simple. It iteratively revises the transformation, including translation and rotation, from an object into another in order to minimize the total distance between the points of the two objects. The algorithm is as follows.

Algorithm 5.1: ICP algorithm

Input: a group of points G_A, another group of points G_B, a set of transformations T_is, an initialized minimum value M, and an initialized transformation T_0.

Output: A transformation T which is the relation between group G_A and group G_B.

Steps:

Step 1. Apply a transformation T_i, which is not used yet, to all points in group G_B.

Step 2. Find points P_{MD}s with the minimum distance in group G_A for each point in group G_B.

Step 3. Compute the values V_{MD}s of the minimum distance between the found points P_{MD}s in group G_A and the corresponding points in group G_B.

Step 4. Sum up the values V_{MD}s to get a total sum T_S.

Step 5. If the total sum T_S is small than the input minimum value M, update the minimum value M with the total sum T_S and the desired transformation T with the transformation T_i.

Step 6. Repeat Step 1 through Step 5 if the transformations T_is are not exhausted yet.

Step 7. Take the last updated transformation *T* as the output.

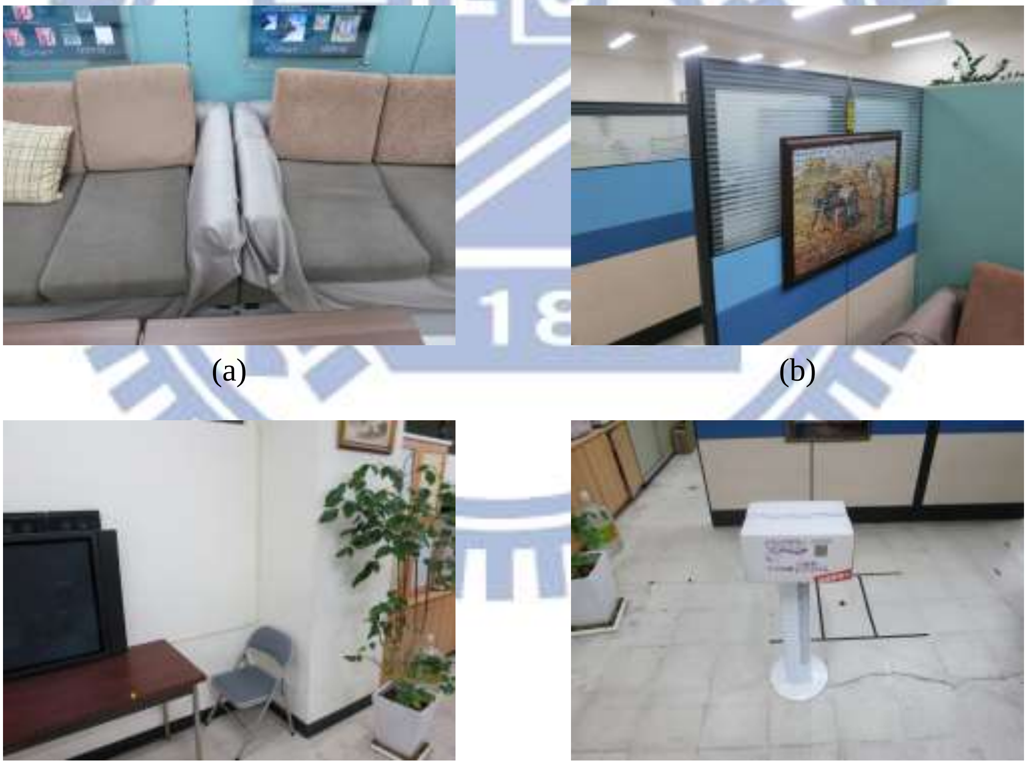
5.2.2 Calibration of Spatial Relation between KINECT Devices

In this section, we want to use the ICP algorithm to calibrate the spatial relations of the nine KINECT devices in the octagonal 9-KINECT imaging device. By using the ICP algorithm to merge the 3D images of two objects which are the same object but come from two different KINECT devices, we can get the result of the transformation between them, which is just the spatial relation of the two KINECT devices, because the transformation between 3D images is equivalent to the transformation between KINECT devices. With the concept above, we should prepare three things before starting calibration.

First, we should decide the range of the transformation parameters, and for this, we divide the transformation into two parts — a rotation and a translation. For the rotation, because the sensing directions of the nine KINECT devices of the octagonal 9-KINECT imaging device are fixed, the angles between the nine KINECT devices are also fixed. We can use the values of these angles for the rotation. For the translation, we divide it into two directions to facilitate running the ICP algorithm. The place of each of the nine KINECT devices is fixed, so the distance between every two of the nine KINECT devices is also fixed. We would like to enlarge values of these distances and divide these distances into two directions for the translation of the two directions.

Second, we should find out the overlap region of the 3D images acquired from every two KINECT devices. Using the overlap region, we can merge the 3D images of an identical object “seen” from different KINECT devices by the ICP algorithm in order to get the result of the transformation. The overlap regions may be found by manpower.

Third, we should choose objects, whose 3D images from different KINECT devices can be merged in the overlap regions, and we will call them *calibration targets*. Basically, we should use a calibration target which is big enough and can appear in the overlap region apparently. For this, we use common objects which appear in the indoor environment as calibration targets, such as couch, table, chair, clapboard, etc. Sometimes, we will also use a box which is put at suitable height as the calibration target, if there is no apparent object in the overlap region. Some calibration targets are shown in Figure 5.1.



(a) (b)

(c) (d)

Figure 5.1 Some calibration targets used in this study. (a) A couch. (b) A clapboard. (c) A chair and a table. (d) A box with a suitable height.

5.2.3 Algorithm for KINECT Device Calibration

With the preparation done, we start to calibrate the spatial relations between the nine KINECT devices in the octagonal 9-KINECT imaging device. Firstly, we label the nine KINECT devices by numbers, and two consecutively numbered KINECT devices mean that they are neighboring. Then, we use the 3D images, which include the pre-selected calibration target in their overlap region, to calibrate the inter-KINECT relation parameters by the ICP algorithm. Totally, we conduct such calibration for eight times.

Before we conduct such calibrations each time, we reset the range of the possible transformations between the two devices for the ICP algorithm, set the two 3D images including the calibration target from two neighboring KINECT devices as inputs to the ICP algorithm, and use the overlap region in the images to assist the calibration work. The proposed algorithm for KINECT-device calibration is as follows.

Algorithm 5.2: KINECT device calibration.

Input: the 3D images CT_0, CT_1, …, CT_8 which are constructed from KINECT images acquired by the nine KINECT devices D_0, D_1, …, D_8 and include the calibration target; the transformation NT_j and the overlap region OR_j between every two neighboring KINECT devices D_j and D_{j+1} where $j = 0, 1, …, 7$; a counter with its value C set to be 0 initially.

Output: the transformation R_k between every two KINECT devices D_k and D_{k+1}, where $k = 0, 1, …, 7$, which can be used to “register” the 3D images CT_k and CT_{k+1}.

Steps:

Step 1. Take two 3D images CT_c and CT_{c+1}, which include the calibration target in their overlapping region, as input data for the ICP algorithm.

Step 2. Set the transformation NT_c to be the transformation sets for the ICP algorithm.

Step 3. Start the ICP algorithm described in Section 5.2.1 while using the overlap region OR_c to assist finding the calibration target for the ICP algorithm.

Step 4. Store the result of transformation of the ICP algorithm as the result of the transformation R_c.

Step 5. Increment the value C of the counter by 1.

Step 6. If the value C is smaller than eight, then repeat Steps 1 through 5; else, exit.

5.3 Environment Model Construction

5.3.1 Idea of Construction

After calibrating the spatial relations between the nine KINECT devices in the octagonal 9-KINECT imaging device, we can get eight transformations between the nine KINECT devices. As we mentioned previously, a transformation between two 3D images is equivalent to the transformation between the two corresponding KINECT devices, and vice versa. So we will use the results of the transformation to “register” the nine 3D images, which are constructed from KINECT images acquired by the nine KINECT devices. By doing so, we can merge the nine 3D images into one to construct the indoor environment model.

5.3.2 Merge of Multiple 3D Images

In Section 5.2.3, we label the nine KINECT devices by numbers. It means that we also label the nine 3D images by numbers which are the same as the numbers of the nine KINECT devices. We then merge the nine 3D images sequentially according to the numbers. We use the first 3D image as a pivot and the others are merged into it, and so to each of the last eight 3D images, more transformations should be applied. The merging processing will be run eight times. The result from merging the nine 3D images is just an indoor environment model which we desire. The merging algorithm is as follows.

Algorithm 5.3: merging nine 3D images.

Input: nine 3D images IS_0, IS_1, …, IS_8 constructed from images acquired by the nine KINECT devices D_0, D_1, …, D_8 respectively; eight transformations RT_0, RT_1, …, RT_7 from the calibration results where RT_i represents the spatial relation between the two KINECT devices D_i and D_{i+1} and i = 0, 1, …, 7; a counter with its value C set to be 0 initially.

Output: the merging result MR.

Steps:

Step 1. Use the 3D image IS_0 as the pivot.

Step 2. Put the 3D image IS_0 into the merging result MR.

Step 3. Merge the transformations RT_0, RT_1, …, RT_C and get a merged transformation called MRT_C.

Step 4. Apply the transformation MRT_C to the 3D image IS_{C+1} and put the result TIS_C into the merging result MR.

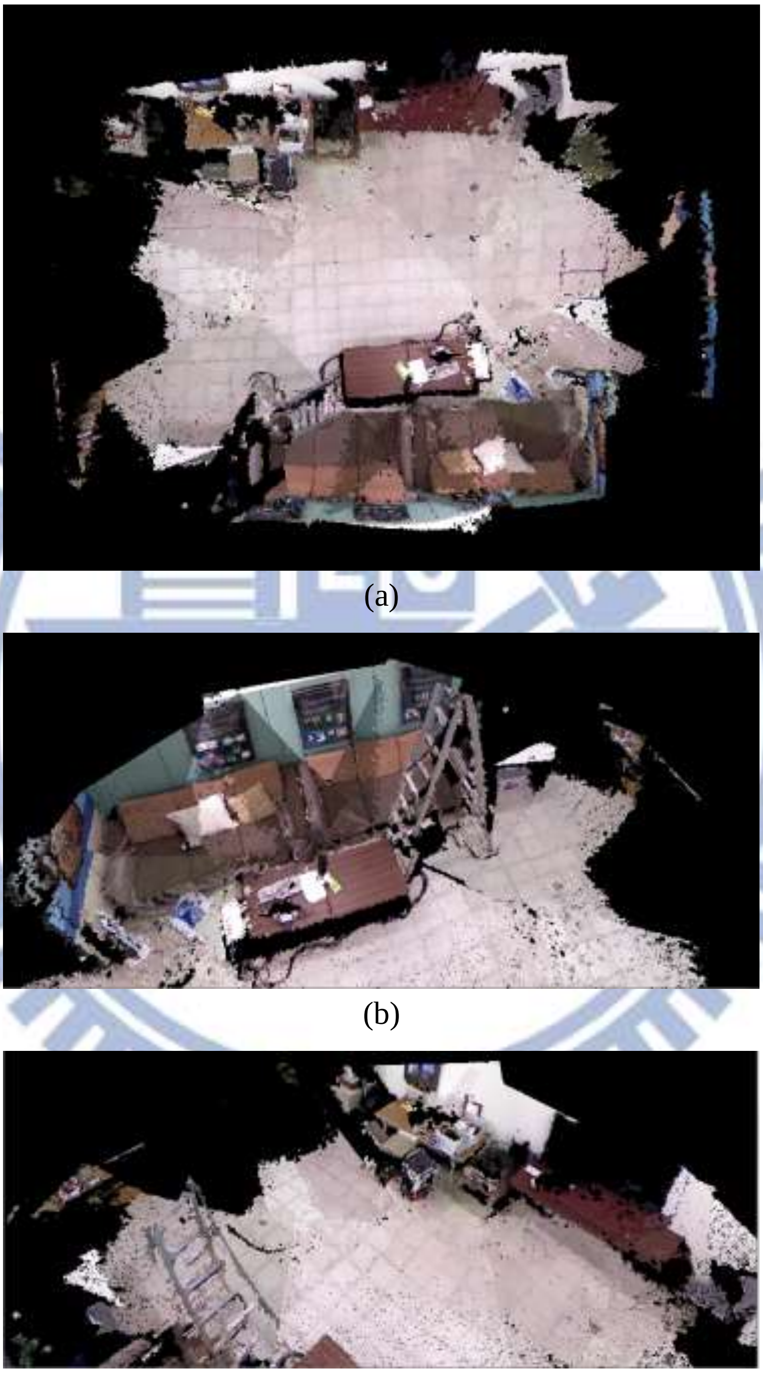
Step 5. Increment the value C of the counter by 1.

Step 6. If the value C is smaller than eight, then repeat Steps 3 through 5; else, go to the next step.

Step 7. Take the final merging result MR as the desired indoor environment model.

5.4 Experimental Results

The result of indoor environment modeling by merging the nine 3D images acquired from the nine KINECT devices of the octagonal 9-KINECT imaging device is shown in Figure 5.2.



(a)

(b)

(c)

Figure 5.2 The constructed indoor environment model. (a) The indoor environment model seen from the top. (b) and (c) The indoor environment model seen from different views.

Chapter 6
Human Tracking by Tilting KINECT Devices

6.1 Introduction

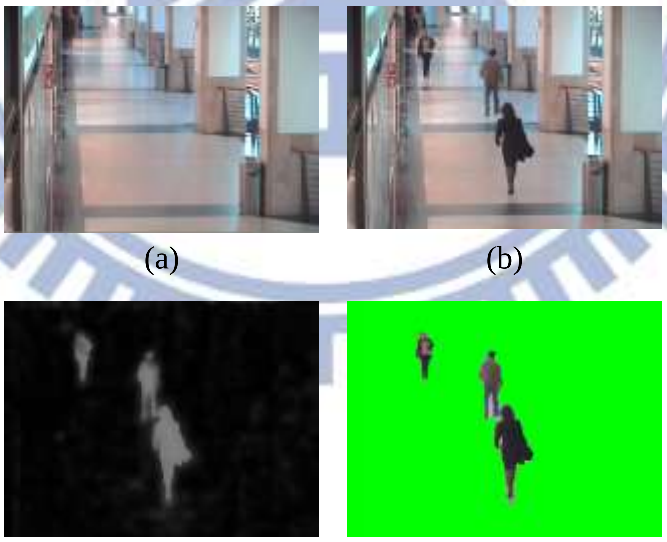
In this chapter, we will introduce the proposed human tracking method by using the octagonal 9-KINECT imaging device for 3D video surveillance system. To track human activities, we should detect the human first. So we will separate the subject into two parts: human detection and human tracking.

In the human detection part, the depth image acquired by the KINECT device may be considered also as a kind of image like gray level image, so we may apply some method of motion object detection, which have been used for the color image, to the depth image to conduct human detection. For this, at first we use the background subtraction technique to detect moving objects in the depth image. Then, we use a noise reduction scheme to reduce noise in the resulting image. Finally, we apply a region growing scheme with a suitable threshold to the image resulting from noise reduction, and get the whole moving object in the depth image as the result. The detail of the proposed detection algorithm will be described in Section 6.2.

In the human tracking part, by analyzing the moving object in two consecutive frames of the depth images, we can know where the object will go and how large the distance the object moves in the two images. Accordingly, we can adjust the tilt angle of the KINECT devices in the 9-KINECT imaging device to track the object or do the handoff between KINECT devices. The detail of the proposed human tracking process will be introduced in Section 6.3. And the experimental result will be shown in Section 6.4.

6.1.1 Review of Background Subtraction

The background subtraction is a technique commonly used in the fields of image processing and computer vision for object segmentation. We can use the background subtraction technique to separate the foreground of the image from its background because when we read the image, we are usually interested in the objects in the foreground of the image. The background subtraction technique is also a widely used approach for detecting moving objects in videos acquired from static cameras. The basic approach is to detect moving objects from the difference between the frame including moving objects and a reference frame often called the background image. An example of background subtraction results is shown in Figure 6.1.



(a) (b)

(c) (d)

Figure 6.1 An example for the background subtraction. (a) The background image. (b) An image with moving objects. (c) The image of the difference between (a) and (b) with some noise. (d) The resulting image of background subtraction.

6.1.2 Review of Noise Reduction Method

There are many methods to reduce noise in the image. Mathematical morphology operations are often used to assist reducing noise in the image. Mathematical morphology is a theory for analysis and processing of geometrical structures. It is most commonly applied to digital images. Mathematical morphology has two basic operators. One is the erosion operator and the other is the dilation operator.

Before explaining the two operators, we should define some variables for input data. We use the variable *A* as the input image and use the variable *B* as the structuring element. The structuring element is a binary image with a simple and pre-defined shape but smaller than the input image. We also use the variable *a* as the pixel of the input image *A* and use the variable *b* as the pixel of the structuring element *B*. With the definitions above, we start to describe the two operators.

For the erosion operator, the erosion of the input image *A* by the structuring element *B* is defined by:

$$A\ominus B = \bigcap_{b\in B} A_{-b}\,. \tag{6.1}$$

When the structuring element *B* has a center and this center is located on an origin, then the erosion of *A* by *B* can be understood as the locus of points reached by the center of *B* when *B* moves inside *A*. An example of the results of applying the erosion operator is shown in Figure 6.2.

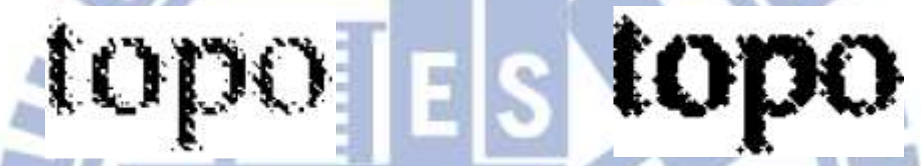
(a) (b)

Figure 6.2 An example for erosion results. (a) The original image. (b) The image after erosion.

For the dilation operator, the dilation of the input image *A* by the structuring element *B* is defined by:

$$A \oplus B = \bigcup_{b \in B} A_b \, . \tag{6.2}$$

If the structuring element *B* has a center on the origin, then the dilation of *A* by *B* can be understood as the locus of the points covered by *B* when the center of *B* moves inside *A*. An example of the respectively of applying the dilation operator is shown in Figure 6.3.



(a) (b)

Figure 6.3 An example for dilation operator. (a) The original image. (b) The image after dilation operator.

From Figure 6.2, the thin parts of the object in the original image disappear in Figure 6.2 (b) after the erosion operator is applied. From Figure 6.3, the thin parts of the object in the original image can be seen to get thicker in Figure 6.3 (b) after the dilation operator is applied. So we can use the erosion operator to reduce noise in the image and use the dilation operator to restore the lost parts of objects which are produced by the erosion operator.

6.2 Human Detection

6.2.1 Background Learning

To use the background subtraction technique, we should conduct background learning of the indoor environment, which is the experimental place in this study, and get the 3D image of the background first. However, when we used a KINECT device to sense a static region, we discovered that the same locations of a pixel in two consecutive depth images of the KINECT image acquired from the same KINECT device are sometimes different. One has a value of depth information, but the other has no value of depth information. This problem comes from the infrared light rays sent out by the KINECT device. Because the reflective path of the infrared light rays will be interfered in some situations, the total amount of reflective infrared light rays will be different for different times of depth information detection and will also affect the production of the depth image indirectly.

To solve the problem and complete the background learning, we use a KINECT device to sense a static region for a while to get a multiple of KINECT images. Then, we average the values of all the pixels in the depth images at the same locations to get an *average depth image*. Also, we choose one color image from the acquired color images as the background color image. And finally, we choose one mapping array from the multiple ones produced by the Kinect-for-Windows SDK as the final mapping array, which will be used for constructing the 3D image of the background. With the three measures above, we can construct a 3D image of the background. Because we want to use the depth image to do background subtraction technique actually, we won’t construct the 3D image of the background immediately but regard the results of the three measures as a set of the *3D image constructor* of the background.

But when the vertical tilt angle of the KINECT device is changed, the field of view of the KINECT device is also changed. So we should redo the background learning with different vertical tilt angles of the KINECT device. Because the vertical tilt angle ranges from －25^{o} to －55^{o}, we do the background learning by the increment steps of 2 degrees of the vertical tilt angle from －25^{o} to －55^{o}. Furthermore, we use nine KINECT devices in the octagonal 9-KINECT imaging device to do the background learning, so we apply the processing task described above to the images taken by the nine KINECT devices in the experimental place and get many sets of 3D image constructors of the background. But the one KINECT device, which is at the center of the octagonal 9-KINECT imaging device and senses from top to bottom, is used to do background learning only once. The background learning algorithm using the nine KINECT devices of the octagonal 9-KINECT imaging device is as follows.

Algorithm 6.1: background learning algorithm for the nine KINECT devices of the octagonal 9-KINECT imaging device

Input: the nine KINECT devices D_0, D_1, …, D_8 of the octagonal 9-KINECT imaging device, where the KINECT device D_0 is at the center of the octagonal 9-KINECT imaging device and senses from top to bottom; the experimental place *EP* without moving objects; an angle value *AG*; a counter with it value *C* set to be 0 initially.

Output: many sets of the 3D image constructors of the background *BG* whose depth images will be used to the background subtraction technique.

Steps:

Step 1. Set the value *AG* of the angle to －25^{o}.

Step 2. If the KINECT device D_C is not the KINECT device D_0, then set the tilt angle of the KINECT device D_C with the angle value *AG*; else, go to the next step.

Step 3. Use the KINECT device D_C to sense the experimental place *EP* for a while, and get a set of depth images, *DI*, of the KINECT images and a set of color images, *CI*, of the KINECT images.

Step 4. Average the multiple depth images in *DI* to get an average depth image *AVGD*.

Step 5. Choose one color image from those color images of the KINECT images as the background color image *BGC*.

Step 6. Use the Kinect-for-Windows SDK with those depth images in *DI* as input to produce a set of mapping arrays, *MA*.

Step 7. Choose one mapping array from *MA* as the final mapping array *FMA* for constructing the 3D image of the background.

Step 8. Regard the average depth image *AVGD*, the background color image *BGC* and the final mapping array *FMA* as a set of the 3D image constructor of the background, CT_{3D}.

Step 9. Put CT_{3D} into the set of the 3D image constructors of the background *BG*.

Step 10. Decrement the angle value *AG* by -2.

Step 11. If the KINECT device D_C is the KINECT device D_0, then go to Step 13; else, go to the next step.

Step 12. If the value *AG* is larger than -55^{o}, then repeat Steps 2 through 10; else, go to the next step.

Step 13. Increment the value *C* of the counter by 1.

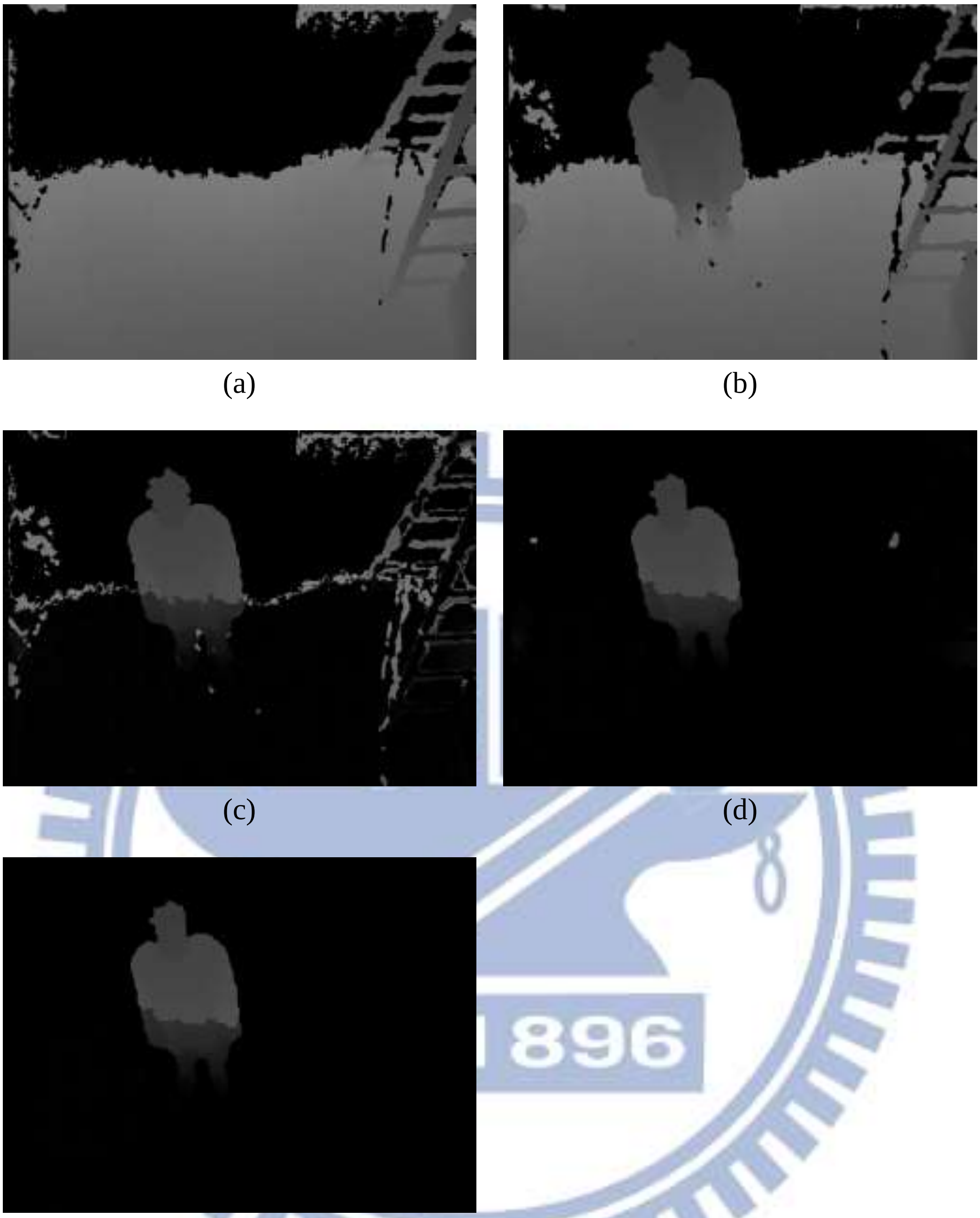
Step 14. If the value *C* is smaller than nine, then repeat Steps 1 through 13; else, exit.

6.2.2 Human Detection by Depth Image

With the background learning done, we start to conduct human detection. As we mentioned in Chapter 1, we make two assumptions as follows.

1. The indoor environment is unchangeable all the time.
2. The detected motion objects are humans for security surveillance.

We will follow these two assumptions to design the background subtraction technique. When we use a KINECT device to sense the indoor environment with human activities and get a pair of KINECT images, we subtract the depth image in this pair from the background depth image acquired from the results of the background learning and get a *subtracted depth image*. In the subtracted depth image, there are many fragments and the human shape. The fragments are caused by the fact that the reflective infrared light rays are interfered in some situations as described in Section 6.2.1 to cause fluctuations in the depth image. We will regard the fragments as a kind of noise. We so apply the erosion operator of mathematical morphology to the subtracted depth image to reduce the small fragments. Then, we apply the dilation operator of mathematical morphology to the resulting depth image to restore the lost parts of the human shape and big fragments which are shrunken by the erosion operator. Finally, we apply the region growing scheme with a suitable threshold to the resulting depth image to find the human shape. An example of the results of human detection is shown in Figure 6.4.



(a)

(b)

(c)

(d)

(e)

Figure 6.4 An example of human detection results. (a) The background depth image. (b) The depth image with human activities. (c) The subtracted depth image with many fragments and the human shape. (d) The depth image with the human shape and big fragments after doing erosion and dilation. (e) The final human depth image after applying the region growing scheme with a suitable threshold.

6.2.3 Detection Algorithm

With the idea of human detection described in Section 6.2.2, we will propose an algorithm to implement the idea by using the nine KINECT devices of the octagonal 9-KINECT imaging device. The result will be used for human tracking. The detection algorithm is as follows.

Algorithm 6.2: Human detection by the nine KINECT devices.

Input: the nine KINECT devices D_0, D_1, …, D_8 of the octagonal 9-KINECT imaging device; the background depth images *BDI*s from the results of the background learning; the threshold value *T* which will be used in the region growing scheme; the indoor environment *IE*; a counter with its value *C* set to be 0 initially.

Output: The device *RD* which detects human activities.

Steps:

Step 1. Use a KINECT device D_C to sense the indoor environment *IE* and get a depth image *DI*.

Step 2. Subtract the depth image *DI* from the background depth image *BDI* and get a *subtracted* depth image *SDI*.

Step 3. Apply the erosion and dilation operators of mathematical morphology to the subtracted depth image *SDI* to get the temporary depth image *TDI*.

Step 4. Apply the region growing scheme to the temporary depth image *TDI* with the threshold value *T*, and get the *final* depth image *FDI*.

Step 5. If there is a human shape in the final depth image *FDI*, then go to Step 8; else, go to the next step.

Step 6. If the value *C* is smaller than nine, then increment the value *C* of the counter by 1; else, set the value *C* to be 0.

Step 7. Repeat Steps 1 through 6.

Step 8. Record the KINECT device D_C as the result device *RD* and exit.

6.3 Human tracking

6.3.1 Human Tracking with Single KINECT Device

Once the human is detected, we can start to track the human's activities. We can know which KINECT device of the nine KINECT devices of the octagonal 9-KINECT imaging device detects the human from the result of the human detection algorithm in Section 6.2 and we call that KINECT device the *tracking KINECT device*. When we use the tracking KINECT device to track the human's activities, we get a multiple of KINECT images. We can apply the methods described in Section 6.2.2 to the depth images of those KINECT images with the background depth images acquired from the results of the background learning to get the *human's depth images.* Next, we construct the human's 3D data from the human's depth images by the algorithm described in Chapter 4. Then, we analyze the human's 3D data together with the frame rate of the tracking KINECT device to get the moving velocity and direction of the human. With the information above, we can predict the next position the human will go, and the tracking KINECT device can adjust accordingly its tilt angle dynamically to track the human.

6.3.2 Handoff between KINECT Devices

When the human is going out of the field of view of the tracking KINECT device, we should use one of the other KINECT devices of the octagonal 9-KINECT imaging device to keep tracking the human. So there is a *handoff problem* between the nine KINECT devices. Because we have the spatial relations between the nine KINECT devices and the overlap regions of every two neighboring KINECT devices of the nine KINECT devices, it is easier to conduct the task of handoff between the nine KINECT devices. The handoff strategy we adopt is that when the human is going into the overlap region of the tracking KINECT device and its neighboring KINECT device, we let the neighboring KINECT device to assume the role of the new tracking KINECT device to complete the handoff task.

6.3.3 Tracking Algorithm

With the dynamic human tracking technique described in Section 6.3.1 and with single KINECT device and the handoff strategy described in Section 6.3.2, we can integrate them into a tracking algorithm using the nine KINECT devices of the octagonal 9-KINECT imaging device. The algorithm is described as follows.

Algorithm 6.3: human tracking using the nine KINECT devices

Input: the tracking KINECT device *TKD* which is assigned according to the result of the human detection algorithm described in Section 6.2; the neighboring KINECT devices *NKD*s of the tracking KINECT device *TKD*; the overlap regions *OR*s of the tracking KINECT device *TKD* and its neighboring KINECT devices *NKD*s.

Output: the new tracking KINECT device *RKD* for keeping tracking the human activities, which will be set to be “null” if the human is going out of the fields of view of the nine KINECT devices.

Steps:

Step 1. Use the tracking KINECT device *TKD* to track the human and get some KINECT images *KI*s.

Step 2. Apply the methods described in Section 6.2.2 to the depth images of the

KINECT images *KI*s to get the human's depth images *HDI*s.

Step 3. Construct the human's 3D data H_{3D}s from the human depth images *HDI*s.

Step 4. Analyze the human's 3D data H_{3D}s and get the moving velocity *MV* of the human and the moving direction *MD* of the human.

Step 5. Use the moving velocity *MV* and moving direction *MD* to predict the next position *NP*.

Step 6. If the position *NP* is still in the field of views of the tracking KINECT device *TKD*, then repeat Steps 1 through 5; else, go to the next step.

Step 7. If the position *NP* is in the field of view of the tracking KINECT device *TKD* with different tilt angles, then change its tilt angle by its tilting device and repeat Steps 1 through 6; else, go to the next step.

Step 8. If the position *NP* is in the one of the overlap regions *OR*, then take the involved neighboring KINECT device *NKD*, which shares this overlap region with the tracking KINECT device *TKD*, as the new tracking KINECT device *RKD* and exit; else, go to the next step.

Step 9. If the position *NP* is out of the fields of view of all the nine KINECT devices, then set the new tracking KINECT device *RKD* as null and exit.

6.4 Experimental Results

An example of the human tracking by tilting KINECT devices is shown in this section. The path of the human activities is shown in Figure 6.5. The 3D image sequences of the tracking human activities are displayed in 3D images by the OpenGL and are shown in Figure 6.6.

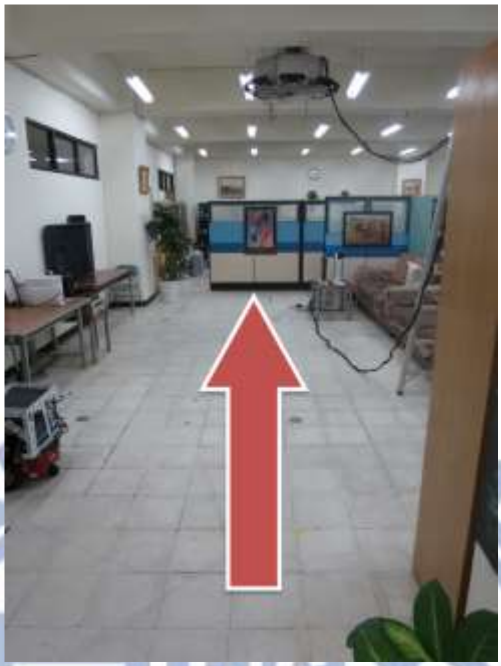
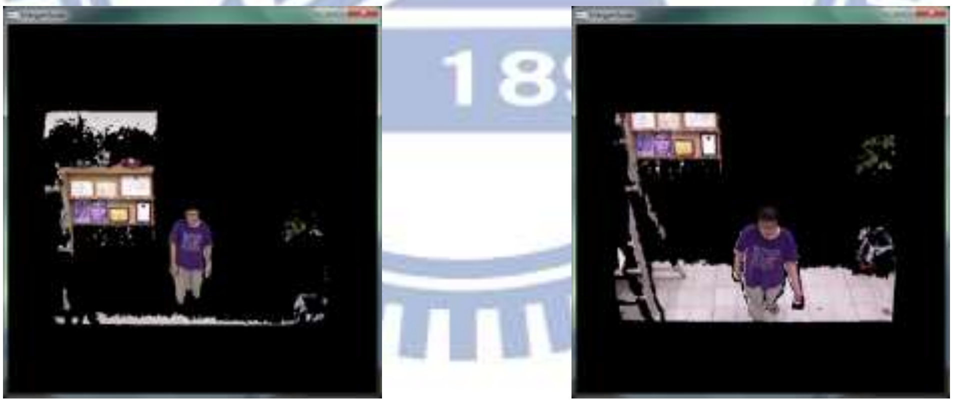
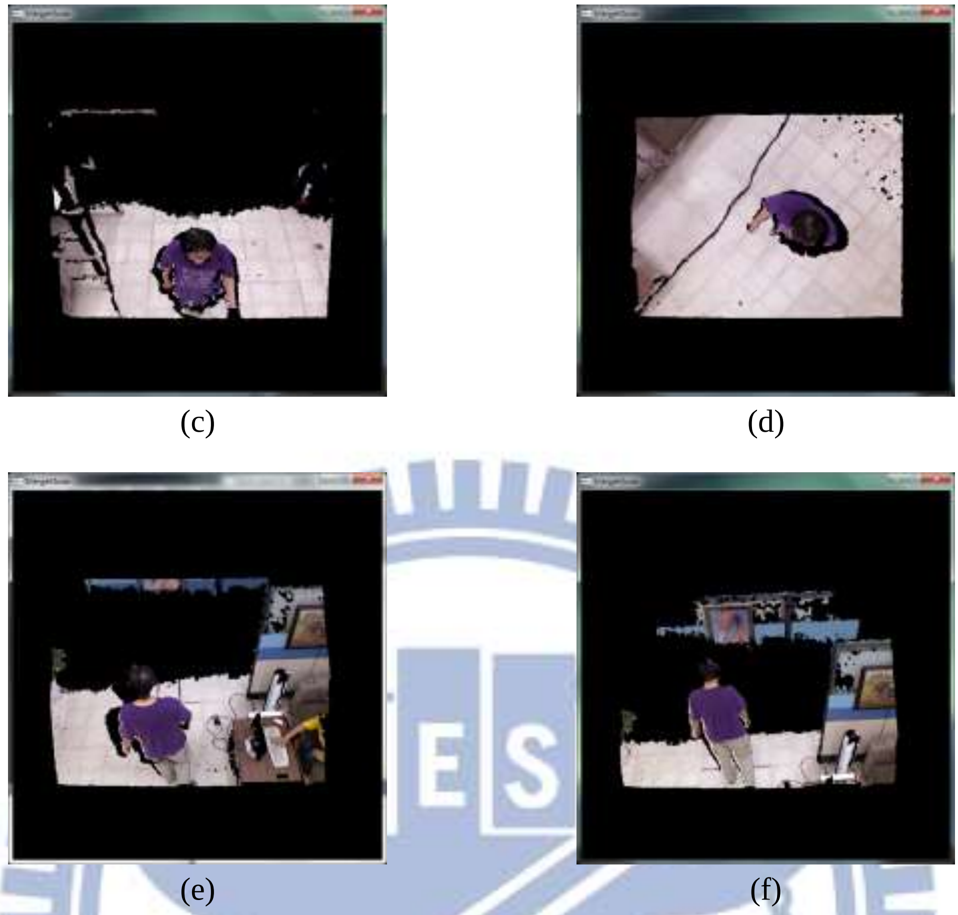


Figure 6.5 The red arrow indicates the path of the human activities.



(a) (b)

Figure 6.6 The 3D image sequences of tracking human activities. In the 3D image sequence from (a) to (f), we applied the tracking algorithm with the nine KINECT devices of the octagonal 9-KINECT imaging device.



(c) (d)

(e) (f)

Figure 6.6 The 3D image sequences of tracking human activities. In the 3D image sequence from (a) to (f), we applied the tracking algorithm with the nine KINECT devices of the octagonal 9-KINECT imaging device (cont’d).

Chapter 7
Human Modeling and Display of Human Activities

7.1 Introduction

When we use the tracking KINECT device, which is one of the nine KINECT devices of the octagonal 9-KINECT imaging device and is assigned from the result of human detection algorithm described in Section 6.2, to track human activities, we get a KINECT image sequence. We store this sequence and the related mapping array sequences, which are acquired by applying the Kinect-for-Windows SDK to the depth images of the KINECT image sequence, as a set of constructor sequences for constructing a 3D image sequence. When we use the tracking KINECT device to track human activities, the handoff problem also occurs between the nine KINECT devices. By using the tracking algorithm described in Section 6.3.3, we can get a new tracking KINECT device to keep tracking human activities. So we store many sets of constructor sequences from different tracking KINECT devices.

In this chapter, we will use the sets of constructor sequences to build up the human model and display accordingly the human activities. First, we will build up several human models by the sets of constructor sequences acquired from different tracking KINECT devices respectively. The modeling method for the each set of constructor sequences will be described in Section 7.2. And then, we merge the resulting human models into one. The merge technique will be described in Section 7.3. The final resulting model will be shown in Section 7.4.

7.2 Human Modeling from Single KINECT Device

7.2.1 Review of Distance-weighted Correlation (DWC)

The measure of distance-weighted correlation (DWC) was proposed by Fan and Tsai [12] originally for automatic Chinese seal identification. The measure is defined as the minimum distance between two groups *S* and *T* of pixels of seal imprint images after the two seal imprint images are overlapped. For each pixel *s* in group *S*, we will find out a pixel *t* in group *T* which has the minimum distance to pixel *s*. If the pixel *t* is in a limited circular region *C* with a pre-selected radius *K*, then a weight $w_p^K = 1/(d_p^2 + 1)$ is defined where d_p is the distance between the pixels *s* and *t*; otherwise, the weight w_p^K is defined to be zero. That is, for each pixel *s*, the weight w_p^K is defined as follows:

$$\begin{aligned} w_p^K &= \frac{1}{d_p^2+1}, && \text{if } \ 0 \le d_p \le K, \\ w_p^K &= 0 && \text{otherwise,} \end{aligned} \tag{7.1}$$

where d_p is the distance of point *s* in *S* to the closest point *t* in *T*. Note that *K* is a threshold used to decide an *effective* distance so that any distance larger than this threshold is discarded. Finally, the DWC for the two groups of pixels, *S* and *T*, is defined as follows:

$$C^K(S,T) = \frac{1}{2} \times \left(\frac{1}{N_S} \sum_{s \in S} w_s^K + \frac{1}{N_T} \sum_{t \in T} w_t^K \right), \tag{7.2}$$

where the coefficient 1/2 is included to treat S and T symmetrically and N_S and N_T are the total numbers of pixels in S and T, respectively. It can be verified that $0 \le C^K \le 1$, and that $C^K = 1$ if and only if $S = T$. The DWC, though defined originally for seal identification, is a general measure for *point-type* object shape matching, and so is utilized in this study.

7.2.2 Review of K-d Tree

The K-d tree is a space-partitioning data structure for organizing points in k-dimensional space. It is a useful data structure for searching nearest points with high dimensions on the tree. The K-d tree is also a kind of binary tree in which every node is a k-dimensional point. We use a group of k-dimensional points to construct the K-d tree. Every non-leaf node on the K-d tree can be considered as a splitting plane, which is perpendicular to the axis of one of the k-dimensions, to divide the spatial domain into two parts. Points to the left of this splitting plane are represented by the left subtree of that node and points to the right are represented by the right subtree. When we want to search the closest point in a group of points, it will take less time to search a K-d tree of the group rather than to search the whole group directly. The time complexity of searching becomes $O(N^{1-1/K})$ instead $O(N)$ where the K is the number of dimensions and is usually greater than one. So when we run algorithms using the DWC, we can convert one of the input groups of points into the K-d tree form. It will reduce the time for the step of searching points of the minimum DWC.

7.2.3 Modeling by Speeded-up DWC Using K-d Tree

We will use sets of constructor sequences described in Section 7.1 to build up several models. For each set of constructor sequences, we convert all depth images in the constructor sequences into the human’s depth images by the method described in

Section 6.2.2. In this way, we can get sets of sequences of 3D human images. We then take one of the sets of the sequences of the 3D human images as a human modeling example. Because the sequence of 3D human images is recorded in accordance with the time sequence, each human's 3D image in the sequence is located at a different position with a small distance from each other. We want to find some transformations which can be used to merge every two consecutive 3D human images in the sequence. And then, we extend these transformations to merge all 3D human images to construct a human model. We will use the DWC measure and the K-d tree structure to assist finding these transformations. An algorithm for finding such transformations between 3D human images by speeding up the DWC computation using the K-d tree is shown as follows.

Algorithm 7.1: Finding transformations between 3D human images.

Input: a sequence of 3D human images I_0, I_1, …, I_N, where N is the total number of sequences of 3D human images; a prepared set of transformations T_is; the threshold value K used in the computation of the DWC measure; an initial maximum value M; a counter with its value C set to be 0 initially; an initial transformation T_0.

Output: a set of transformations RT_js where each RT_j is the transformation which, when applied to 3D human image I_{j+1}, will merge 3D human image I_{j+1} and 3D human image I_j, where $j = 0, 1, …, N – 1$.

Steps:

Step 1. Set the maximum value M to be 0.

Step 2. Apply the K-d tree structure to the 3D human image I_C and get a 3D image $IKDT$ in the K-d tree structure form.

Step 3. Apply a transformation T_i, which is not used yet in the prepared transformation set, to the 3D human image I_{C+1} and get a 3D image IT.

Step 4. Take the two 3D images $IKDT$ and IT and the threshold value K as input data for computing the DWC between them.

Step 5. Start the DWC computation to get a DWC value RV.

Step 6. If the value RV is greater than the maximum value M, then update the maximum value M with the value RV and the desired transformation RT_C with the transformation T_i.

Step 7. If the transformations T_is are not exhausted yet, then repeat Steps 2 through 6; else, go to the next step.

Step 8. Increment the value C of the counter by 1.

Step 9. If the value C is smaller than N, then repeat Steps 1 through 8; else, exit.

7.2.4 Modeling Algorithm

After executing the algorithm for finding transformations described above in Section 7.2.3, we get a set of transformations, by which we can start to merge all 3D human images in the sequence. First, we use the first 3D human image as a pivot. And then, merge the other 3D human images into the first one. Finally, we get a human model. The merging algorithm is as follows.

Algorithm 7.2: merging 3D human images.

Input: the sequence of 3D human images I_0, I_1, …, I_N, where N is the total number of the sequences of 3D human images; the transformations RT_0, RT_1, …, RT_{N-1} obtained from Algorithm 7.1, where RT_j is the transformation which, when applied to 3D human image I_{j+1}, will merge 3D human image I_{j+1} into 3D human image I_j where j = 0, 1, …, N-1; a counter with its value C set to be 0 initially.

Output: the merging result *MR*.

Steps:

Step 1. Use the first 3D human image I_0 as a pivot.

Step 2. Put 3D human image I_0 into the merging result *MR*.

Step 3. Merge the transformations RT_0, RT_1, …, RT_C to get a merged transformation MRT_{C}.

Step 4. Apply the transformation MRT_{C} to 3D human image I_{C+1} to get an integrated 3D human image TI_C.

Step 5. Put the resulting 3D human image TI_C into the merging result *MR*.

Step 6. Increment the value *C* of the counter by 1.

Step 7. If the value *C* is smaller than *N*, then repeat Steps 3 through 6; else, go to the next step.

Step 8. Take the final merging result *MR* as the desired human model.

7.3 Merging Human Models from Multiple KINECT Devices

7.3.1 Calibration of Models from Multiple KINECT Devices

As we mentioned in the previous section, we can get several human models by the algorithms described in Section 7.2 with the multiple sets of sequences of 3D human images as input. Because the data sources of these human models come from different KINECT devices, each of these human models is displayed by the viewpoint of their original KINECT device and this is inconvenient for the merging process. If we want to merge these human models more easily, we should calibrate the spatial relation between these human models. Luckily, we have calibrated the spatial relation between the nine KINECT devices of the octagonal 9-KINECT imaging device in Chapter 5, and get the calibration results. We can use the calibration results directly for the spatial relation between these human models, and convert them into the same view.

7.3.2 Merge of Models by Speeded-up DWC Using K-d Tree

With the human models displayed in the same view, there still existing some distances between the models. So we can use the DWC and the K-d tree structure to assist finding transformations between these human models. However, because each of these human models contains too many 3D data of the 3D image, it will take a long time for the processing work. To solve this problem, we would like to process their data source — the sequences of 3D human image. Instead of using all 3D human images in those sequences, we use the first 3D human images of these sequences as pivot 3D images for the processing. To run the processing in order, we label these pivot images by number. It means that we also label the related human models by numbers. The algorithm for finding the transformations between the human models by speeding up computation of the DWC using the K-d tree is as follows.

Algorithm 7.3: finding transformations between human models.

Input: the pivot images PI_0, PI_1, …, PI_N acquired from human models HM_0, HM_1, …, HM_N where N is the total number of the human models; a prepared set of transformations MT_is; a threshold value K used in computing the DWC; an initial maximum value M; a counter with its value C set to be 0 initially; an initial transformation MT_0

Output: a set of resulting transformations RMT_js where RMT_j is the transformation which will apply to human model HM_{j+1} and let human model HM_{j+1} be merged to human model HM_j and j = 0, 1, …, N-1.

Steps:

Step 1. Set the maximum value M to be 0.

Step 2. Apply the K-d tree structure to the pivot image PI_C to get a 3D image *PIKDT* in the K-d tree structure form.

Step 3. Apply a transformation MT_i, which is not used yet in the prepared set of transformations, to the pivot image $PI_{\mathrm{C}+1}$ to get a 3D image *PIMT*.

Step 4. Take two images *PIKDT* and *PIMT* and the threshold value K as input data for computing the DWC between them.

Step 5. Start the DWC computation to get a DWC value *RV*.

Step 6. If the value *RV* is greater than the maximum value M, then update the maximum value M with the value *RV* and the desired transformation RMT_C with the transformation MT_i.

Step 7. If the transformations MT_is are not exhausted yet, then repeat Steps 2 through 6; else, go to the next step.

Step 8. Increment the value C of the counter by 1.

Step 9. If the value C is smaller than N, then repeat Steps 1 through 8; else, exit.

7.3.3 Merging Algorithm

With the resulting transformations which are acquired from the algorithm described in Section 7.3.2, we can start to merge all human models. Because we label the human models by numbers which are the same as those of the pivot images as mentioned in Section 7.3.2, we will merge the human models sequentially according to these numbers. We use the first human model as a pivot model and the other models will be merged into it one by one. The merging algorithm for human models is as follows.

Algorithm 7.4: merging multiple human models.

Input: the human models HM_0, HM_1, …, HM_N where N is the total number of the human models; the resulting transformations RMT_0, RMT_1, …, $RMT_{N\text{-}1}$ obtained from the results of Algorithm 7.3 where RMT_j is the transformation, which when applied to human model HM_{j+1}, will merge human model HM_{j+1} into human model HM_j, where $j = 0, 1, \ldots, N - 1$; a counter with its value C set to be 0 initially.

Output: a model-merging result *FMR*.

Steps:

Step 1. Use the first human model HM_0 as a pivot model.

Step 2. Put the first human model HM_0 into the merging result *FMR*.

Step 3. Merge the transformation RMT_0, RMT_1, …, RMT_C to get a merged transformation $MRMT_C$.

Step 4. Apply the transformation $MRMT_C$ to the human model HM_{C+1} to get a transformed human model THM_C.

Step 5. Put the resulting human model THM_C into the merging result *FMR*.

Step 6. Increment the value *C* of the counter by 1.

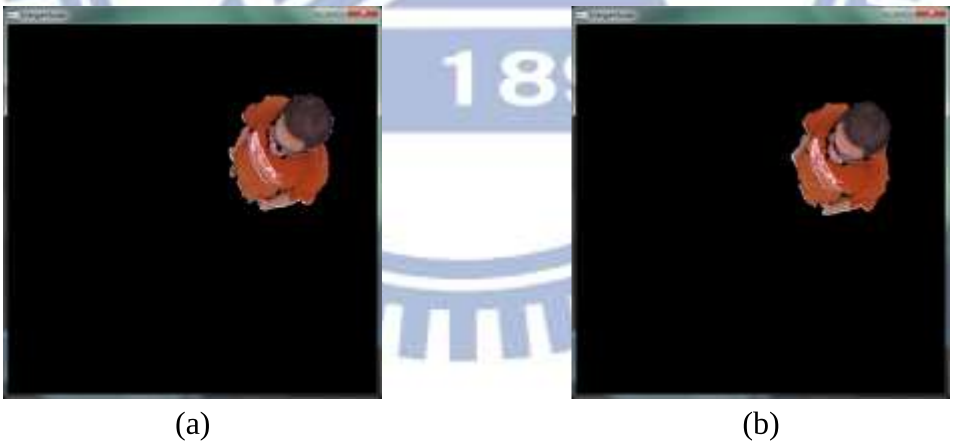
Step 7. If the value *C* is smaller than *N*, then repeat Steps 3 through 6; else, go to the next step.

Step 8. Take the final merging result *MR* as the desired human model.

7.4 Display of Human Activities

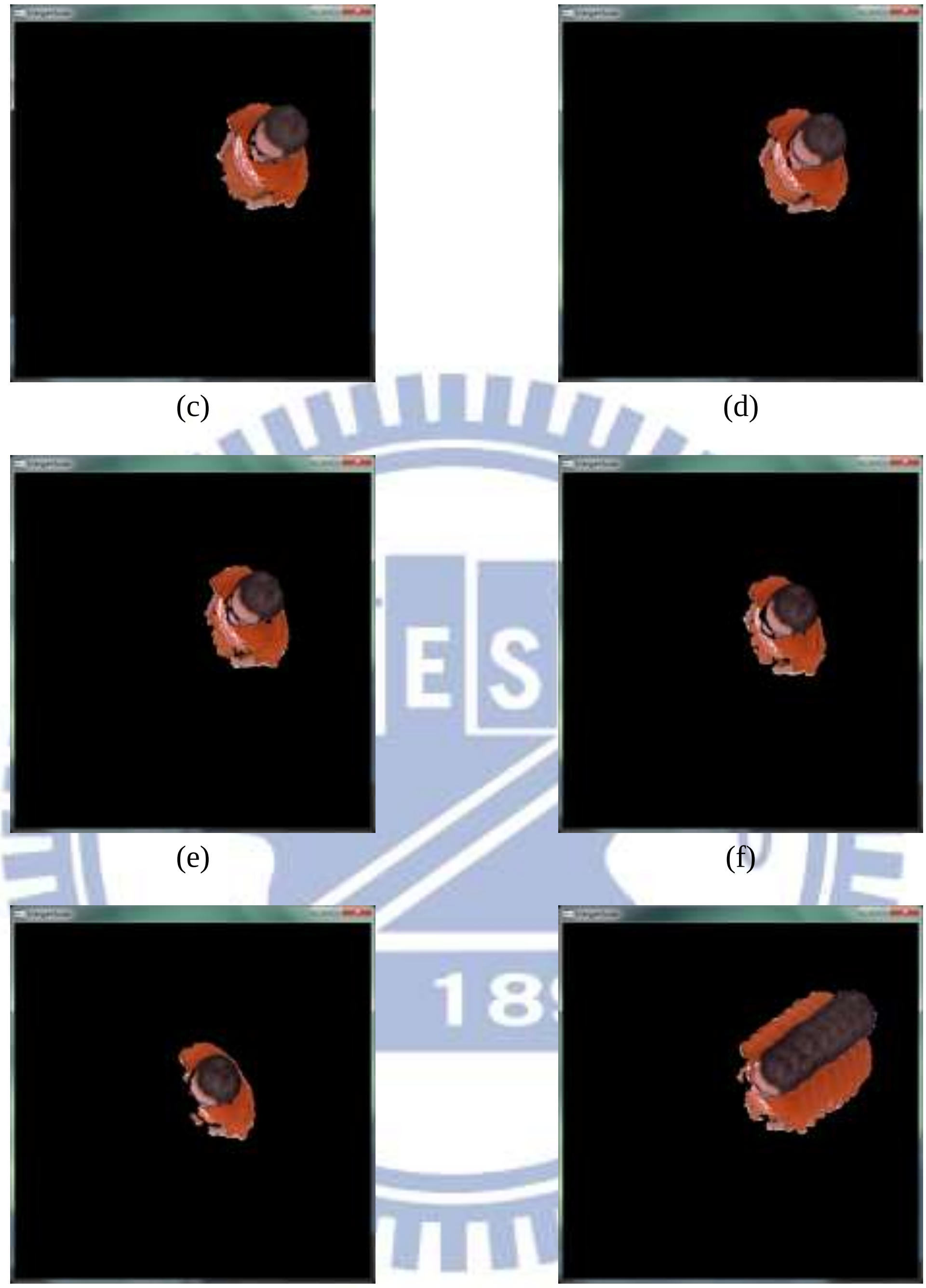
7.4.1 Display of Merged Results

In this section, we will show the whole process for building up the human model. First, we have a sequence of seven 3D human images, which we show by examples in Figure 7.1. Then, we merge the sequence of the seven 3D human images into a human model, which we show by an example in Figure 7.2. Next, we have two human models acquired from two different sequences and we show the two pivot images of the two human models in Figure 7.3. We use the result from calibrating the nine KINECT devices to the two pivot images and show the result in Figure 7.4. Finally, we merge the two pivot images and show the result in Figure 7.5.



(a) (b)

Figure 7.1 A sequence of 3D human images. In (g), the sequence from (a) to (f) is displayed in the meantime.



(c) (d)

(e) (f)

(g) (h)

Figure 7.1 A sequence of 3D human images. In (g), the sequence from (a) to (f) is displayed in the meantime (cont'd).

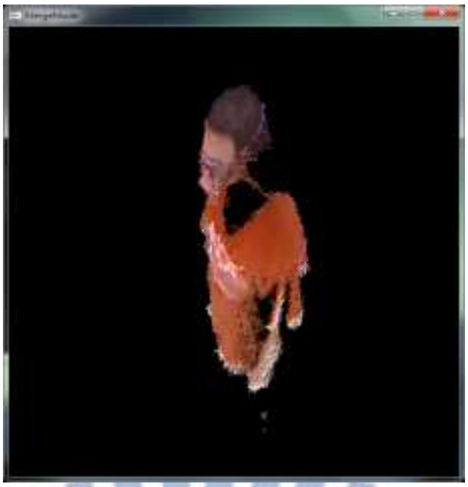
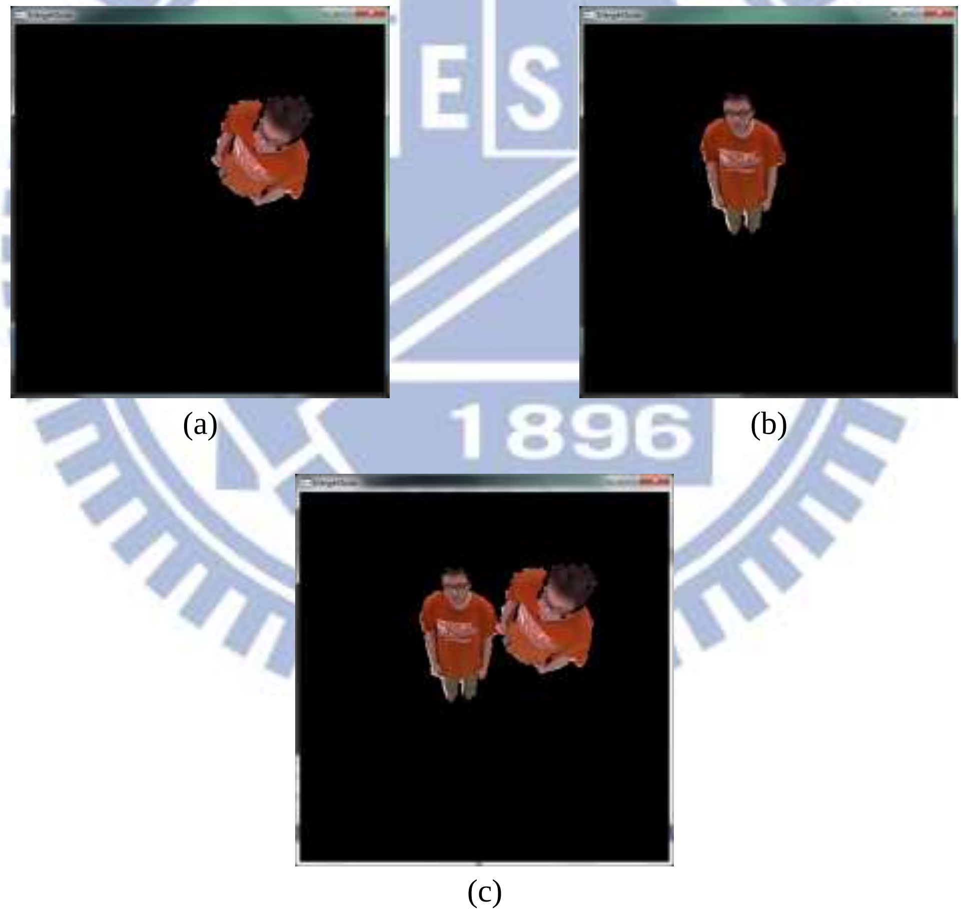


Figure 7.2 A human model constructed from a sequence of 3D human images seen from a side view.



(a)

(b)

(c)

Figure 7.3 Two pivot images of two models. In (c), we display the two pivot images in (a) and (c) at the same time.

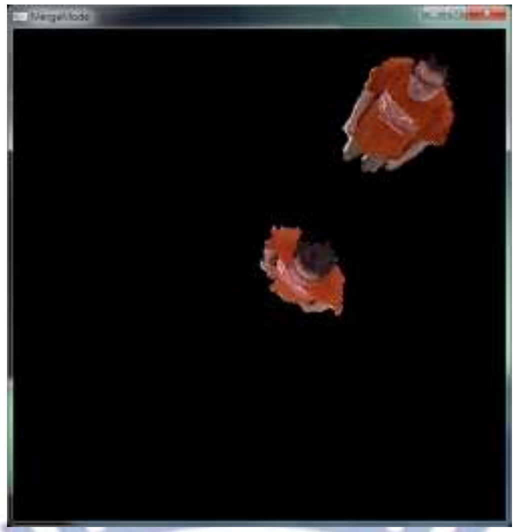


Figure 7.4 Applying the calibration result to the two pivot images.

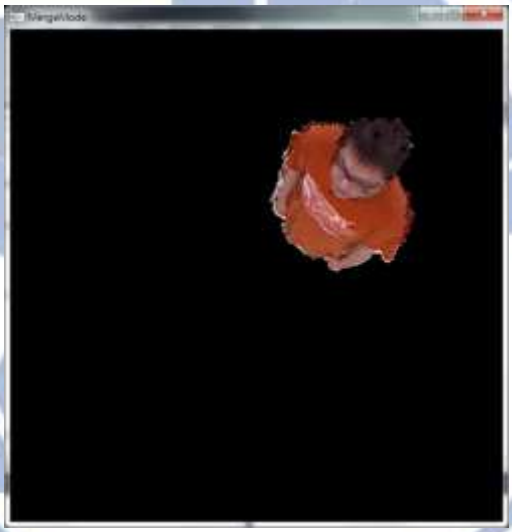
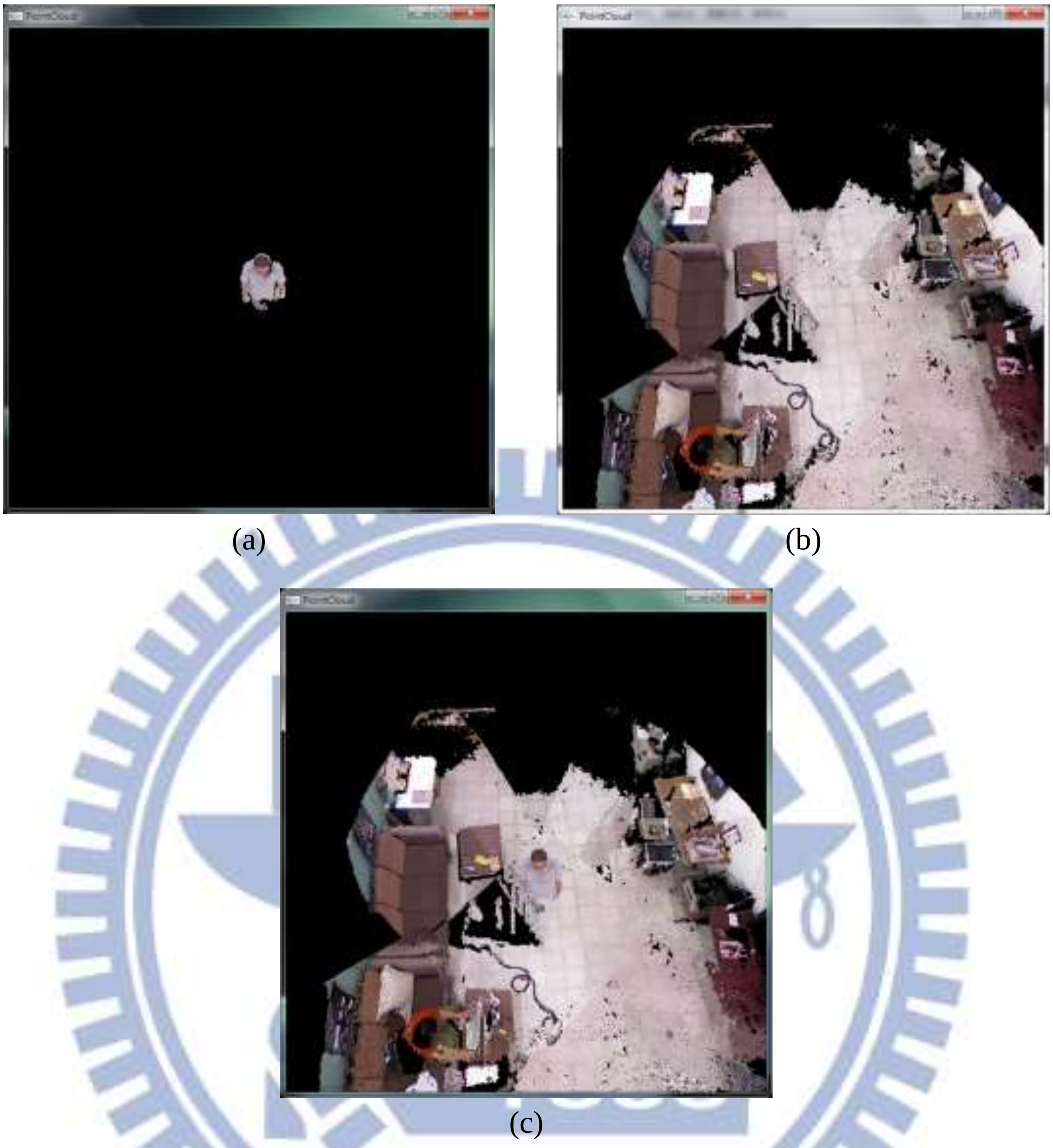


Figure 7.5 Merging result of the two pivot images.

7.4.2 Merge of Human Model and 3D Background

Because we assume that the indoor environment is always static, we can merge the background model and the human model directly. An example of the merging result is shown in Figure 7.6.



(a) (b)

(c)

Figure 7.6 An example of human model and background merging result. (a) The human model. (b) The background model. (c) The merge result.

7.4.3 Extraction of Human Features from Human Model

With the human model constructed, we can analyze the human model. And then, we can get some features of the human such as height, body width, body thickness, etc. Though these features may not be accurate because of the moving actions of the human activities, they are still useful for security monitoring and person identification purposes. An example for extracting human features from the human model is shown in Figure 7.7.

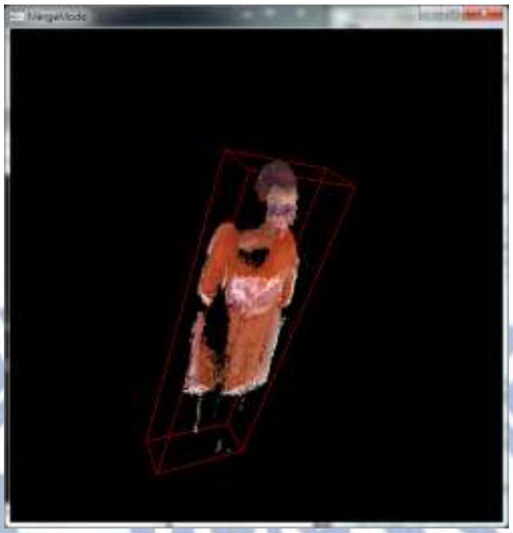


Figure 7.7 An example of human feature extraction from the human model. The red frame can be used to compute the approximate human features like height, body width and body thickness.

Chapter 8
Conclusions and Suggestions for Future Works

8.1 Conclusions

In this study, a system for 3D environment modeling and monitoring via KINECT images for video surveillance has been proposed. To implement such a system, several methods and strategies have been proposed, as summarized in the following.

1. *A conversion method* based on the pinhole camera model has been proposed, which is used to convert KINECT images into 3D images.
2. *A method for geometric correlation* has been proposed, which is used to correct the bending phenomenon existing in the 3D image constructed from KINECT images.
3. *A method for calibration of spatial relations between KINECT devices* based on the concept of the ICP has been proposed, whose results are used to build up indoor environment models.
4. *A method for construction of indoor environment models* has been proposed, which uses the calibration results and 3D images converted from the KINECT images acquired from the octagonal 9-KINECT imaging device to construct indoor environment models.
5. *A strategy for background learning* has been proposed, whose results are used for human detection.
6. *A strategy for human detection based on background subtraction, mathematical*

morphology, and region growing has been proposed, which is applied to the depth image acquired by the octagonal 9-KINECT imaging device and whose result is used for human tracking.

7. *A strategy of human tracking* has been proposed, which utilizes the result of human detection and the tilting device of the KINECT device to conduct dynamic human tracking and solve the handoff problem between the nine KINECT devices of the octagonal 9-KINECT imaging device.
8. *A method for human modeling using the DWC and K-d tree structure* has been proposed, which is a two-step modeling method for constructing human models.

The experimental results shown in the previous chapters have revealed the feasibility of the proposed methods.

8.2 Suggestions for Future Works

The proposed methods and strategies, as mentioned in the last section, have been implemented on the proposed 3D video surveillance system. Based on our experimental experience, several suggestions and related interesting issues worth further investigation in the future are listed as follows.

1. It is desired to extend monitoring regions by using more octagonal 9-KINECT imaging devices.
2. It is desired to increase the total frame rate of the octagonal 9-KINECT imaging device by employing distributed computing systems.
3. It is desired to find a new conversion method for constructing 3D images from KINECT images, which will produce more accurate 3D images.
4. It is worth studying techniques for filling tiny holes in the indoor environment

model constructed from KINECT images.

5. It is desired to find a method to reduce the number of 3D images used in constructing a human model.
6. It is desired to create more accurate human models by applying the mesh structure to the model and rendering the model using textures.

References

[1] M. Zollhöfer, M. Martinek, G. Greiner, M. Stamminger, J. Süßmuth, “ Automatic reconstruction of personalized avatars from 3D face scans, ” *Computer Animation and Virtual Worlds, CASA' 2011 Special Issue*, vol. 22, Issue 2-3, pp. 195–202, April - May 2011.

[2] Shahram Izadi, Richard Newcombe, David Kim, Otmar Hilliges, David Molyneaux, Steve Hodges, Pushmeet Kohli, Jamie Shotton, Anderw Davison and Andrew Fitzbiggon, “KinectFusion: Real-Time Dynamic 3D Surface Reconstruction and Interaction, ” *ACM SIGGRAPH Talks 2011*.

[3] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, “RGB-D Mapping: Using depth cameras for dense 3D modeling of indoor environments,” in *the 12th International Symposium on Experimental Robotics (ISER)*, December 2010.

[4] MIT, U. of Washington and Intel Labs. at Seattle, Visual Odometry For GPS-Denied Flight And Mapping Using A Kinect [Online], 2011, Retrieved from http://groups.csail.mit.edu/rrg/index.php?n=Main.VisualOdometryForGPS-DeniedFlight.

[5] N. Chaiyawatana, B. Uyyanonvara, T. Kondo, P. Dubey and Y. Hatori, “ Robust object detection on video surveillance, ” *Computer Science and Software Engineering (JCSSE), 2011 Eighth International Joint Conference on, Nakhon Pathom*, pp. 149 - 153, May 2011.

[6] Y. L. Tian, M. Lu and A. Hampapur, “Robust and Efficient Foreground Analysis for Real-time Video Surveillance,” *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR’05)*, vol.1, pp.1182-1187, 2005.

[7] L. Xia, C. -C. Chen and J. K. Aggarwal ,“Human Detection Using Depth Information by Kinect,” in *IEEE, International Workshop on Human Activity Understanding from 3D Data in conjunction with CVPR (HAU3D)*, Colorado Springs, pp. 15-22, June 2011.

[8] D. Meltem, G. Kshitiz, and G. Sadiye, “Automated person categorization for video surveillance using soft biometrics,” *Proceedings of the SPIE,* vol. 7667, pp. 76670P-76670P-12, 2010.

[9] J. J. Pantrigo , J. Hernández and A. Sánchez, “Multiple and variable target visual tracking for video-surveillance applications, ” *Pattern Recognition Letters,* vol.31, no. 12, pp. 1577-1590, 2010.

[10] Wikipedia, “Pinhole camera model, ”, March 2013.

http://en.wikipedia.org/wiki/Pinhole_camera_model.

[11] Wikipedia, “Iterative closest point, ”, May 2013.

http://en.wikipedia.org/wiki/Iterative_closest_point.

[12] T. C. Fan and W. H. Tsai (1984). "Automatic Chinese seal identification," *Computer Vision, Graphics, and Image Processing,* Vol. 25, pp. 311-330.