國立交通大學

多媒體工程研究所

碩士論文

利用雙鏡面環場影像攝影和超音波感測技術作戶外自動車學習與導航之研究

A Study on Learning and Guidance for Outdoor Autonomous Vehicle Navigation by Two-mirror Omni-directional Imaging and Ultrasonic Sensing Techniques

研 究 生：黃政揆

指導教授：蔡文祥　教授

中華民國九十九年六月

利用雙鏡面環場影像攝影和超音波感測技術作戶外自動車學習與導航之研究

A Study on Learning and Guidance for Outdoor Autonomous Vehicle Navigation by Two-mirror Omni-directional Imaging and Ultrasonic Sensing Techniques

研究生：黃政揆　　Student：Jheng-Kuei Huang

指導教授：蔡文祥　　Advisor：Wen-Hsiang Tsai

國立交通大學
多媒體工程研究所
碩士論文

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

Computer Science

June 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年六月

利用雙鏡面環場影像攝影和超音波感測技術作戶外自動車學習與導航之研究

研究生：黃政揆　　　指導教授：蔡文祥 博士

國立交通大學多媒體工程研究所

摘要

隨著電腦視覺技術之發展，立體式攝影機逐漸受到歡迎。本研究使用一種新型的立體攝影機與新的引導技術來建製一台機械導盲犬，用以帶領使用者在人行道環境中行走。

在本論文中，我們提出一個新型立體式攝影機的設計方法與公式，讓使用者可以輕易地設計一支立體式攝影機。接著提出一種基於空間對映法的攝影機校正方法來校正此立體式攝影機。基於同軸旋轉不變性質，我們提出適用於此新型立體攝影機的立體資訊之計算方法；不同於其他計算方法，本系統無須將環場影像轉換為全景影像即可計算影像對映點與立體資訊。此外自動車在行走中累積的機械誤差會影響導航時的計算，對此我們亦提出一個誤差校正模型以解決問題。接著我們發展一動態調整相機感光值的方法與一動態調整參考值的方法，來適應環境中不均勻亮度的問題。

我們在本系統中以人行道之路延石作為行走時的特徵點，提出了兩種擷取特徵點的方法。當自動車進行學習時，系統計算特徵點的立體值，找出行走時的方向與距離並據以自動行走，同時自動紀錄與分析路徑點，以建立環境地圖。此外我們亦提出一種人機互動的技術，允許使用者在任何時候都可以手勢控制自動車，此時系統將關閉計算特徵點的程序，並進行盲走之程序。

當自動車進行導航模式時，我們亦提出一種分析串列超音波訊號的方法，使自動車能配合使用者的速度，調整自身行走速度並帶領使用者於環境中行走。接

著我們提出一種改良的閃避障礙物之方法與自動車座標計算之方法，讓自動車能在環境中判斷障礙物的高度，並進行閃避。最後我們提出相關的實驗結果證明本系統的完整性與可行性。

A Study on Learning and Guidance for Outdoor Autonomous Vehicle Navigation by Two-mirror Omni-directional Imaging and Ultrasonic Sensing Techniques

Student: Jheng-Kuei Huang Advisor: Wen-Hsiang Tsai

Institute of Multimedia Engineering, College of Computer Science
National Chiao Tung University

ABSTRACT

With the progress of development in computer vision technologies, 3D stereo cameras nowadays become more popular than in the past. In this study, a new imaging device and new guidance techniques are proposed to construct an autonomous vehicle for use as a robot guide dog navigating on sidewalks to guide the blind people.

A general formula for designing a new stereo camera consisting of two mirrors and a single conventional projective camera is proposed. People can use the formula to design other stereo cameras easily. Then, a calibration technique based on a so-called pano-mapping technique for this type of camera is proposed. Using an autonomous vehicle to navigate in the environment, the incrementally increasing mechanical error is a big problem in the experiment. A calibration model based on the curve fitting technique is proposed to correct such errors. Also, a 3D data acquisition technique using the proposed two-mirror omni-camera based on the rotational invariance property of the omni-image is proposed. The 3D data can be obtained directly without transforming taken omni-images into panoramic images.

The autonomous vehicle is designed to follow the curb line of the sidewalk using the line following technique. In the path learning procedure, two methods are proposed to extract the curbstone feature points. If there exits no curbstone features or the features are hard to extract, a new human interaction technique using hand pose position detection and encoding is proposed to issue a user's guidance command to the vehicle. To adapt the adopted image processing operations to the varying light intensity condition in the outdoor environment, two techniques, called dynamic exposure adjustment and dynamic threshold adjustment are proposed. To create a path map, a path planning technique is proposed, which reduces the number of the resulting path nodes in order to save time in path corrections during navigation sessions.

In the navigation procedure after learning, there may exits unexpected obstacles blocking the navigation path. A technique using a concept of virtual node is proposed to design a new path to avoid the obstacle. Finally, to allow the vehicle to guide a blind person to walk smoothly on a sidewalk, a sonar signal processing scheme is proposed for synchronization between the speed of the vehicle and that of the person, which is based on computation of the location of the vehicle with respect to the person using the sonar signals.

A series of experiments were conducted on a sidewalk in the campus of National Chiao Tung University. And the experimental results show the flexibility and feasibility of the proposed methods for the robot guide dog application in the outdoor environment.

ACKNOWLEDGEMENTS

I am really appreciative of the kind guidance, discussions and support from my advisor, Dr. Wen-Hsiang Tsai. Something Dr. Tsai gave me is not only aspects of development of this thesis, but also attitude of relationship between people and encouragement of facing to problems.

Thanks also given to all classmates in Computer Vision Laboratory in the institute of Computer Science and Engineering at National Chiao Tung University. They have their own personalities which I have to learn with them- careful derivative thinking of Mr. Bo-Jhih You and Mr. Guo-Fong Yang, great programming skills of Miss. I-Jen Lai and Miss. Mei-Hua Ho, positive attitude of Miss. Pei-Hsuan Yuan and Mr. Chih-Hsien Yao, and friendly of Che-Wei Lee who shares his experience with me and inspires me when I lost my emotions in the institute interval.

Finally, I extend my profound thanks to my family for their love, care and support. I cannot acommplish this dissertation without them.

CONTENT

LIST OF FIGURES

LIST OF TABLES

Chapter 1
Introduction

1.1 Motivation

According to the report of Taiwan Guide Dog Association in 2010, if all blind people in Taiwan want to have their own guide dogs, at least the following problems will arise in our society:

1. there are fifty thousand blind people, but only twenty-eight guide dogs in Taiwan till now;
2. it costs at least one million NT dollars to train a guide dog;
3. besides problems about quantity and funds, it is unavoidable to spend much time to get guide dogs acquainted with blind people;
4. problems about personality between guide dogs and blind people also have to be solved.

We believe that technology is used to solve problems that people encounter, so in this thesis study it is desired to solve the problem of guide dog shortage by constructing a *robot guide dog* with new visual sensing equipments and new autonomous vehicle guidance techniques. Specifically, we try to design a new type of camera, called *two-mirror omni-directional camera,* (or just *two-mirror omni-camera*), and use it to capture stereo images of the environment. Also, we desire to use an autonomous vehicle to implement the robot guide dog and develop new methods to carry out the autonomous vehicle guidance work. To achieve these goals in this study, at least we have to solve the following problems:

1. how to design a new type of two-mirror omni-camera?
2. how to train the autonomous vehicle to learn specific environments?

3. how to design intelligent methods for the vehicle to perform specified actions after learning?

We will describe the methods we propose in this study for solving these problems in the subsequent chapters. In the following sections, a survey of related works is given in Section 1.2. An overview of the proposed system is described in Section 1.3. And some contributions of this study are reported in Section 1.4.

1.2 Survey of Related Works

In this section, we categorize existing research results related to this study as follows. A survey of the types of guide dog robot is given in Section 1.2.1. A survey of the types of omni-camera used as the eye of the dog is given in Section 1.2.2. A survey of the types of stereo omni-camera for use in image taking and 3D information inference is given in Section 1.2.3. Finally, a survey of different learning and guidance methods for autonomous vehicles (used as robots) is given in Section 1.2.4.

1.2.1 Types of guide dog robot

According to the problems mentioned previously, developing a guide dog robot is a popular research topic recently. We may categorize the research results of this topic into four types roughly as follows.

1. Use of electronic travel aids:

Some researchers tried to use electronic sensors to help the blind people to avoid obstacles. Borenstein and Ulrich [1] developed a device using sonar sensors to detect obstacles. The user pushes the device ahead of him/ herself to perform this function. When the device detects any obstacle, it uses sound or vibration to let the user know the existence of obstacles around.

2. Use of navigable belts:

In this type, researchers developed navigable devices equipped on the user's body to detect obstacles. Sun and Su [2] developed a vision-based aid system which uses two webcams worn on the head of the user to detect obstacles and sends the information back to the user by sound.

3. **Use of mobile robots as guide dogs:**

Based on powerful computability of modern computers, some researchers developed vision-based mobile robots to detect obstacles, do path planning, localize current spots, and navigate in various environments. Some of such developments in the industry were found in [3][4].

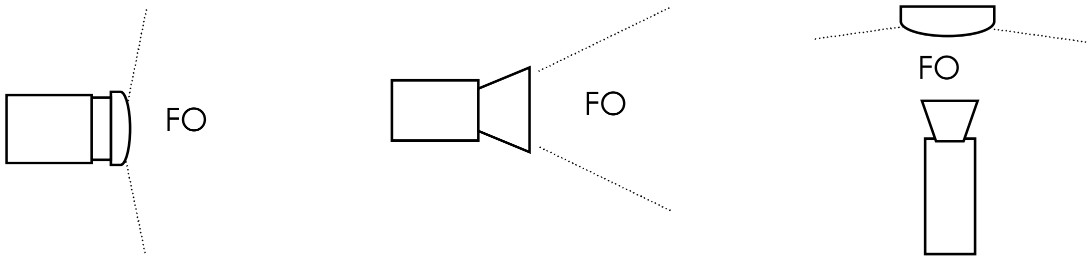
4. **Use of white canes:**

The white cane is a device that most of the blind people use to assist walking. Guo [5] developed a digital white cane which used five sonar sensors and an embedded system to detect obstacles and water on the road.

1.2.2 Types of omni-camera

Contrary to conventional cameras, omni-cameras have larger fields of view (FOV's), and may be used to take images with less blind areas and more visual information. Omni-cameras can be categorized into two types according to their optics architecture, namely, *dioptric* and *catadioptric*. The structures of these two types of camera are shown in Figure 1.1. A well-known example of dioptric omni-cameras is the fish-eye camera [6]. It captures incoming light directly which goes to the camera sensors through the lens.

Contrary to the dioptric camera, a catadioptric camera captures incoming light from a reflective mirror. The mirror of the catadioptric camera can be made of different shapes, like conic, parabolic, hyperbolic, etc. [8]. If all light rays pass through a common point in the reflective mirror, the catadioptric camera is categorized as of the *single-view-point* (*SVP*) type; otherwise, as of the *non-single-view-point* (*non-SVP*) type [9].



(a) (b) (c)

Figure 1.1 FOV's of different camera types [7]. (a) Dioptric camera. (b) Conventional camera. (c) Catadioptric camera.

1.2.3 Types of stereo omni-camera

Many different combinations of mirrors and cameras have been proposed to construct omni-cameras. According to the literature, Ukida et al. [10] used two vertically-aligned omni-cameras to construct a stereo camera system and proposed a method to calibrate it. The epipolar lines are constrained to vertical lines. Zhu [11] used two horizontally-aligned omni-cameras to construct another type of stereo camera system. While using two or more cameras to construct a stereo camera system, keeping the mutual parallelism or alignment of the optical axes of the multiple omni-cameras is a difficult problem to be solved. Therefore, in the recent years, some other researchers tried to avoid this problem by using a single camera with multiple mirrors to construct a stereo camera system. He et al. [12] used an omni-directional stereo camera of this type to detect objects. The author used mirrors of the same shape to construct the omni-camera. Yi and Ahuja [13] used a convex mirror and a concave lens to construct a stereo omni-camera with a single camera. Jang et al. [14] used two hyperbolic mirrors to construct a stereo camera system in which one mirror surface is faced to the other mirror surface. A common point of all these researches is the trial to design new camera structures to solve the problem of optical-axis parallelism or alignment.

1.2.4 Different learning and guidance methods for autonomous vehicles

About the procedure of learning which should be conducted before autonomous vehicle navigation, many methods have been proposed to control the autonomous vehicle to learn the environment. Chen and Tsai [15] and Wang and Tsai [16] proposed algorithms for controlling the autonomous vehicle manually to record navigation paths in the environment. Tsai and Tsai [17] and Chen and Tsai [18] used the color of clothes as features and used a person following technique to guide the autonomous vehicle to learn the path in the indoor environment. Chen and Tsai [19] used ultrasonic sensors to detect the feet of a person with fuzzy control techniques to guide the autonomous vehicle to learn the path in the indoor environment. Wang and Tsai [20] proposed a method using an elliptic skin model to detect human faces. The model defines a region of skin color described by the YC_bC_r color space to describe the features of human faces.

In the procedure of navigation, the mobile robot navigates in environments by path maps which are learned in the procedure of learning. Avoiding dynamic obstacles is a major task in the navigation procedure. Tsai and Tsai [17] proposed a guidance method using sequences of ultrasonic signals in an indoor environment. They analyzed the ultrasonic signals which were detected during navigation and guided people on the learned path. Chiang and Tsai [21] proposed a method to avoid dynamic obstacles. After detecting an obstacle, the robot computes two passing paths to the next path node. And it decides which path to go by comparing the distances from the current position to the next path node through the two passing paths.

1.3 Overview of Proposed System

The goal of this study is to design a vision-based robot guide dog in outdoor environments using an autonomous vehicle. We summarize the work into four major stages in the following and show a flowchart of the system processes in Figure 1.2.

1. Design of proposed two-mirror omni-camera:

At the beginning, a new two-mirror omni-camera is designed. The shapes of the two mirrors are made of a hyperbolic shape. The design of the two-mirror omni-camera is based on an optical property of the hyperbolic shape — light rays, after being reflected by either mirror, pass through one focus point of the hyperbola and then to the other focus point definitely. We equip the camera on the autonomous vehicle. This phenomenon of the light ray reflection satisfies the SVP property mentioned previously, which is less complicated than the non-SVP property.

2. Calibration of proposed two-mirror omni-camera and the autonomous vehicle:

In this stage, we proposed a method to construct space-mapping tables to calibrate the proposed two-mirror omni-camera based on the pano-mapping method proposed by Jeng and Tsai [22]. The detail of the camera design and the calibration procedure will be introduced in Chapter 3 and Chapter 4, respectively. The next step in this stage is to calibrate the autonomous vehicle we use as the guide dog. The problem we have to solve is that the odometer in the vehicle we use to record the position and the turning angle of the vehicle at specified spots is not precise enough. A curve fitting technique is proposed to correct this problem which is also introduced in Chapter 4.

3. Learning procedure:

In the procedure of learning which is necessary before the vehicle can be navigated, we adopt the technique of line following for vehicle guidance. The line to follow by the vehicle is the curb of the road in the outdoor environment. And to detect the along-road curb line, we use the color of the curbstone and describe it by its HIS color features in the omni-image captured by the proposed two-mirror omni-camera. Furthermore, a hand position detection technique is proposed to guide the autonomous vehicle to learning navigation paths. Finally, brightness fluctuation in the outdoor environment is also a problem to be solved in this study. We propose two methods, called *dynamic threshold adjustment* and *dynamic exposure adjustment*, to reduce the influence of the problem on the imaging analysis result for use in vehicle guidance. The detail is introduced in Chapter 5.

4. Navigation procedure:

In Chapter 6, we will introduce the techniques proposed in this study for use in the navigation procedure. We use ultrasonic sensors and design a human-foot detection algorithm to synchronize the speed of the autonomous vehicle with the person guided by the robot dog (the vehicle itself). We also proposed a method adapted from a method, called *goal-directed minimum-path following*, proposed by Chiang and Tsai [21] to avoid dynamic obstacles encountered in the navigation stage.

1.4 Contributions of This Study

Some contributions made by this study are listed in the following.

A new type of stereo camera, called two-mirror omni-camera, is designed in this study.

1. A space-mapping calibration method is proposed for the two-mirror

omni-camera to generate a relation between the image coordinates and the world coordinates.

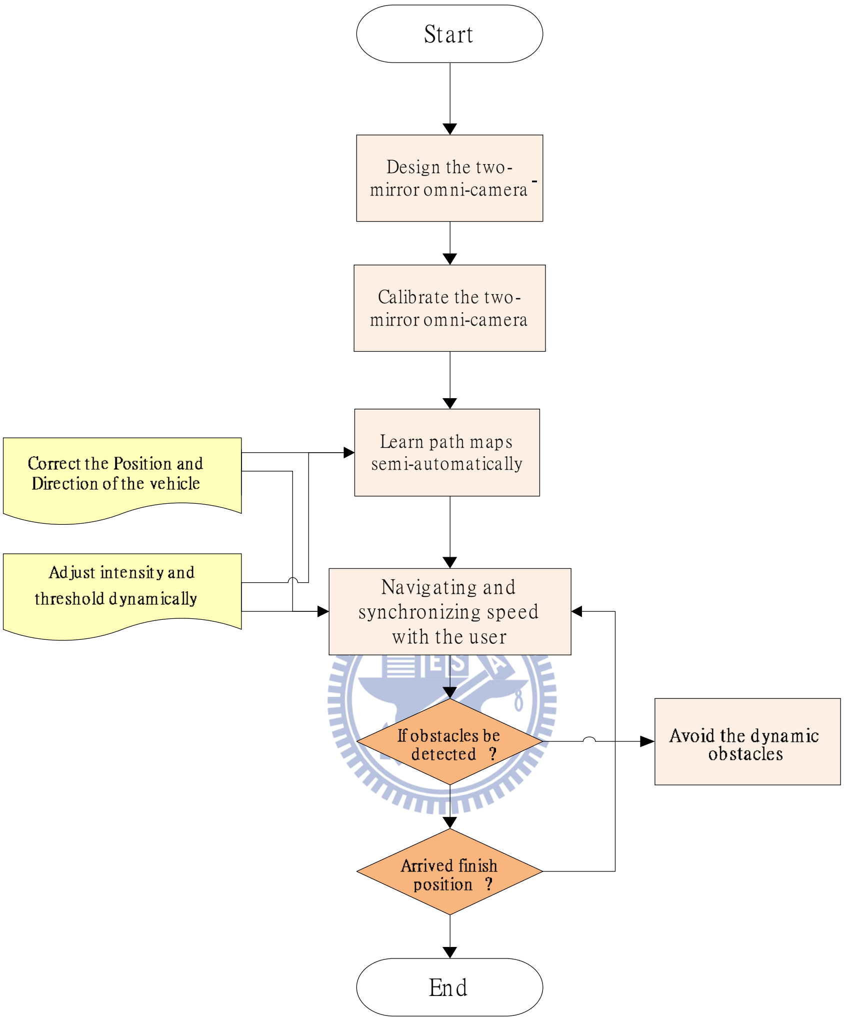

Start
Design the two-mirror omni-camera
Calibrate the two-mirror omni-camera
Learn path maps semi-automatically
Correct the Position and Direction of the vehicle
Adjust intensity and threshold dynamically
Navigating and synchronizing speed with the user
If obstacles be detected ?
Avoid the dynamic obstacles
Arrived finish position ?
End

Figure 1.2 Flowchart of proposed system.

2. A method is proposed to correct accumulated mechanical errors of the vehicle position and direction resulting from long-time autonomous vehicle navigation and inaccurate odometer readings.
3. A method is proposed to guide the autonomous vehicle to learn the navigation

path semi-automatically using a technique of hand position detection proposed in this study.

4. A technique is proposed to adjust image thresholds dynamically in outdoor environments under different intensity for the purpose of object detection in omni-images.
5. Two techniques are proposed to extract the guiding (curb) line of the sidewalk in the environment.
6. A method is proposed to calculate the height and depth of in-road objects for the vehicle to avoid dynamic obstacles.
7. A foot detection method is proposed to synchronize the speed of the autonomous vehicle with the guided person by ultrasonic sensing techniques.

1.5 Thesis Organization

The remainder of this thesis is organized as follows. In Chapter 2, the proposed ideas of the study and the system configuration are introduced. In Chapter 3, the design of the proposed two-mirror omni-camera is described. In Chapter 4, the proposed techniques for calibration of the two-mirror omni-camera and correction of the mechanical error of the autonomous vehicle are described. In Chapter 5, we describe the supervised learning method for generating the navigation path map in the outdoor environment. In Chapter 6, an adaptive method used to avoid dynamic obstacles and a technique used to synchronize the vehicle speed with the user’s are described. In Chapter 7, experimental results and discussions are included. Finally, conclusions and some suggestions for future works are given in Chapter 8.

Chapter 2
Proposed Ideas and System Configuration

2.1 Introduction

In this chapter, we will introduce the major ideas proposed in the thesis study. In Section 2.1.1, we will introduce the main ideas of the proposed learning method for outdoor navigation and use some figures to illustrate our ideas. In Section 2.1.2, we will describe the main ideas of the proposed guidance method. To develop the system of this study, we designed a two-mirror omni-camera and equipped it on an autonomous vehicle. The entire hardware configuration is described in Section 2.2.1. The software configuration and the system architecture are presented in Section 2.2.2.

2.1.1 Idea of proposed learning methods for outdoor navigation

The main purpose in the learning procedure is to create a path map and then instruct the vehicle to navigate according to the map in the outdoor environment. As mentioned in our survey, many methods have been proposed to guide the autonomous vehicle for learning navigation maps. Different from those methods which essentially conduct learning *manually*, we use special features in the environment to guide the autonomous vehicle to learning the path map *semi-automatically*.

In addition, in this study we use the color of the curbstone at the sidewalk, as the example shown in Figure 2.1, in the outdoor environment to guide the autonomous vehicle for the purpose of learning a path map. If the color is similar to the ground color of the sidewalk or if no special color appears on the curbstone in the environment, the autonomous vehicle will not know how it should go in the procedure of learning. Therefore, it is desired to propose another method to handle these conditions. A new approach we use in this study consists of several major steps as follows.

1. Allow the user aside the vehicle to touch the transparent plastic enclosure of the omni-camera at several pre-selected positions on the enclosure which represent different meanings of commands for guiding the vehicle.
2. Detect the hand touch position from the omni-image taken when the hand is put on the camera enclosure.
3. Determine the command from the detected hand position.
4. Instruct the vehicle to conduct navigation according to the command.

In addition, when navigating in outdoor environments, the vehicle often encounters the problem of varying light intensity in the outdoor environment which has a serious influence on the image analysis work. Our general idea to solve this problem is to dynamically adjust the camera exposure value as well as the image threshold value used in the hand detection process just mentioned.

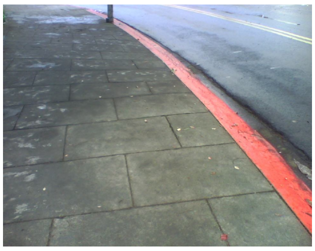


Figure 2.1 The color of curbstone at a sidewalk in National Chiao Tung University.

We use seven major steps to accomplish the learning procedure as described in the following.

1. Decide whether it is necessary to adjust the camera exposure value and the image threshold values for use in image analysis.
2. Use the hue and saturation of the HSI color space to detect sidewalk curbstone features.
3. Detect the edge between the sidewalk region and the curbstone region.
4. Compute the range data of the guide line formed by the curb using the detected edge.
5. Instruct the vehicle to perform actions to follow the guide line.
6. Detect hand positions in the taken image to guide the vehicle if necessary.
7. Execute the procedure of path planning to collect the necessary path nodes and record them in the learning procedure.

2.1.2 Idea of proposed guidance methods for outdoor navigation

In this study, the autonomous vehicle is designed to guide the blind people to walk safely on the sidewalk environment which is learned in the learning procedure. To accomplish this, we may encounter at least two difficulties as follows.

The first is whether the vehicle knows the existence of the user who follows the autonomous vehicle. In several studies, some developers design a pole to let the user hold it and follow the guide robot. But if the user releases the pole, the robot would not know that the user is behind. To solve this problem, we propose a method using ultrasonic sensors to synchronize the speed of the autonomous vehicle with that of the blind.

Another difficulty is that some dynamic obstacles may block the navigation path of the autonomous vehicle. In Chiang and Tsai's method [21], they used a 2D projection method to compute the depth distance and the width distance of the obstacle, and generated the minimum avoidance path from the current position to the next path node. Because they used only one camera to capture the image in the environment, the vehicle cannot know the height distance of the obstacle. If the obstacle is not flat and blocks the navigation path, it could go forward instead of avoiding the obstacle. In our study, we use the proposed two-mirror omni-camera to detect path obstacles and so can get the range data of the obstacle to solve this problem.

In addition, we propose an adaptive collision avoidance technique based on Chiang and Tsai's method [21] to guide the vehicle to avoid dynamic obstacles after detecting the existence of them.

We use seven major steps to accomplish the procedure of navigation as described in the following.

1. Decide whether it is necessary to adjust the camera exposure value and the threshold value for use in image analysis.
2. Compute the distance from the vehicle to the user by the ultrasonic sensors equipped on the vehicle to synchronize their speeds.
3. Detect the guide line.
4. Localize the position of the vehicle with respect to the guide line.
5. Detect the existence of any dynamic obstacle.
6. Plan a collision-avoidance path.
7. Navigate according to the collision-avoidance path to avoid the obstacle if any.

2.2 System configuration

2.2.1 Hardware configuration

All hardware equipments we use in the study are listed in Table 2.1. There are an autonomous vehicle, a laptop computer, two reflective mirrors designed by ourselves, a perspective camera, and a lens. We will introduce them in detail as follows.

Table 2.1 The hardware equipments we use in this study.

Equipments	Product name	Produced corporation
Autonomous vehicle	Pioneer 3	MobileRobots Inc.
Computer	W7J	ASUSTek Inc.
Two reflective mirrors		Micro-Star Int'l Co.
Camera	ARCAM-200SO	ARTRAY co.
Lens	HV3Z615T	Sakai co.

To construct an omni-camera, it is desired to reduce the size of the perspective camera so that its projected area in the omni-image can be as the small as possible (note that the camera appears as a black circle in an omni-image). We use a projective camera of the model *ARCAM-200SO* which is produced by ARTRAY Co. to construct the proposed two-mirror omni-camera, and the used lens is produced by Sakai Co. The picture and the specification of the camera and the lens are listed in Figure 2.2, and Table 2.2 and Table 2.3, respectively The camera size is 33mm × 33mm × 50mm with 2.0 M pixels. The size of the *CMOS* sensor is 1/2 inches (6.4mm × 4.8mm). The specifications of the camera and the lens are important, because we use the parameters in the specifications to design the proposed two-mirror omni-camera, and we will introduce the design procedure in detail in Chapter 3.

Before explaining the structure of the proposed two-mirror omni-camera, we illustrate the imaging system roughly in Figure 2.3. The light rays from a point *P* in the world go into the sensor of the camera both though the reflective mirrors and the center of the lens (the blue line and the red one in the figure).

(a)

(b)

Figure 2.2 (a) The Arcam-200so produced by ARTRAY co. (b) The lens produced by Sakai co.

Table 2.2 The specification of Arcam-200so.

Max resolution	2.0 M pixels
Size	33mm × 33mm × 50mm
CMOS size	1/2" (6.4 × 4.8mm)
Mount	C-mount
Frame per second with max resolution	8 fps

Table 2.3 The specification of lens.

Mount	C-mount
Iris	F1.4
Focal length	6-15mm

We equipped the two-mirror omni-camera on the autonomous vehicle in such a way that the optical axis of the camera is parallel to the ground *originally*. Figure 2.4(a) shows the omni-image taken by the two-mirror omni-camera when it was affixed on the vehicle to be parallel to the ground. The regions in the omni-image drawn by red lines are the overlapping area reflected by the two mirrors. We can see that part of the omni-image reflected by the bigger mirror is covered by that reflected by the small mirror, and the overlapping area is so relative smaller. To compute the range data of objects in the world, each object should be captured by both mirrors. And if this overlapping area is too small, the precision of the obtained range data will become worse.

To solve this problem, it is observed that a front-facing placement of the two-mirror omni-camera with its optical axis parallel to the ground reduces the angle of incoming light rays and so also reduces the overlapping area of the two omni-images taken by the camera. To enlarge this overlapping area, we made a wedge-shaped shelf with an appropriate slant angle and put it under the camera to elevate the angle of the optical axis with respect to the ground, and an omni-image taken with the camera so installed is shown in Figure 2.4(b). We can see that the overlapping area becomes relatively larger now. The proposed two-mirror omni-camera system with its optical axis elevated to a certain angle and affixed on the autonomous vehicle is shown in Figure 2.5.

The autonomous vehicle we use is called *Pioneer 3* which was produced by MobileRobots Inc., a picture of it is shown in Figure 2.6, and its specification is listed in Table 2.4. Pioneer 3 has a 44cm × 38cm × 22cm aluminum body with two 16.5cm wheels and a caster. The maximum speed of the vehicle is 1.6 meters per second on flat floors, the maximum speed of rotation is 300 degrees per second and the maximum degree to climb is 25^o. It can carry payloads up to 23 kg. The autonomous vehicle carries three 12V rechargeable lead-acid batteries and if the batteries are fully charged initially, it can run for 18 to 24 hours continually. The vehicle is also equipped with 16 ultrasonic sensors. A control system embedded in the vehicle provides many functions for developers to control the vehicle.

The laptop computer we use in this study is *ASUS W7J* which was produced by ASUSTeK Computer Inc. We use an *RS-232* to connect the computer and the autonomous vehicle and use a *USB* to connect the computer and the camera. A picture of the notebook is shown in Figure 2.7, and the hardware specification of the laptop is listed in Table 2.5.

2.2.2 System configuration

To develop our system, we use *Borland C++ Builder 6* with updated pack 4 on the operating system Windows XP. The Borland C++ Builder is a GUI-based interface development environment (IDE) software and it is based on the C^{++} programming language. The company, ARTAY, provides a development tool of the camera, called *Capture Module Software Developer Kit*, to assist developers to construct their systems. The SDK is an *object-oriented* interface which is based on the Windows 2000 and Windows XP systems, and written in many computer languages such as C^{++}, C, VB.net, C#.net and Delphi. The company, MobilRobots Inc., provides an *application programming interface (API)*, called ARIA, for developers to control the vehicle. The ARIA is an object-oriented interface which is usable under Linux or Windows system in C^{++} programming language. We use the ARIA to communicate with the vehicle to control the velocity, heading and some navigation settings of it.

Reviewing of the design process of the proposed system, there are four major steps as described in the following.

1. We design a two-mirror omni-camera and equip it on the autonomous vehicle.
2. We propose a calibration method based on Jeng and Tsai's method [22] to calibrate the camera and propose a calibration method using a curve fitting technique to correct the odometer values available from the autonomous vehicle system.

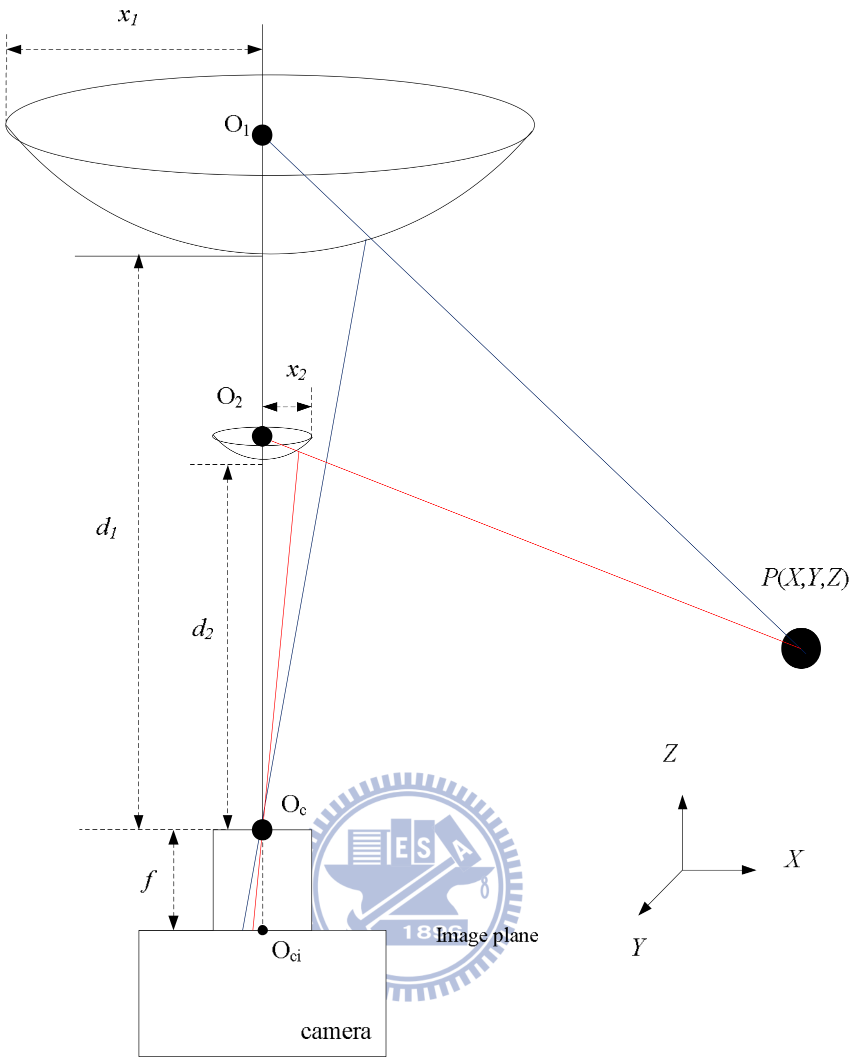
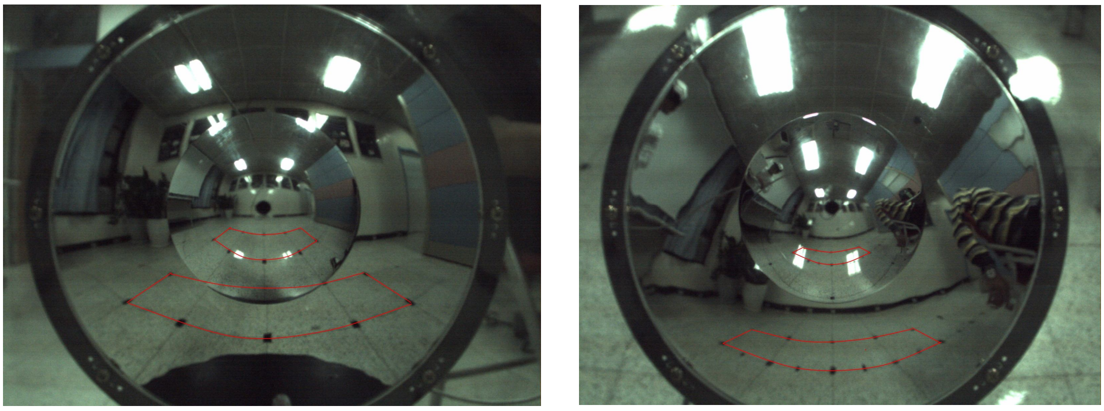
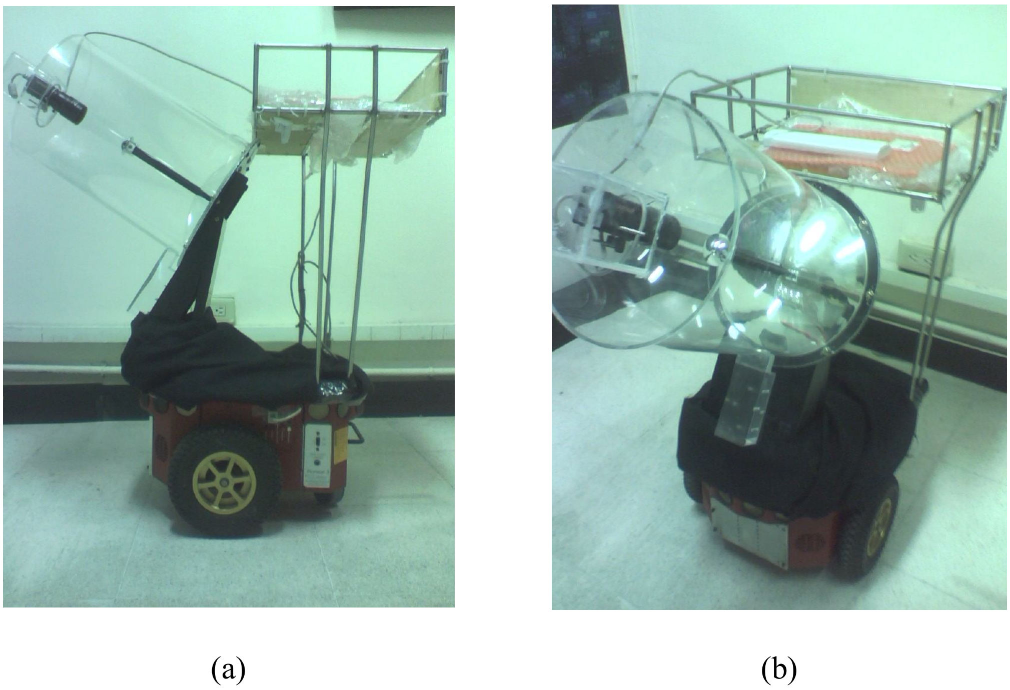


Figure 2.3 A prototype of the proposed two-mirror omni-camera.



(a) (b)

Figure 2.4 The omni-images taken by the two-mirror omni-camera which is placed at different elevations of angles. (a) Image taken when the optical axis of the camera is parallel to the ground. (b) Image taken when the optical axis of the camera is placed at an angle of 45° with respect to the ground.



(a)

(b)

Figure 2.5 The proposed two-mirror omni-camera is equipped on the autonomous vehicle. (a) A side view of the autonomous vehicle. (b) A 45° view of the autonomous vehicle.

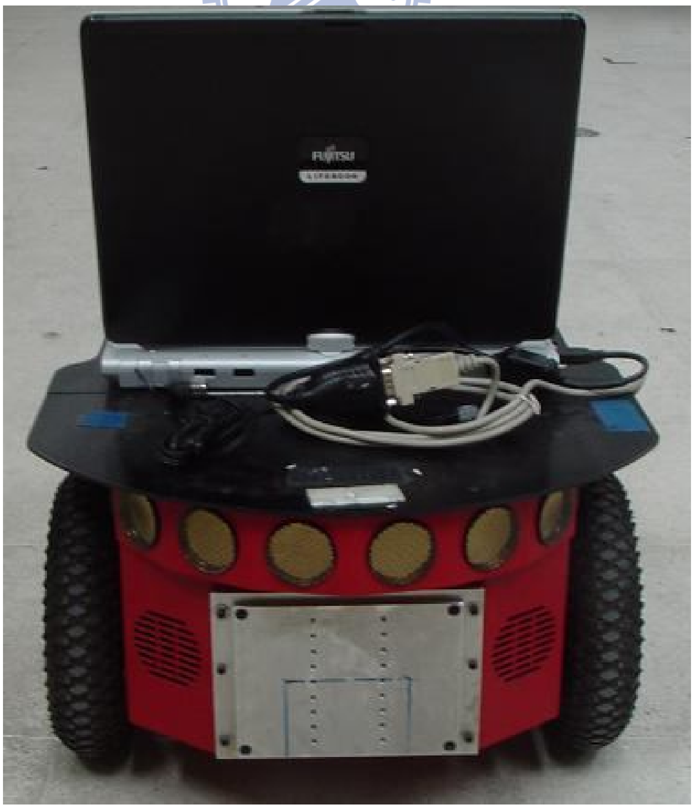


Figure 2.6 Pioneer 3 is the vehicle we use in the study [19].

Table 2.4 Specification of the vehicle Pioneer 3

Size	44cm × 38cm × 22cm
Max speed	1.6 m / sec
Max degree of climbing	25°
Max load	23 kg

Figure 2.7 The notebook ASUS W7J we use in the study.

Table 2.5 Specification of ASUS W7J.

System platform	Intel Centrino Duo
CPU	T2400 Duo-core with 1.84 GHz
RAM size	1.5 GB
GPU	Nvidia Geforce Go 7400
HDD size	80 GB

3. In the learning phase, we record environment information and conduct the path planning procedure to get necessary path nodes. We then detect the color of the curbstone, and use the result as well as a line fitting technique to compute the direction and position of the autonomous vehicle. We also use a method to find the user's hand positions to command the autonomous vehicle.

4. In the navigation phase, the autonomous vehicle can guide the person in the environment that has been learned in the learning phase. We design a method to synchronize the vehicle speed with that of the user. Then, we use the proposed two-mirror omni-camera to compute the range data of obstacles and use a method to avoid dynamic obstacles.

We divide our system architecture into three levels and show our concept in Figure 2.8. Several hardware controlling modules are in the base level. Imaging process techniques, pattern recognition techniques, some methods for computing 3D range data, and some data analysis techniques are in the second level. The highest level shows our main processing modules based on techniques and knowledge in the second level.

The road detection module is used to detect the guide line (the curb) along the sidewalk to guide the autonomous vehicle. The hand position detection module is used to detect the user's hand positions to guide the autonomous vehicle when necessary in the learning procedure. The speed synchronization module adjusts the vehicle speed dynamically to synchronize with the human speed. The position processing module is used to record the learned paths in the learning phase and to calculate the vehicle position in the navigation phase. And the obstacle avoidance module handles the avoidance process to avoid collisions with obstacles after the autonomous vehicle detects the existence of dynamic obstacles in the navigation procedure. We will show more complete ideas and describe these techniques in more detail in the following chapters.

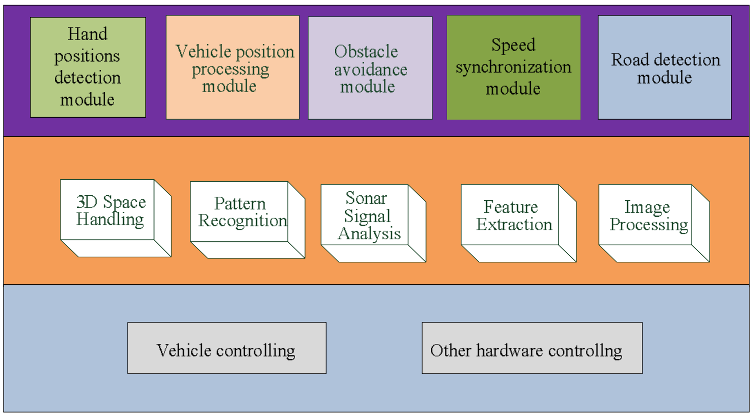

Hand positions detection module
Vehicle position processing module
Obstacle avoidance module
Speed synchronization module
Road detection module
3D Space Handling
Pattern Recognition
Sonar Signal Analysis
Feature Extraction
Image Processing
Vehicle controlling
Other hardware controllng

Figure 2.8 Architecture of proposed system.

Chapter 3
Design of a New Type of Two-mirror Omni-camera

3.1 Review of Conventional Omni-cameras

A conventional omni-camera is composed of a reflective mirror and a perspective camera. An example is shown in Figure 3.1(a), and an omni-image captured by the omni-camera is shown in Figure 3.1(b). We can see that the omni-image has larger FOV's with the aid of a reflective mirror, compared with those taken with the traditional camera. The shape of the mirror may be of various types such as hyperbolic, ellipsoidal, parabolic, circular, etc. An illustration is shown in Figure 3.2. More about the design principle of the two-mirror omni-camera we use in this study will be described in the next section.

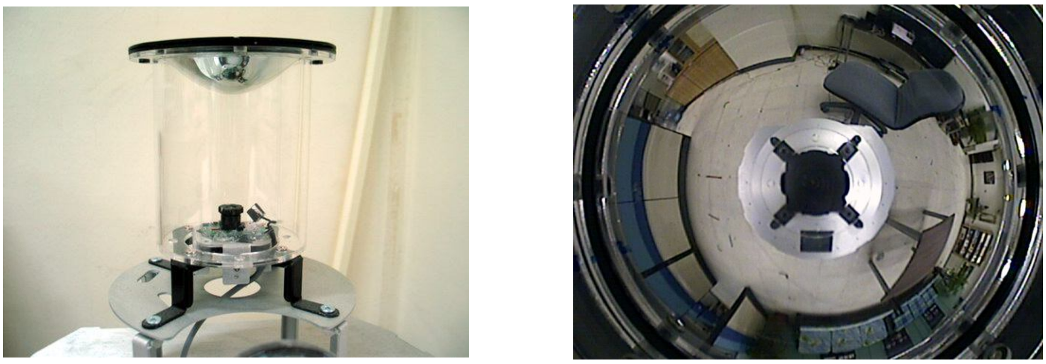


Figure 3.1 A conventional catadioptric camera. (a) Structure of camera. (b) The acquired omni-image [7].

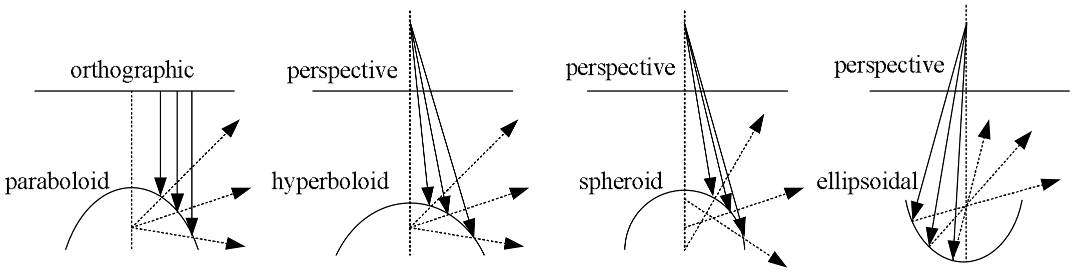


Figure 3.2 The possible shapes of reflective mirrors are used in omni-cameras [7].

3.1.1 Derivation of Equation of Projection on Omni-image

The two-mirror omni-camera used in the study is constructed by two reflective mirrors made of the hyperbolic shape and a camera with a perspective lens. Here, we use an omni-camera with a mirror made of the hyperbolic shape to show the derivation of the equation of image projection first. An important optical property of the hyperbolic-shape mirror is illustrated in Figure 3.3, which says that when a light ray goes through one focus point F_1 of the hyperbolic shape at a space point P which is on the hyperbolic curve, the ray will be reflected to the other focus point F_2 definitely.

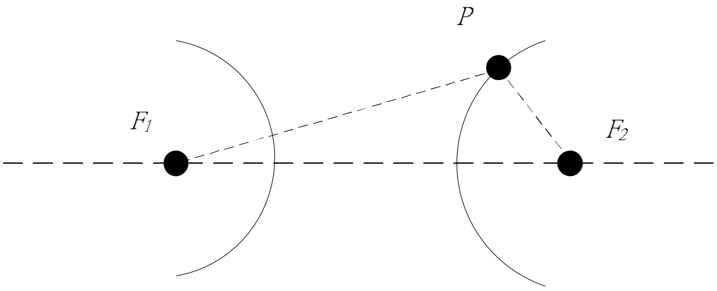


Figure 3.3 An optical property of the hyperbolic shape.

We use this property of the hyperbolic shape to construct the omni-camera we need in this study. The image projection phenomenon of the omni-camera is shown in Figure 3.4 and two coordinate systems, namely, the *camera coordinate system (CCS)* and the *image coordinate system (ICS)*, are used to illustrate the principle of the imaging process. Based on the property mentioned previously, the light ray of a point $P(X, Y, Z)$ in the CCS first goes to the center of the mirror $\mathrm{O_m}$, then is reflected by the mirror to the center of the lens $\mathrm{O_c}$ by the hyperbolic mirror surface, and finally is projected on the image plane to form the image point $q(u, v)$ in the ICS. To satisfy this property of the hyperbolic shape, the optical axis of the perspective camera and the mirror are assumed to be aligned such that the omni-camera becomes to be of the SVP type.

The point O is taken to be the origin in the CCS which is also the middle point between the two focus points $\mathrm{O_m}$ and $\mathrm{O_c}$ of the hyperbola, where $\mathrm{O_m}$, which is also the center of the base of the hyperbolic-shaped mirror, is located at $(0, 0, -c)$; and $\mathrm{O_c}$ is located at $(0, 0, +c)$ in the CCS. The hyperbolic shape of the mirror of the omni-camera can be described by the following equation:

$$\frac{R^2}{a^2} - \frac{Z^2}{b^2} = -1, \qquad R = \sqrt{X^2 + Y^2} \tag{3.1}$$

where a and b are two parameters of the hyperbolic shape, and

$$c = \sqrt{a^2 + b^2} \tag{3.2}$$

is the distance to the focus point $\mathrm{O_m}$. The projection relationship between the CCS and the ICS can be described [7] as:

$$u = \frac{Xf(b^2 - c^2)}{(b^2 + c^2)(\mathrm{Z} - c) - 2bc\sqrt{(\mathrm{Z} - c)^2 + X^2 + Y^2}},$$

$$v = \frac{Yf(b^2 - c^2)}{(b^2 + c^2)(\mathrm{Z} - c) - 2bc\sqrt{(\mathrm{Z} - c)^2 + X^2 + Y^2}}, \tag{3.3}$$

where *f* is the focal length of the camera.

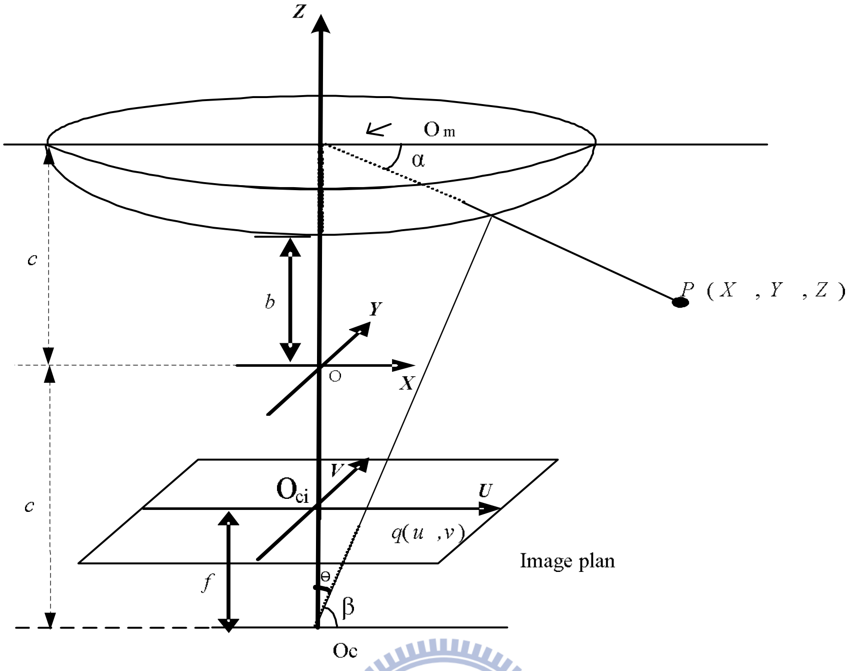


Figure 3.4 Relation between the camera coordinates and the image coordinates.

3.2 Proposed Design of a Two-mirror Omni-camera

3.2.1 Idea of design

The conventional non-stereo camera can only take images of single geometric planes, and cannot be used to get 3D range data, including the depth, width, and height information of scene points, compared with the stereo camera. While using two or more cameras to construct a stereo camera system, keeping the mutual parallelism or alignment of the optical axes of the cameras is a difficult problem. We desire to solve this problem by designing a new stereo camera, therefore a two-mirror omni-camera is proposed in this study. In addition to resolving this problem, the proposed omni-camera also has larger FOV’s than the conventional camera.

3.2.2 Details of design

In this section, we introduce the design procedure of the proposed two-mirror omni-camera in detail. An illustration of its structure is shown in Figure 2.2. The upper mirror is bigger, named *Mirror 1*, and the lower is smaller, named *Mirror 2*. They are both made to be of the hyperbolic shape. To construct the desired two-mirror omni-camera, we took two major steps as described in the following.

Step 1: Decision of the size and the position of the mirrors.

To construct the desired two-mirror omni-camera, the first step is to decide the light injection and reflection positions on the two mirrors in the imaging process. An assumption is made in advance, that is, the radius which is calculated as the distance from "the spot where the light rays are reflected from the Mirror 2" to O_{ci} is half of the radius which is calculated as the distance from "the spot where the light rays are reflected from the Mirror 1" to O_{ci}.

Based on the assumption mentioned above, a simulated omni-image is shown in Figure 3.5, where the red part is the region which is reflected by Mirror 2, and the blue part is the region which is reflected by Mirror 1. The radius of the blue region is two times to the radius of the red region.

Before deriving the formulas, we define some terms or notations appearing in Figure 3.6 which is simplified from Figure 2.2 as follows:

(1) x_1 is the radius of Mirror 1;

(2) x_2 is the radius of Mirror 2;

(3) d_1 and d_2 are the distance from Mirror 1 to the lens center, O_c and the distance from Mirror 2 to O_c, respectively;

(4) f is the focal length of the perspective camera;

(5) w is the width of the CMOS sensor;

(6) b_1, c_1 are the parameters of Mirror 1; and

(7) $b_{2,}$ c_2 are the parameters of Mirror 2.

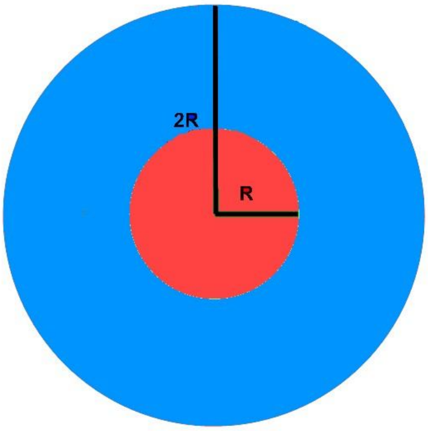


Figure 3.5 The desired omni-image.

In Figure 3.6, the blue lines are drawn to represent the light rays which go into the center of Mirror 1 with an incidence angle of 0° and are reflected onto the image plane (the CMOS sensor). The red lines are similarly interpreted which go into the center of Mirror 2. Then, focusing on the marked area in Figure 3.6, we get two pairs of similar triangles, ($\triangle AO_1O_c$, $\triangle DO_{ci}O_c$) and ($\triangle BO_2O_c$, $\triangle CO_{ci}O_c$). By the property of similar triangles, we can describe the relationship between them by the following equations:

$$\frac{x_1}{d_1} = \frac{\overline{\mathrm{DO}_{\mathrm{ci}}}}{f}, \tag{3.4}$$

$$\frac{x_2}{d_2} = \frac{\overline{\mathrm{CO}_{\mathrm{ci}^{\sim}}}}{f} \tag{3.5}$$

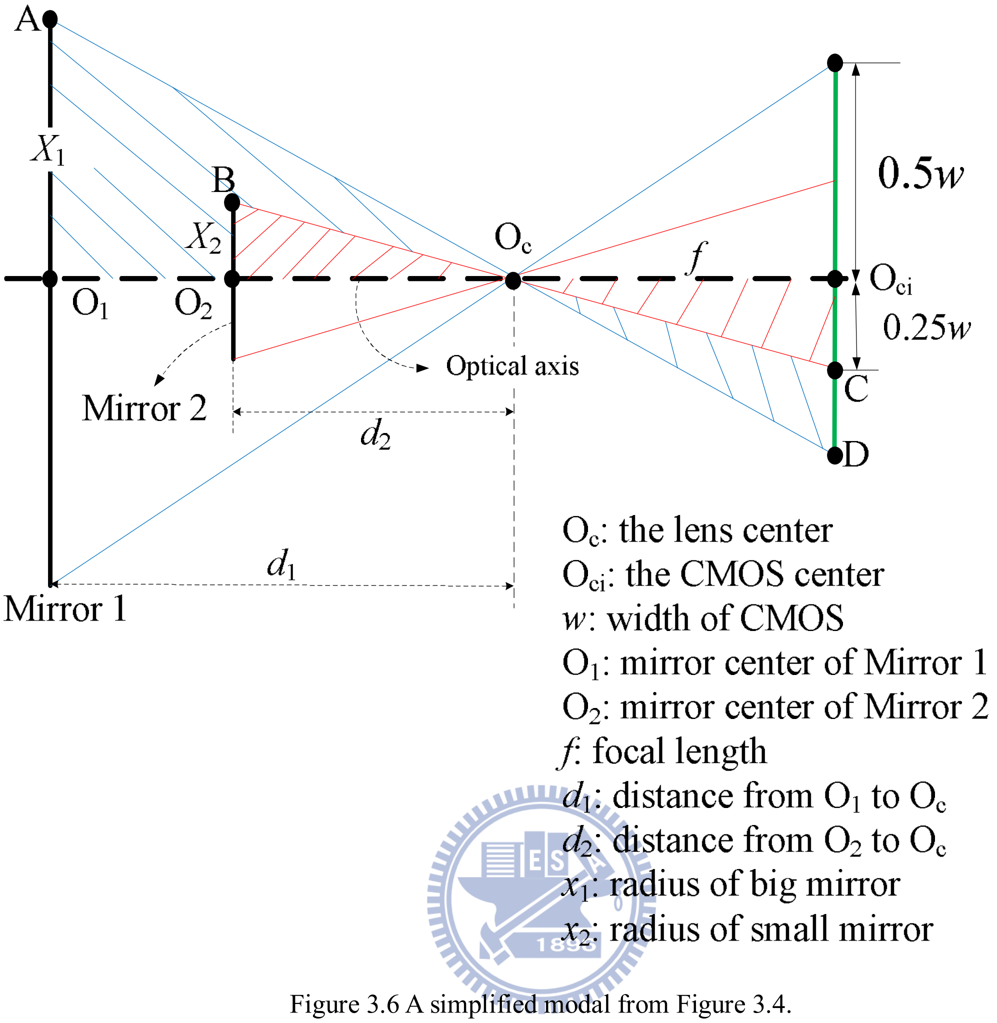

A
B
X_1
X_2
O_1
O_2
O_c
f
O_{ci}
$0.5w$
$0.25w$
C
D
Optical axis
Mirror 2
Mirror 1
d_2
d_1
O_c: the lens center
O_{ci}: the CMOS center
w: width of CMOS
O_1: mirror center of Mirror 1
O_2: mirror center of Mirror 2
f: focal length
d_1: distance from O_1 to O_c
d_2: distance from O_2 to O_c
x_1: radius of big mirror
x_2: radius of small mirror

Figure 3.6 A simplified modal from Figure 3.4.

Step 2: Calculation of the parameters of the mirrors.

The second step is to calculate the parameters of the hyperbolic shape of the mirrors. As shown in Figure 3.4, the relation between the CCS and the ICS of an omni-camera system, as derived by Wu and Tsai [7], may be described by

$$\tan\alpha = \frac{(b^2+c^2)\sin\beta - 2bc}{(b^2-c^2)\cos\beta}, \tag{3.6}$$

$$\cos\beta = \frac{r}{\sqrt{r^2+f^2}}, \tag{3.7}$$

$$\sin\beta = \frac{f}{\sqrt{r^2+f^2}}, \tag{3.8}$$

where $r = \sqrt{u^2 + v^2}$ in the ICS, α is the elevation angle of the mirror, and β is the depression angle of O$_c$.

Let the omni-camera have the largest FOV, and the incidence angle be 0^o, which specifying a light ray going from a point $P(X, Y, Z)$ into the mirror center and is reflected to the image plane. As illustrated in Figure 3.7 which is a simplified version of Figure 3.4, from △O$_1$RO$_c$ we can obtain the following equations:

$$\tan\theta = \frac{x}{d}, \tag{3.9}$$

$$\beta = \frac{\pi}{2} - \theta, \tag{3.10}$$

where d is the distance from the mirror center to the lens center, and x is the radius length of the mirrors. In addition, we can obtain the following equation:

$$\tan\alpha = 0 = \frac{(b^2 + c^2)\sin\beta - 2bc}{(b^2 - c^2)\cos\beta} \tag{3.11}$$

based on Equation (3.6) with 0.

Equation (3.11) may be transformed, by multiplying the denominator by $(b^2 - c^2)\cos\theta$, to be:

$$(b^2 + c^2)\sin\theta - 2bc = 0, \tag{3.12}$$

or equivalently, to be

$$b^2\sin\theta - 2cb + c^2\sin\theta = 0 \tag{3.13}$$

from which we can get the parameter b by the following equations:

$$b = \frac{2c \pm \sqrt{(2c)^2 - 4c^2 \sin^2 \beta}}{2\sin^2 \beta},$$

or equivalently,

$$b = \frac{2c \pm \sqrt{(2c)^2[1 - \sin^2 \beta]}}{2\sin^2 \beta}. \qquad (3.14)$$

Combining Equation (3.10) and Equation (3.14), we get

$$b = \frac{2c \pm \sqrt{(2c)^2[1 - \sin^2(\frac{\pi}{2} - \theta)]}}{2\sin^2(\frac{\pi}{2} - \theta)} \qquad (3.15)$$

which can be simplified in the following way:

$$b = \frac{2c \pm \sqrt{(2c)^2[1 - \cos^2 \theta]}}{2\cos\theta},$$

$$= \frac{2c \pm \sqrt{(2c)^2 \sin^2 \theta}}{2\cos\theta},$$

$$= \frac{2c \pm (2c) \times \sin\theta}{2\cos\theta},$$

$$= \frac{c \times (1 \pm \sin\theta)}{\cos\theta}. \qquad (3.16)$$

Because d is the distance from a focus point of the hyperbolic shape (the mirror center) to the other focus point O_c (the lens center) as shown in Figure 3.6, we can obtain the parameter c of the mirror by

$$c = 0.5 \times d. \qquad (3.17)$$

By equations (3.9) and (3.17), we can solve Equation (3.16) to get

$$b = \frac{0.5\times d\times(1\pm\sin[\tan^{-1}(\frac{x}{d})])}{\cos[\tan^{-1}(\frac{x}{d})]}. \qquad (3.18)$$

Now we can use the general analytic equations, Equations (3.17) and (3.18), to compute the parameters of the mirrors as long as we have the values of x and d.

To calculate the parameters of Mirror 2, we have to make a second assumption, that is, the radius x_2 is a pre-selected value. Accordingly, we can obtain the distance d_2 in Equations (3.5), and then use Equations (3.17) and (3.18) to calculate the parameters b_2 and c_2. And the parameter a_2 may be computed by Equation (3.2).

Finally, we want to get the parameters of Mirror 1. It is mentioned in [12] that a longer *baseline* (shown as $\overline{O_1O_2}$ in Figure 3.6, whose width is equal to $d_1 - d_2$) of the two mirrors will yield a larger *disparity* (the difference between the elevateon angles) of the two mirrors in the image. If the baseline is too smaller, the range data will not be accurate. Therefore, we assume the baseline to be of an appropriate length, denoted say as l_b as shown in Table 3.1, so that we can get $d_1 = d_2 + l_b$ and so the value of x_1 by Equation (3.4). Then, we can get the parameters b_1 and c_1 of Mirror 1 by using the general equations of (3.17) and (3.18), and the parameter a_1 may be computed accordingly by Equation (3.2).

The calculated parameters of the proposed two-mirror omni-camera are shown in Table 3.1 and some simulation images of the camera are shown in Figure 3.8. We use Equation (3.3) to create the simulation images with a focal length of 6 mm and the parameter values listed in Table 3.1. As illustrated in Figure 3.8, the top-left image is the perspective image; the top-right image and the bottom-right image are the simulation images taken by Mirror 1 and Mirror 2, respectively; and the bottom-left image is the simulation image which is composed of the top-right and the bottom-right one.

Table 3.1 The parameters of the proposed two-mirror omni-camera obtained from the design process.

	a	*b*	Baseline l_b
Mirror 1	11.46cm	9.68cm	20cm
Mirror 2	2.41cm	4.38cm	

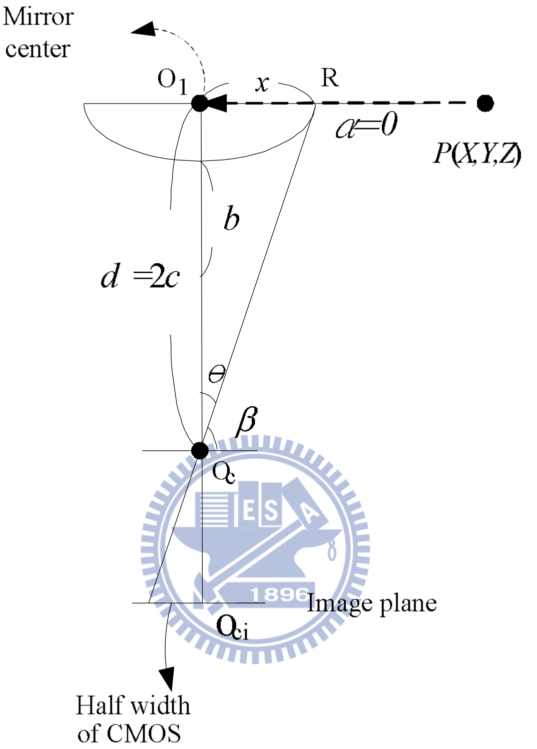


Figure 3.7 The light ray is reflected from the biggest radius from the mirror.

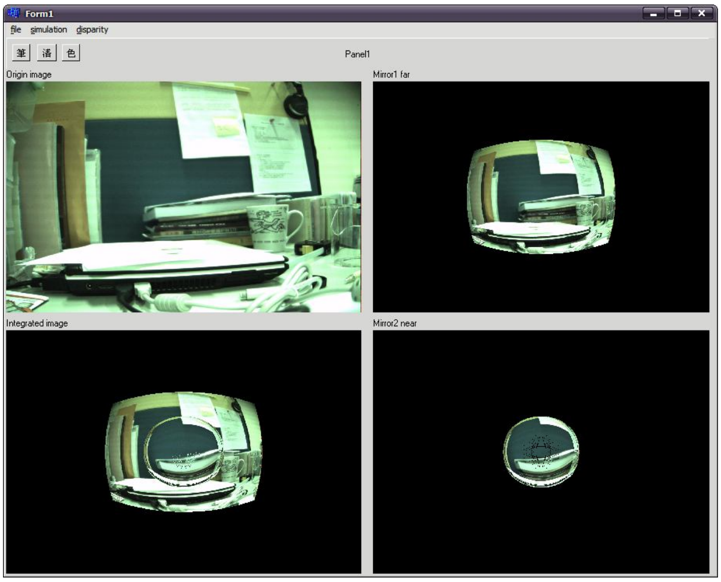


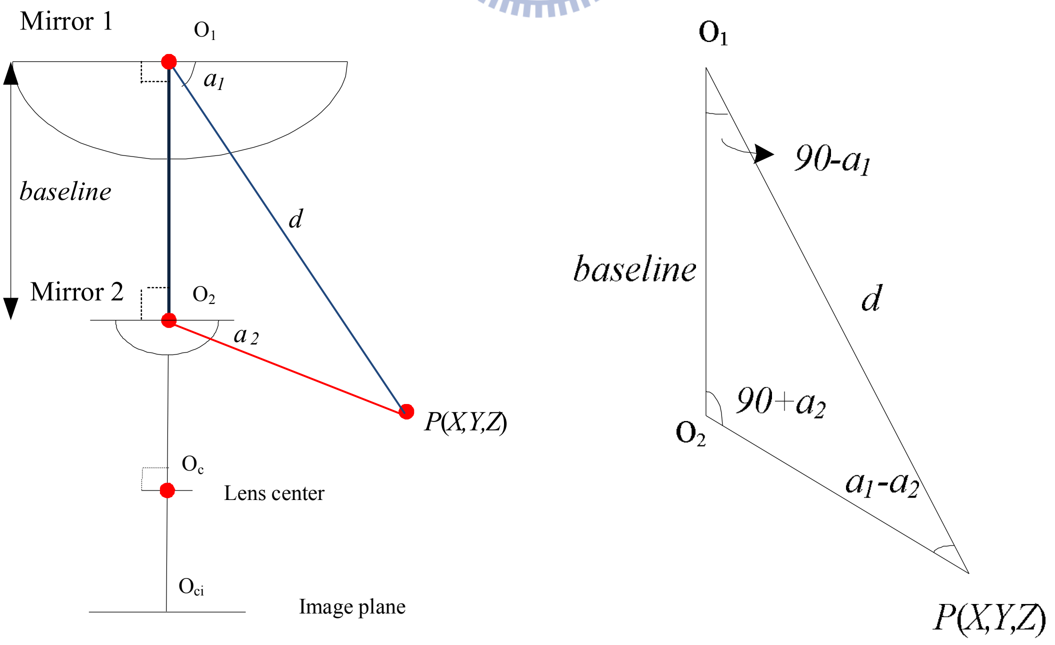
Figure 3.8 The simulation images.

3.2.3 3D data acquisition

In this section, we will introduce the principle of the computation of the 3D range data and the derivation of it in the proposed system. In Figure 3.9(a), the point *P* in the CCS goes through each center of the two mirrors, and α_1 and α_2 are the elevation angles in each mirror, respectively. A triangle $\triangle O_1O_2P$ formed by two light rays (the blue line and the red line) is shown in Figure 3.9(b). The distance *d* is between the point *P* and O_1, and the baseline length of the two mirrors is the distance *h* between O_1 and O_2. We can derive the following equations by the *law of sines* based on the geometry shown in Figure 3.9(b):

$$\frac{d}{\sin(90+\alpha_2)} = \frac{h}{\sin(\alpha_1-\alpha_2)}, \tag{3.12}$$

$$d = \frac{h\cdot\sin(90+\alpha_2)}{\sin(\alpha_1-\alpha_2)}. \tag{3.13}$$



(a) (b)

Figure 3.9 Computation of 3D range data using the two-mirror omni-camera. (a) The ray tracing of a point *P* in the imaging device. (b) A triangle in detail (part of (a)).

Based on Figure 3.9, a front-facing placement of the two-mirror omni-camera on the autonomous vehicle is shown in Figure 3.10 with the CCS specified by the axes of (X, Y, Z) and the ICS specified by the axes of (U, V). The optical axis O_cO_1 is parallel to the ground, where O_c is the camera center, O_2 is the center of Mirror 2, and O_1 is the center of Mirror 1.

From Figure 3.10, the azimuth θ in the ICS can be computed by use of the image coordinates (u, v) of P in the following:

$$\sin\theta = \frac{u}{\sqrt{u^{2}+v^{2}}}; \qquad \cos\theta = \frac{v}{\sqrt{u^{2}+v^{2}}}. \tag{3.14}$$

After we obtain d by Equation (3.13), by involved geometry we can compute the values of (X, Y, Z) by the following equations:

$$\begin{aligned} X &= d \times \cos\alpha_1 \times \sin\theta, \\ Y &= d \times \cos\alpha_1 \times \cos\theta, \\ Z &= d \times \sin\alpha_1. \end{aligned} \tag{3.15}$$

In the proposed system, the optical axis instead is not arranged to be parallel to the ground in order to expand the overlapping area of the both mirrors, and the reason is mentioned in detail in Chapter 2. Figure 3.11 is a side view of Figure 3.10 and is shown to illustrate the slanted-up placement of the two-mirror omni-camera in the proposed system. Specifically, let the optical axis of the camera of the proposed camera system be elevated by the angle of γ, and the position of P(X, Y, Z) be translated to P'(X', Y', Z') in the new coordinate system. Then, according to the trigonometry principle, we can use a rotation matrix R to represent the relationship between P and P' as follows for computing the new coordinates (X', Y', Z'):

$$P' = R \times P;$$

$$R = \begin{bmatrix} \cos(-\gamma) & -\sin(-\gamma) \\ \sin(-\gamma) & \cos(-\gamma) \end{bmatrix}. \tag{3.16}$$

As a summary, we have introduced in this chapter the design principle of the proposed two-mirror omni-camera and derived some equations we used for the related imaging processing and 3D data acquisition processes. We will describe the proposed method to calibrate the designed two-mirror omni-camera in the next chapter.

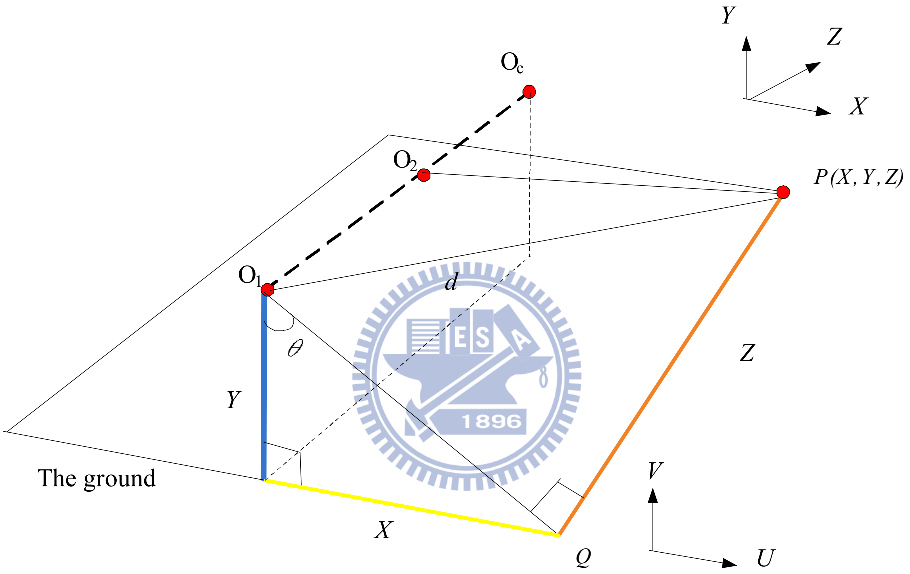


Figure 3.10 The frontal placement of the two-mirror omni-camera.

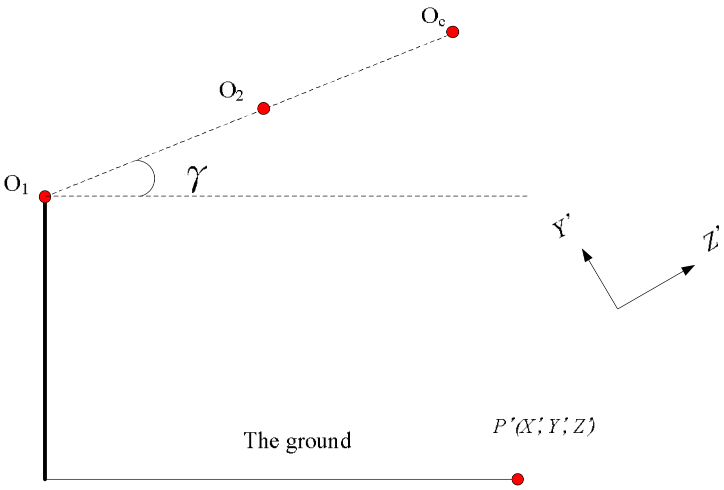


Figure 3.11 A side view of Figure 3.8.

Chapter 4
Calibrations of Vehicle Odometer and Proposed Two-mirror Omni-camera

4.1 Introduction

In this chapter, we describe how we calibrate the two major devices in the proposed system, namely, (1) the odometer which is used to record the position and the direction of the autonomous vehicle, and (2) the proposed two-mirror omni-camera which is used to "see" the situation in environments. We will show the calibration method to correct the vehicle odometer in Section 4.2 and the calibration of the proposed two-mirror omni-camera in Section 4.3.

4.2 Calibration of Vehicle Odometer

4.2.1 Idea of proposed odometer calibration method

Before using any machine, a calibration process for it is usually necessary. When the autonomous vehicle is moving, the information about its direction and position is provided by the odometer in the vehicle. In a short experiment which we conducted for calibration of the odometer, we controlled the autonomous vehicle manually, and recorded the real position reached by the vehicle as well as the odometer readings at the reached spot. We found that the odometer values are not the same as the real position, and that the mechanical error increases as the vehicle's move distance increases.

4.2.2 Problem definition

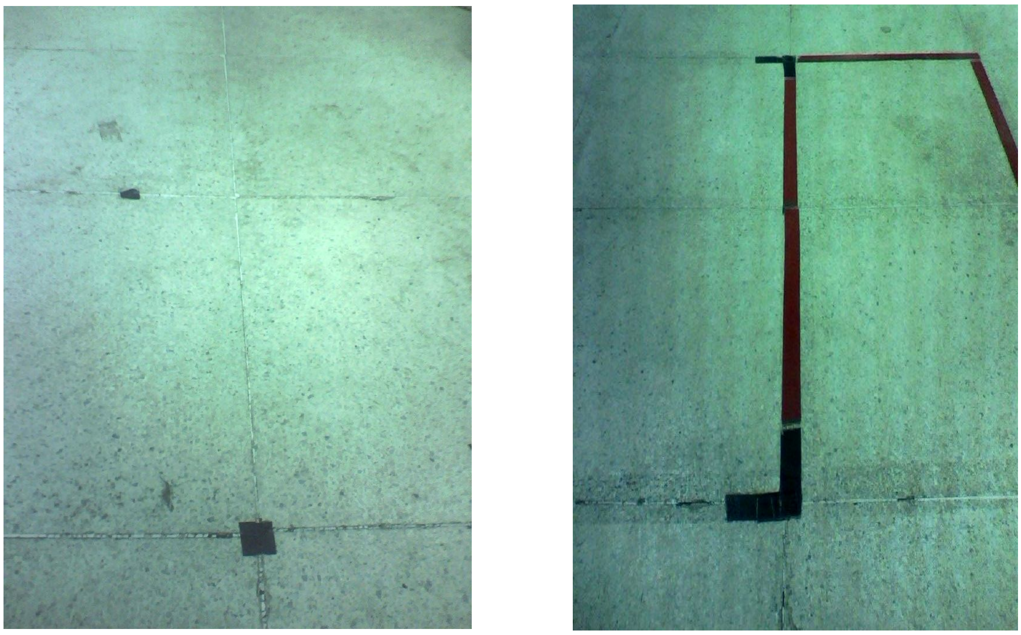
In this study, we have designed a careful experiment to test the mechanical error. We used the tiles on the floor in our laboratory to assist locating the autonomous vehicle in the experiment. Figure 4.1 shows the floor of the laboratory and each tile on it has a square shape with size 30.5cm×30.5cm. The equipments we used in this experiment are sticky tapes, a measuring tape, two laser levelers, and the autonomous vehicle.

The steps of the experiment are described as follows and shown in Figure 4.2.

1. Paste sticky tapes forming a shape of L on the floor by extending the straight lines between several tiles.
2. Use the laser levelers to adjust the heading of the autonomous vehicle to be parallel to a line of the L-shape and let the vehicle lie on the L-shape.
3. Command the vehicle to move forward on the L-shape with a distance of N tiles, where $N = 3, 6, 9, 12$.
4. Stop the vehicle after it moves for a distance of N tiles.

We found the position that the vehicle arrived at is no more on the L-shape and that the moving path is deviated to the left. We then measured the amount of deviation, fit the resulting deviations of several vehicle movements as a curve of the second order, and use the curve to calibrate the odometer reading of the vehicle in subsequent navigations. More details will be described subsequently. The curve fitting technique we use is described in the next section.

4.2.3 Proposed curve fitting for mechanical error correction



(a) (b)

Figure 4.1 The floor of *Computer Vision Laboratory* in National Chiao Tung University. (a) Using the lines on the floor of the laboratory to assist the experiment. (b) Pasting the tape to form an L-shape on the floor.

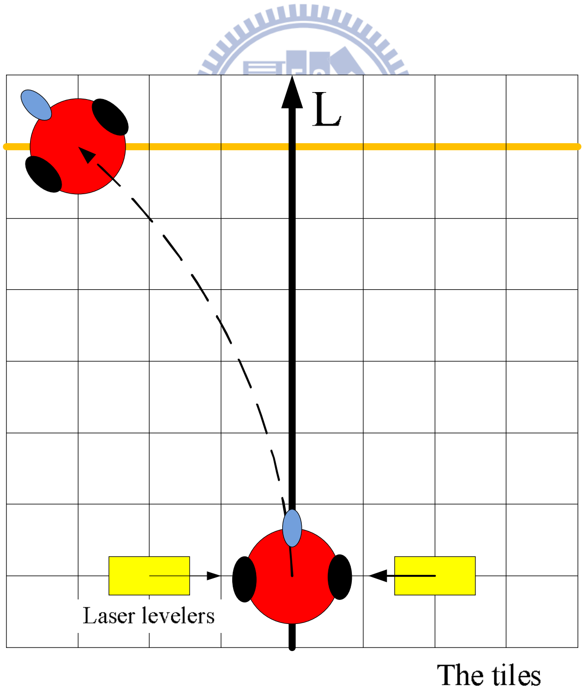


Figure 4.2 An illustration of the experiment.

The curve fitting scheme we used is described in the following.

1. Generate a curve of a *k*th-degree polynomial function:

$$y_i = a_0 + a_1 x_i + \ldots + a_k x_i^k, \text{ where } i = 1, 2, \ldots, n. \quad (4.1)$$

2. Use the least-square-error technique to fit the measured deviation data (x_1, y_1), (x_2, y_2), …, (x_n, y_n) by

$$A = \sum_{i=1}^{n}[y_i - (a_0 + a_1 x_1 + ... + a_k x_i^k)^2]. \tag{4.2}$$

3. To get the coefficients of Equation (4.1), set the partial derivatives of A with respect to a_0 through a_k equal to zero to get

$$\frac{\partial A}{\partial a_k} = -2\sum_{i=1}^{n}[y - (a_0 + a_1 x + ... + a_k x^k)]x^k = 0\,, \tag{4.3}$$

which leads to

$$y\sum_{i=1}^{n} x_i^k = a_0 \sum_{i=1}^{n} x_i^k + a_1 \sum_{i=1}^{n} x_i^{k+1} + ... + a_k \sum_{i=1}^{n} x_i^{2k}\ . \tag{4.4}$$

4. Transform Equation (4.1) into a matrix form $XA = Y$ as follows:

$$\begin{bmatrix} 1 & x_1 & \dots & x_1^k \\ 1 & x_2 & \dots & x_2^k \\ 1 & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^k \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}. \tag{4.5}$$

5. Multiply both sides of (4.5) above by a transpose of X to obtain the following equation:

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1^k & x_2^k & \dots & x_n^k \end{bmatrix} \begin{bmatrix} 1 & x_1 & \dots & x_1^k \\ 1 & x_2 & \dots & x_2^k \\ 1 & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^k \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1^k & x_2^k & \dots & x_n^k \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{4.6}$$

leading to

$$\begin{bmatrix} n & \sum_{i=1}^{n} x_i & \cdots & \sum_{i=1}^{n} x_i^k \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 & \cdots & \sum_{i=1}^{n} x_i^{k+1} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} x_i^k & \sum_{i=1}^{n} x_i^{k+1} & \cdots & \sum_{i=1}^{n} x_i^{2k} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_1 \\ \sum_{i=1}^{n} x_i y_i \\ \vdots \\ \sum_{i=1}^{n} x_i^k y_i \end{bmatrix}. \tag{4.7}$$

which is a the matrix form of Equation (4.4).

6. Rewrite Equations (4.5) through (4.7) in the following ways:

$$XA = Y, \tag{4.8}$$

$$X^{\mathrm{T}}XA = X^{\mathrm{T}}Y, \tag{4.9}$$

leading to the following solution for the problem:

$$A = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y. \tag{4.10}$$

4.2.4 Proposed calibration method

To obtain the data of deviation distances, we conducted the steps of the experiment mentioned in Section 4.2.2 and used the curve fitting technique to build a calibration model of the odometer described by Equation (4.1) in which the coefficients a_0 through a_k may be obtained by Equation (4.10). A list of the deviation data of our experiment are shown in Table 4.1 and the curve fitting result described by Equation (4.1) through Equation (4.10) is shown in Figure 4.3. The final calibration model is $y = 0.00004x^2 + 0.00096x - 0.4$.

The overall odometer calibration process is described as an algorithm in the following.

Algorithm 4.1. Building a calibration model of the odometer.

Input: An incremental number $N = 1$.

Output: A calibration model of the odometer described by Equation (4.1) through (4.10).

Steps:

Step 1. Paste sticky tapes to form an L-shape on the ground by extending the straight lines between the ground tiles.

Step 2. Use the laser levelers to adjust the heading of the autonomous vehicle to be parallel to a line of the L-shape and let the vehicle lie on the shape.

Step 3. Command the vehicle to move forward on the L-shape with a distance of N tiles, stop the vehicle after it moves for a distance of N tiles, and measure the deviation distance between the vehicle and the L-shape, for all $N = 1, 2, \ldots, 17$.

Step 4. Repeat Step3 for three times.

Step 5. Use the curve fitting technique to build a calibration model of the odometer.

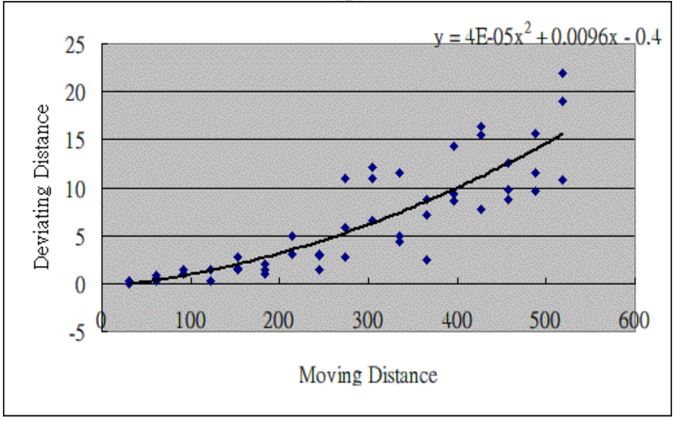


Figure 4.3 The calibration model of the odometer built by the curve fitting technique.

4.3 Calibration of Designed Two-mirror Omni-camera

4.3.1 Problem definition

The distance information is important for the autonomous vehicle while the vehicle is navigating in the environment. In the proposed method, the autonomous vehicle can detect the guide line and navigate along it. The autonomous vehicle can avoid the obstacles in the environment. As we know that the images that the camera captured are projected from the 3D *world coordinate system* or *Global coordinate system* (*GCS*) onto the 2D image coordinate system (ICS), it is necessary to conduct a calibration work for the camera to establish a relationship between the 3D world coordinates and the 2D image coordinates.

Table 4.1 The data of the experiment of building the calibration model of the odometer.

Move Distance (cm)	Distance of Deviation (cm)	Distance of Deviation (cm)	Distance of Deviation (cm)
30.5 × 1	0.2	0	0
30.5 × 2	0.8	0.5	0.2
30.5 × 3	1	1	1.5
30.5 × 4	0.3	1.5	1.5
30.5 × 5	1.5	1.6	2.8
30.5 × 6	2	1.5	1
30.5 × 7	3	3.1	5
30.5 × 8	3.1	2.9	1.5
30.5 × 9	2.8	5.9	10.9
30.5 × 10	6.5	10.9	12.1
30.5 × 11	4.3	11.6	5
30.5 × 12	8.8	2.5	7.2
30.5 × 13	9.4	8.6	14.3
30.5 × 14	15.5	16.3	7.8
30.5 × 15	12.5	8.7	9.8
30.5 × 16	15.6	9.7	11.6
30.5 × 17	19	22	10.8

4.3.2 Idea of proposed calibration method

As mentioned in Figure 3.4, the light ray of a point *P*(*X*, *Y*, *Z*) in the CCS first goes onto the mirror center O_m, then is reflected to the lens center O_c by the hyperbolic-shaped mirror surface, and finally is projected on the image plane to form the image point *q*(*u*, *v*) in the ICS. The elevation angle shown in Figure 3.4 is formed by the incoming light ray which goes from P to O_m. The idea here is to design a model to describe the relationship between the elevation angle and the point q in the ICS. The proposed calibration method is described in detail in the next section.

4.3.3 Proposed calibration process

In [7], Wu and Tsai mentioned two misalignments in an omni-camera system. The first is the so-called *axial-directional misalignment* and the other is the so-called *lateral-directional misalignment*. In an axial-directional misalignment, the optical axis of the camera does not go through the mirror center; instead, it is parallel to the mirror axis. The omni-image center is shifted a distance from the lens center in this phenomenon. In a lateral-directional misalignment, the mirror axis is not parallel to the optical axis of the camera. This misalignment of the mirror axis not only destructs the SVP property but also the *rotational invariance property* in omni-images.

A prototype of the proposed two-mirror omni-camera is shown in Figure 4.4 which is a modified version of Figure 3.4. To avoid the two misalignment problems mentioned previously, two properties are necessary to be satisfied: (1) the optical axis of the camera is maintained to be perpendicular to the two mirrors; and (2) the optical axis of the camera is aligned with the mirror axis $\overline{O_1O_2}$. There are three purposes we have to achieve to calibrate the proposed two-mirror omni-camera as follows.

Purpose 1 — making the optical axis of the camera perpendicular to the two mirror bases with the lens center and the mirror center both aligned with the vertical axis.

Some equipments as shown in Figure 4.5 are prepared in advance for works to achieve the goal of "*making the optical axis of the camera perpendicular to the two mirror bases with the lens center and the mirror center both aligned with the vertical* *axis*," including:

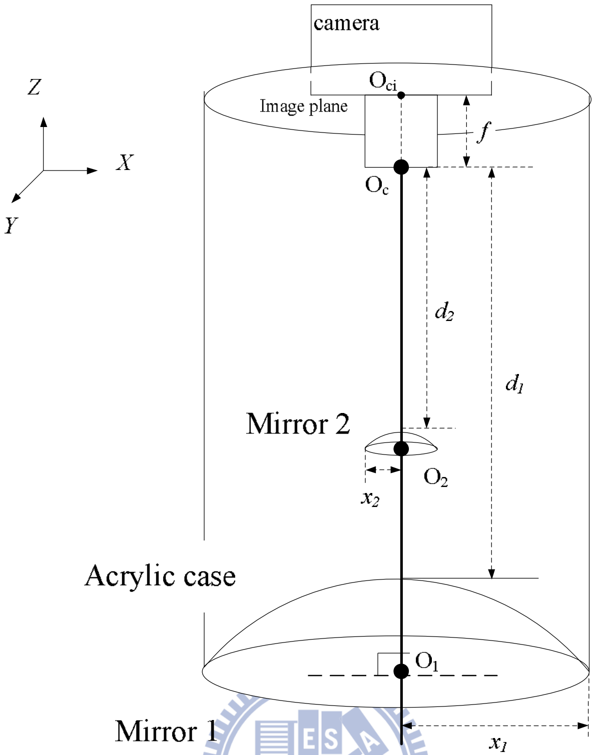


Figure 4.4 An illustration of the proposed two-mirror omni-camera.

(1) a horizontal leveler; (2) a planar mirror M; (3) a blueprint of Mirror 1, C_1; (4) a transparent plastic enclosure; and (5) a perspective camera. Mirror 1 was produced according to the design shown by the blueprint of Figure 4.5(b). The red circles on the outer ring of the base of the enclosure in the figure are fixed points, and the enclosure is affixed with steel nails punched through these fixed points. The enclosure is used to cover the mirrors in the camera system to prevent rain and unexpected touch from the environment.

Referring to Figure 4.5, we define several notations in advance. The black circle C_2 in Figure 4.5(b) is the image of the camera lens reflected by the planar mirror M. CC_1 is the center of C_1 which is the crossing point by the two dotted lines in Figure 4.5, and CC_2 is the center of C_2 which is detected by a binary thresholding technique.

We use four steps to achieve Purpose 1 in the following:

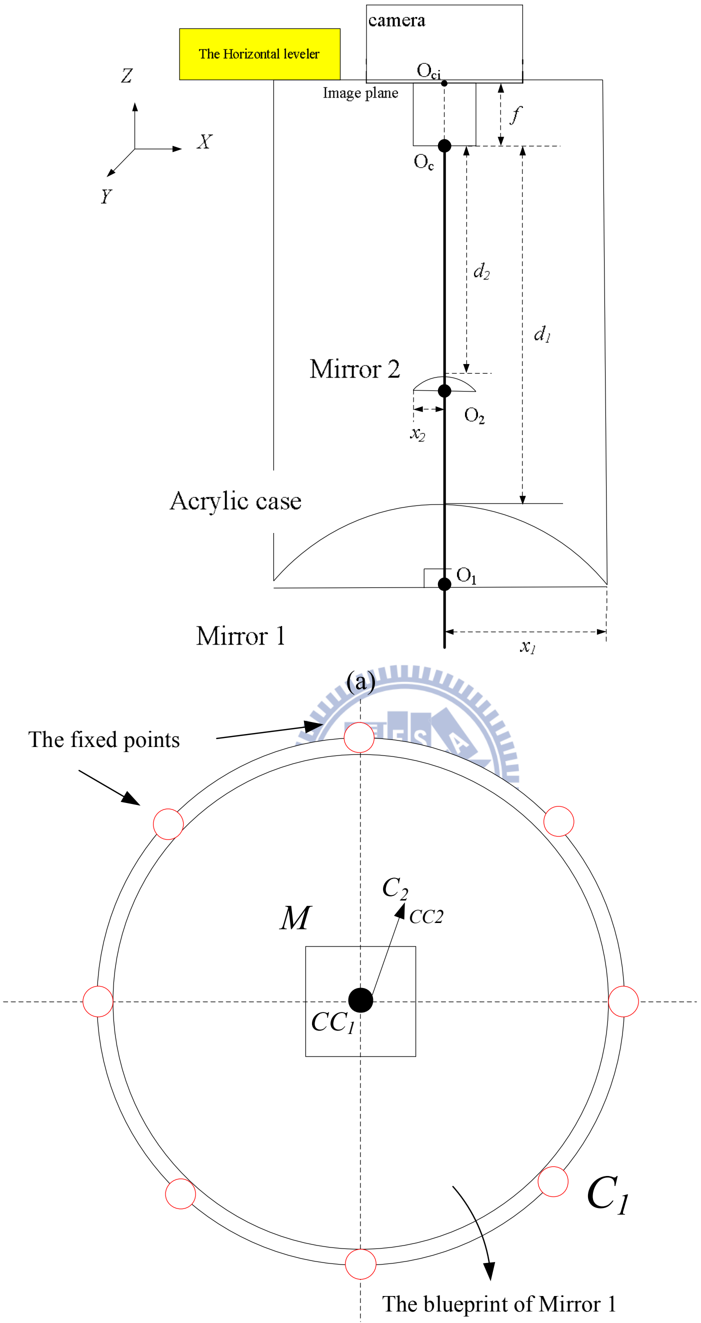
1. Use the horizontal leveler to make sure that the top cover of the enclosure is horizontal to the base of the bigger mirror;
2. Embed the perspective camera in the top cover of the enclosure and take images, while the camera is not fixed in it yet (see the illustration shown in Figure 4.4);
3. Overlap CC_2 on CC_1 by moving the perspective camera to make $\overline{O_1O_c}$ perpendicular to the base of Mirror 1 in Figure 4.4.
4. Fix the perspective camera now.

Purpose 2 － making the smaller mirror base parallel to that of the bigger one and their axes aligned.

In the last step, we make the perspective camera parallel to Mirror 1, the optical axis of the camera perpendicular to Mirror 1, and the lens center and the center of Mirror 1 to align with the vertical axis, as shown in Figure 4.5(a). In this step, we want to make the base of Mirror 2 parallel to that of Mirror 1. If this can be achieved, there is no alignment problem about the axes of the two cameras and Mirror 1, Mirror 2 and the lens are parallel to each other.

We use the rotational invariance property of the omni-image in this step and illustrate it with Figure 4.6. The property says: if the two mirrors, Mirror 1 and Mirror 2, are parallel, two image points q_1 and q_2 in the ICS, which are projected by both mirrors from a single scene point P in the GCS are on an identical radial line in the image (the blue line and the red line in the image plane shown in Figure 4.6).

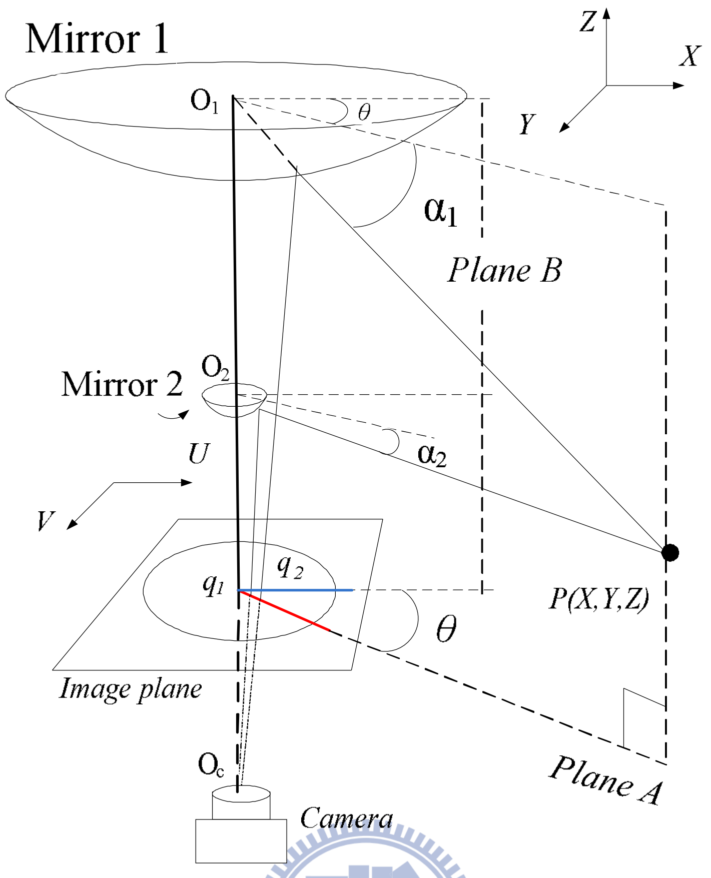
Based on this property, we attached several pieces of paper strips on the enclosure to test if the base of Mirror 2 parallel to that of Mirror 1. If not, the edge lines of these paper strips will not be in the same radial directions (i.e., will not align respectively) in the omni-image, as shown in Figure 4.7(a). The contrary case with the edge lines aligned is shown in Figure 4.7(b).



(a)

(b)

Figure 4.5 The first step of the proposed calibration method. (a) Make sure the enclosure is parallel to the Mirror 1. (b) Make sure the lens center is match on the mirror center.

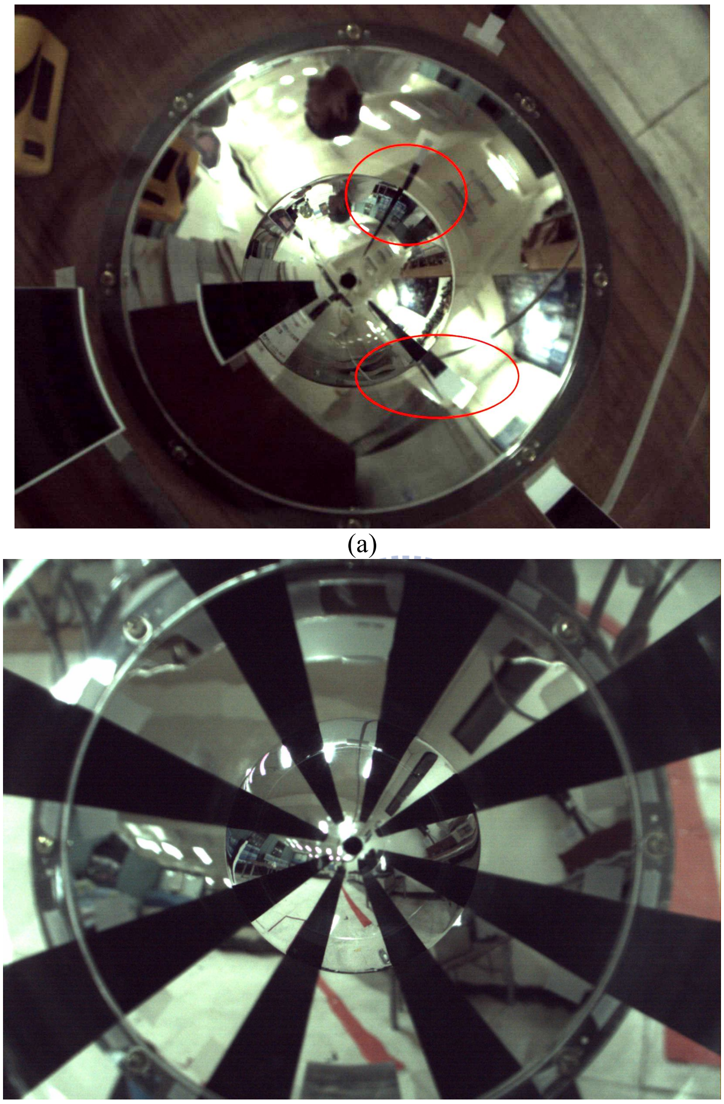


O_1: the center of Mirror 1
O_2: the center of Mirror 2
O_c: the center of the camera lens

Figure 4.6 The property, rotation invariance of the omni-camera.

Purpose 3 — Recording the relationship between the elevation angle and the image coordinates.

An idea of a *mapping-based* approach proposed by Jeng and Tsai [22] is adopted for the purpose of establishing the relationship between the elevation angle and the image coordinates for use in this study. This idea does not conduct camera calibration to estimate camera *intrinsic* and *extrinsic* parameters, but creates a so-called *pano-mapping table* as a substitute for use in *image unwarping*. We propose a calibration method based on the pano-mapping table to record the relationship between the elevation angle of the mirror and the image coordinates. As illustrated in Figure 4.6, let a point in the GCS be denoted as (α_1, θ) and (α_2, θ) for Mirror 1 and Mirror 2, respectively, with the pairs (α_1, θ) and (α_2, θ) describing the elevation angles and the azimuth angles, respectively.


(a)

(b)

Figure 4.7 Experimental results of mirror base parallelism test. (a) An omni-image captured by the two-mirror omni-cameras *not* parallel (b) An omni-image captured by the two-mirror omni-cameras after being made parallel.

Assume that P is a landmark point in the plane A. Then, we can get an infinite set S_{ij} of points, denoted as $P(X, Y, Z)$, in the GCS with the elevation angle α_i and the azimuth θ_j; as well as a set Q_{ij} of the corresponding projection points $q(u, v)$ in the ICS for Mirror 1. Similarly, we can get an infinite set S_{ij}' of points in the GCS with the elevation angle α_i' and azimuth θ_j'; as well as a set Q_{ij}' of the corresponding projection points in the ICS for Mirror 2. The projection points in Q_{ij} and Q_{ij}' are all on the same radial line in the ICS by the rotational invariance property, therefore, θ_j is equal to θ_j'.

Based on this property and the data mentioned, we may specify the relationship between the elevation angles and the radial distance by using a nonlinear function f_r, and use a table to record the relationship between the elevation angle and the pixel as shown in Table 4.2. Each entry E_{ij} of Table 4.2 is the the pixel in the ICS and the entry value is the elevation angle α_i, where W is the width of the input image and H is the height of the input image.The mapping table is established by the following altorithm.

Algorithm 4.2. Establishment of the adopted pano-mapping table.

Input: (S_{ij}, Q_{ij}) and (S_{ij}', Q_{ij}').

Output: A mapping table for camera calibration.

Steps:

Step 1. Create a non-linear function to map the elevation angles and the radial length data as follows:

$$r_i = f_r(\alpha_i) = a_0 + a_1\times \alpha_i^1 + a_2\times\alpha_i^2 + a_3\times\alpha_i^3 + a_4\times\alpha_i^4 + a_5\times\alpha_i^5, \qquad (4.11)$$

where r_i is computed by the coordinates of the projection point in Q_{ij} as $r_i^2 = u_i^2 + v_i^2$, and α_i is computed by the coordinates of the point in S_{ij} as $\alpha_i = \tan^{-1}(\frac{Z_i}{\sqrt{X_i^2+Y_i^2}})$ as illustrated in Figure 4.8.

Solve the equations described by Equation (4.11) to get the coefficients (a_o, a_1, $a_{2,}$ $a_{3,}$ a_4, a_5) of f_r by the use of Equations (4.8) through (4.10).

Step 2. Calculate α_{max} and $\alpha_{\min}$ in S_{ij} respectively and divide the range $[\alpha_{max}, \alpha_{min}]$ of the elevation angle into N intervals and compute the ith elevation angle by

$$\alpha_i = i\times[(\alpha_{max} - \alpha_{min}) / N] + \alpha_{min}, \text{ for } i = 0, 1, \ldots, N\!-\!1. \qquad (4.12)$$

Step 3. Compute $r_i = f_r(\alpha_i)$ by Equation (4.11) to find the corresponding radial length.

Step 4. Scan each point $q_i(u, v)$ in the ICS, compute the radical radial R_i from the image center, and compare it with r_i, by the following decision rule:

"for each point $q_i(u, v)$ in the ICS, assign its elevation angle α_i into E_{ij}, if $d(R_j, r_i)$ is minimum," (4.13)

where the function $d(a, b)$ is computed as the Euclidean distance between a and b.

Step 5. Repeat Steps 1 through 4 with the landmark set S_{ij}' and the projection point set Q_{ij}' for Mirror 2.

Table 4.2 The proposed calibration table.

α_{11}	α_{12}	…	α_{1W}
α_{21}	α_{22}	…	α_{2W}
…	…	…	…
α_{H1}	α_{H2}	…	α_{HW}

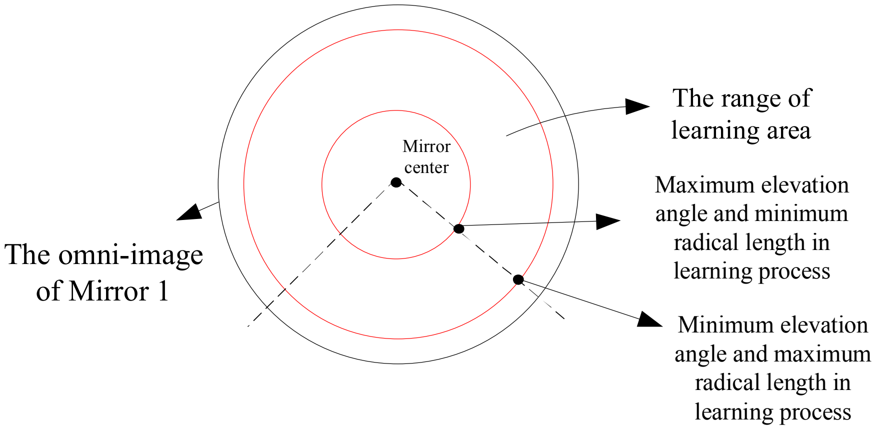

The range of learning area
Mirror center
Maximum elevation angle and minimum radical length in learning process
The omni-image of Mirror 1
Minimum elevation angle and maximum radical length in learning process

Figure 4.8 The illustration of the learning process of the pano-mapping method.

4.4 Experimental Results

In this section, we show some experimental results of the proposed calibration method for the proposed two-mirror omni-camera. Result 1 is the coefficients of the proposed non-linear function. Result 2 is the elevation angles corresponding to the pixels. Result 3 is the error ratios in landmark location estimations.

Result 1:

The obtained coefficients of the proposed non-linear function $f_r(\alpha_i)$ in Equation (4.12) are listed in the Table 4.3, and in the computation we use radians instead of degrees as units for the elevation angles .

Table 4.3 The coefficients of f_r.

a_0	820.955200195313	a_3	-33481.98046875
a_1	-3812.36010742188	a_4	34658.2734375
a_2	15473.1533203125	a_5	-13728.5224609375

Result 2:

The elevation angles are shown in Figure 4.9. We use degrees of saturation of red color to indicate the different magnitudes of the elevation angles, with red color with the higher saturation representing larger elevation angles and vice versa. There are three circles in Figure 4.9, the biggest one representing the omni-image portion reflected by the bigger mirror, the medium one representing the omni-image portion reflected by the smaller mirror, and the smallest one representing the omni-image portion reflected by the camera lens in the omni-camera. The elevation angles from far to nearby with respect to the center of the circles are given red color from less saturated to more.

Result 3:

The error ratios in landmark location estimations using the result of the proposed calibration method are shown in Tables 4.4 and 4.5. Table 4.4 includes the locations of the calibration landmark and the estimated data of their locations, Table 4.5 is the computed error ratios in these location estimation data. As can be seen, the errors are relatively small, and will not affect the effectiveness of the proposed two-mirror omni-camera in our application of vehicle navigation for person guidance.

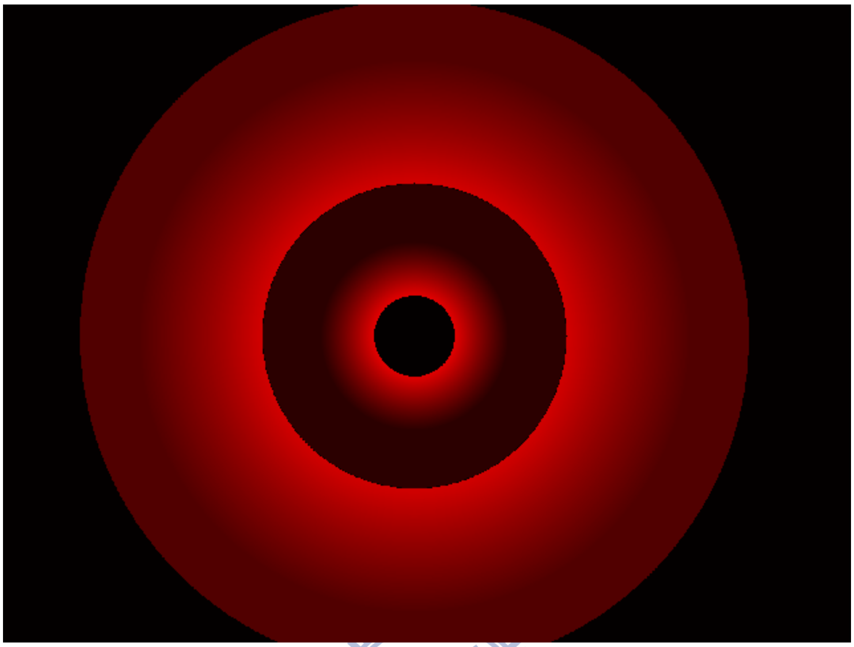


Figure 4.9 The graphical representation identifies the elevation angles.

Table 4.4 Locations (*X*, *Y*, *Z*) of the calibration landmarks and the estimated location (*estX*, *estY*, *estZ*).

X	*Y*	*Z*	*estX*	*estY*	*estZ*
-120	-57	-80	-102.96	-45.203	-77.43
-100	-57	-80	-95.47	-52.73	-82.13
-80	-57	-80	-82	-57.37	-84.11
-60	-57	-80	-64.11	-61.43	-87.62
-40	-57	-80	-42.38	-60.72	-85.96
-20	-57	-80	-20.69	-59.42	-85.01
0	-57	-80	1.039356	-55.1732	-79.40129212
20	-57	-80	20.37	-54.45	-78.44
40	-57	-80	37.84	-52.02	-75.45
60	-57	-80	52.89	-48.01	-72.38
80	-57	-80	77.28	-51.71	-78.62
100	-57	-80	97.4	-50.91	-81
120	-57	-80	104.99	-42.91	-76
-120	-57	-120	-116.55	-54.86	-112.1
-100	-57	-120	-64.39	-27.5	-69.2
-80	-57	-120	-81.1	-57.069	-124.32

-60	-57	-120	-62.6	-59.49	-128.34
-40	-57	-120	-41.39	-59.01	-127.76
-20	-57	-120	-19.1	-57.46	-124.34
0	-57	-120	-0.32	-60.1	-128.62
20	-57	-120	20.04	-53.32	-115.89
40	-57	-120	37.9	-46.22	-109.54
60	-57	-120	55.92	-48.93	-109.78
80	-57	-120	74.57	-50.141	-111.2
100	-57	-120	84	-43.76	-101
120	-57	-120	103.35	-45.17	-103.92

Table 4.5 Error ratios in location estimations with calibration landmarks.

	errX	*errY*	*errZ*	*abs(errX)*	*abs(errY)*	*abs(errZ)*
	-14%	-21%	-3%	14%	21%	3%
	-5%	-7%	3%	5%	7%	3%
	3%	1%	5%	3%	1%	5%
	7%	8%	10%	7%	8%	10%
	6%	7%	7%	6%	7%	7%
	3%	4%	6%	3%	4%	6%
	N/A	-3%	-1%	NA	3%	1%
	2%	-4%	-2%	2%	4%	2%
	-5%	-9%	-6%	5%	9%	6%
	-12%	-16%	-10%	12%	16%	10%
	-3%	-9%	-2%	3%	9%	2%
	-3%	-11%	1%	3%	11%	1%
	-13%	-25%	-5%	13%	25%	5%
	-3%	-4%	-7%	3%	4%	7%
	-36%	-52%	-42%	36%	52%	42%
	1%	0%	4%	1%	0%	4%
	4%	4%	7%	4%	4%	7%
	3%	4%	6%	3%	4%	6%
	-4%	1%	4%	4%	1%	4%
	N/A	5%	7%	NA	5%	7%
	0%	-6%	-3%	0%	6%	3%
	-5%	-19%	-9%	5%	19%	9%
	-7%	-14%	-9%	7%	14%	9%
	-7%	-12%	-7%	7%	12%	7%
	-16%	-23%	-16%	16%	23%	16%
	-14%	-21%	-13%	14%	21%	13%
Average	-5%	-9%	-3%	7%	11%	7%

Chapter 5
Supervised Learning of Navigation Path by Semi-automatic Navigation and Hand Pose Guidance

5.1 Idea of Proposed Supervised Learning Method

The main purpose of the learning procedure is to create a path map while instructing the vehicle to navigate in an environment to learn. In this chapter, we propose two techniques to detect the guide line (the curbstone) in the outdoor environment. And if the color of the curbstone is too similar to the ground color to be detected or if there is no special color in the environment, we propose a new technique of detecting the hand pose in front of or on top of the camera enclosure to instruct the vehicle to navigate properly in the environment.

In addition, when navigating in outdoor environments, the vehicle often encounters the problem of varying light intensity in the environment which has a serious influence on the image analysis work. We propose a method, called dynamic exposure adjustment, to adjust the exposure of the camera automatically and propose another method, called dynamically thresholding adjustment, to adjust the image threshold value used in the hand pose detection procedure in the outdoor environment. An illustration of the learning procedure is in Figure 5.1.

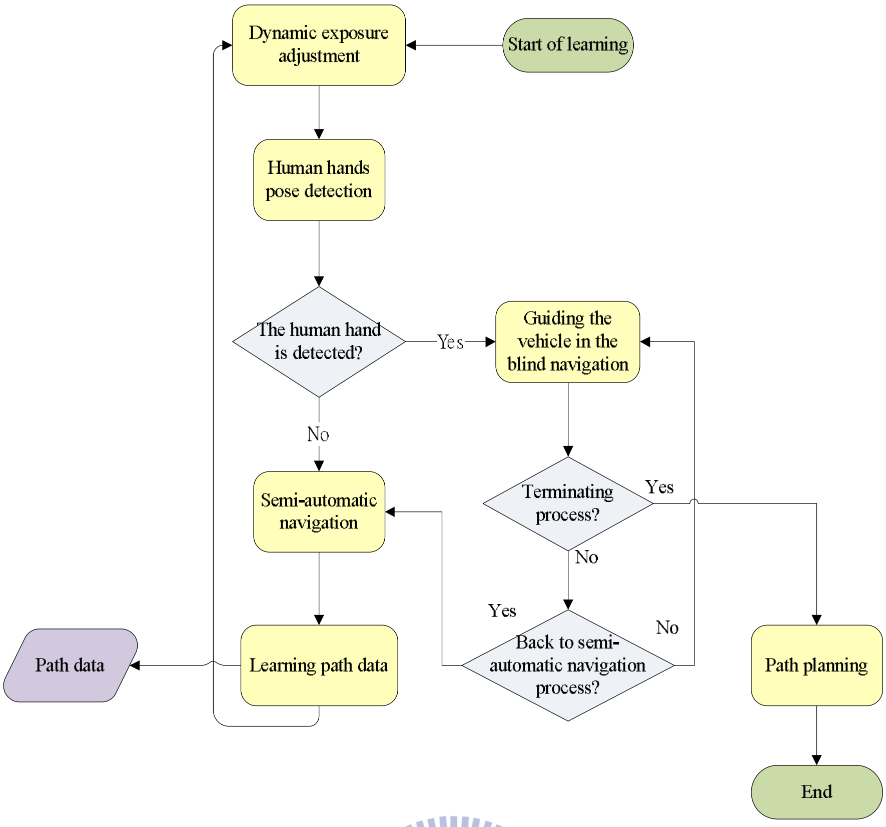

Dynamic exposure adjustment
Start of learning
Human hands pose detection
The human hand is detected?
Yes
Guiding the vehicle in the blind navigation
No
Semi-automatic navigation
Terminating process?
Yes
No
Yes
No
Back to semi-automatic navigation process?
Path data
Learning path data
Path planning
End

Figure 5.1 An illustration of the learning procedure.

5.2 Coordinate Systems

In this section, we summarize the four coordinate systems utilized in this study to describe the relationship between the devices and the navigation environment. The coordinate systems are illustrated in Figure 5.2 and the definitions of all the coordinate systems are described in the following.

(1) *Image coordinate system* (*ICS*): denoted as (u, v). The u-v plane coincides with the image plane and the origin O_I of the ICS is placed at the center of the image plane.

(2) *Vehicle coordinate system* (*VCS*): denoted as (V_x, V_y). The V_x-V_y plane is coincident with the ground. The V_x-axis of the VCS is parallel to the line segment joining the two driving wheels. The origin O_V is placed at the middle of the line segment. The V_y-axis is perpendicular to the V_x-axis and through the

origin O$_V$.

(3) *Global coordinate system* (*GCS*): denoted as (x, y). The origin O$_G$ of the GCS is a pre-defined point on the ground. In this study, we define O$_G$ as the starting position of the vehicle navigation.

(4) *Camera coordinate system* (*CCS*): denoted as (X, Y, Z). The origin O$_1$ of the CCS is a focal point of Mirror 1. And the X-Y plane coincides with the image plane and the Z-axis coincides with the optical axis of the camera.

The GCS is determined when a navigation session is started. The CCS and VCS follow the vehicle during navigation. The coordinate transformation between the VCS and the GCS can be described by the following equations:

$$x = V_x \times \cos\theta + V_y \times \sin\theta + x_p, \qquad (5.1)$$
$$y = V_x \times \sin\theta + V_y \times \cos\theta + y_p, \qquad (5.2)$$

where (x_p, y_p) represent the coordinates of the vehicle in the GCS, and θ is the directional angle between the positive direction of the x-axis in the GCS and the positive direction of the V_x-axis in VCS.

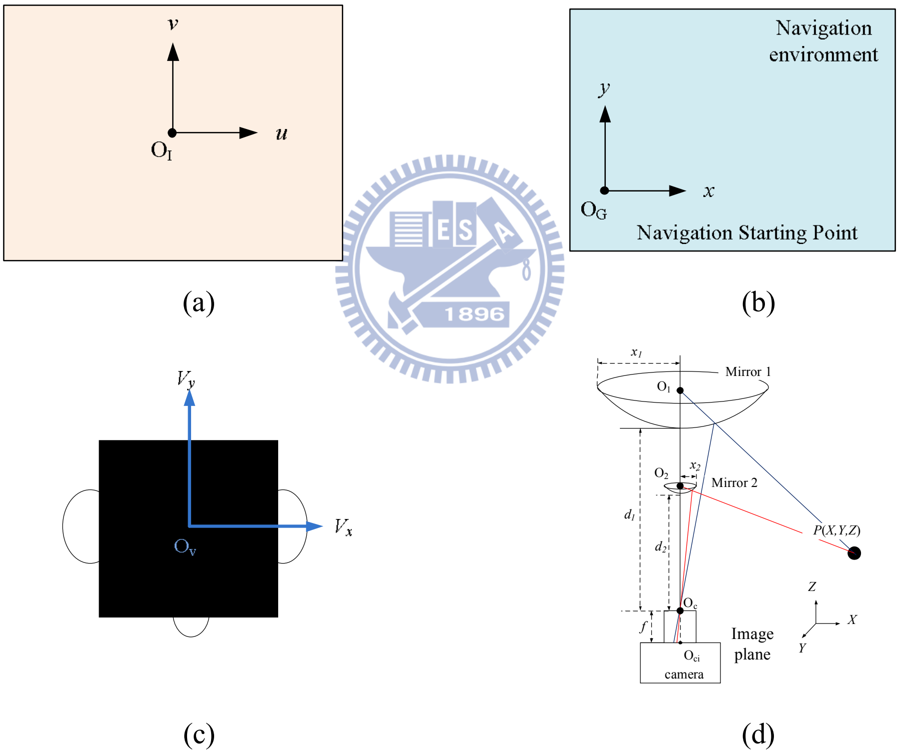
5.3 Proposed Semi-automatic Vehicle Navigation for Learning

5.3.1 Ideas and problems of semi-automatic vehicle navigation

In the thesis study, we use the curbstone features to guide the autonomous vehicle to navigate in the outdoor environment, and we have at least three problems in the following to solve before this idea can be implemented.

- Features are not easy to detect with varying light intensities in the outdoor environment, and how should the autonomous vehicle adjust itself appropriately to deal with the varying light intensities?
- How does the autonomous vehicle extract curbstone features in the outdoor environment automatically?
- How does the autonomous vehicle compute range data to move in the outdoor environment automatically?

We propose several techniques to solve these problems in the following sections.



(a) (b)

(c) (d)

Figure 5.2 The coordinate systems used in this study. (a) The image coordinate system. (b) The global coordinate system. (c) The vehicle coordinate system. (d) The camera coordinate system.

5.3.2 Adjustment of the exposure value of the camera

In the outdoor environment, the light is changing. The varying light influences the navigation experiment very much. The dynamic exposure adjustment we propose to adjust the exposure value of the camera dynamically is carried out with four major steps as discussed in the following:

1. create a so-called *basic image* where the features can be detected as well and then the local intensity I_{basic} can be recorded into the memory storage;
2. record the intensity of each input image I_i with different exposure values Exp_i before the navigation procedure started;
3. generate a linear function f_{exp} to fit I_i and Exp_i using a *line fitting technique*;
4. adjust the exposure value dynamically in the navigation procedure.

To increase the efficiency, we use a *bisecting method* to adjust the exposure value of the camera automatically when generating f_{exp}. The line fitting scheme we used is like that we used in Section 4.2.3 and is described in the following.

1. Generate a line equation:

$$y = mx + b, \tag{5.3}$$

where m is the slope and b is the constant of the line.

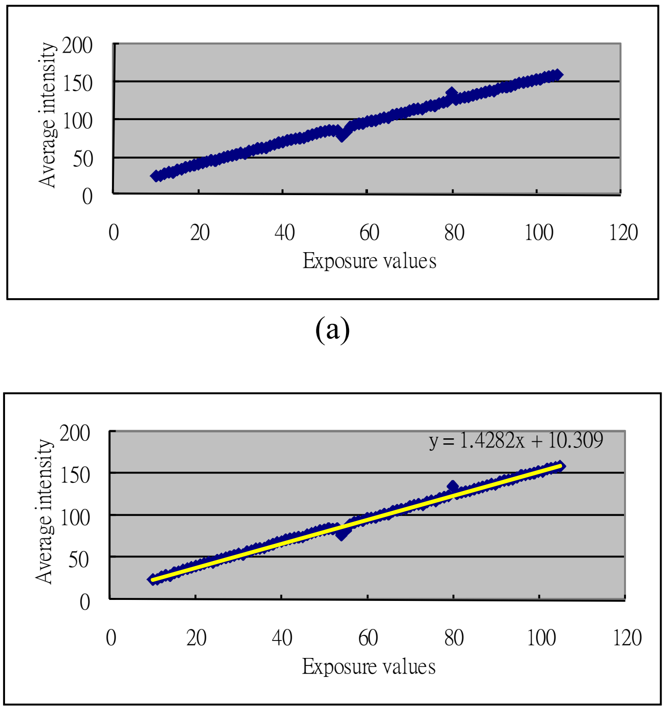
2. Compute m and b by

$$m = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}, \tag{5.4}$$

$$b = \frac{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} x_i y_i \sum_{i=1}^{n} x_i}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \tag{5.5}$$

where x_i is the intensity value of I_i and y_i is the exposure value of Exp_i for n data items.

Figure 5.3(a) shows the relationship between the exposure Exp_i of the camera and the intensity value I_i of the input image for 105 data items. Figure 5.3(b) shows the result of line fitting using the data of Figure 5.3(a).



(a)

(b)

Figure 5.3 The relationship between the exposure of the camera and the average of the intensity of the input image. (a) The distributed data. (b) The distributed data with a fitting line.

Algorithm 5.1. Establishment of the model for dynamic exposure adjustment.

Input: the intensity I_{basic} of the base image; the input images I_{input}; the range of the exposure value of the camera $[E_{min}, E_{max}]$.

Output: the exposure value E_{exp} of the camera and the mapping function f_{exp}.

Steps:

Step 1. Set the termination condition to be $| E_{exp} - E_{max} | < 1$ or $| E_{exp} - E_{min} | < 1$.

Step 2. Computer $E_{exp} = (E_{min} + E_{max}) / 2$.

Step 3. Compute the intensity I_{input} of the image and compare it with I_{basic}; if $I_{input} - I_{basic} < 0$, then set $E_{min} = E_{exp}$; else, set $E_{max} = E_{exp}$.

Step 4. Record I_{input} and E_{exp}.

Step 5. Repeat Steps 2 through 4 until the termination condition is satisfied.

Step 6. Create a mapping function $f_{exp}(x) = ax + b$, where $x = I_{input}$ and $f_{exp}(x) = E_{exp}$.

In the navigation procedure, the system adjusts the exposure value of the camera at the beginning of each cycle to deal with the varying light condition. The adjustment equation is defined in the following:

$$m = (Exp_{new} - Exp_{now}) / (Y_{base} - Y_{now}), \tag{5.6}$$

or equivalently,

$$Exp_{new} = (Y_{base} - Y_{now}) \times m + Exp_{now} \tag{5.7}$$

where m is the slope of the fitting line, Exp_{new} is the new exposure value, Exp_{now} is the current exposure value, Y_{now} is the current intensity value of the input image and Y_{base} is the intensity value I_{base} of the base image.

5.3.3 Single-class classification of HSI colors for sidewalk detection

Two methods are used to classify the curbstone features on the sidewalk in this study. First, we use the special color of the curbstone at the sidewalk to detect the guide line. As illustrated in Figure 5.4, a cylinder is used to represent the HSI color model. It translates the *RGB* space into three channels, called *hue*, *saturation* and *intensity*, forming an HSI color space with hue representing the perceived colors in 0^{o} to 360^{o}, saturation representing the brightness of the color, and intensity representing the gradient of the gray level images.

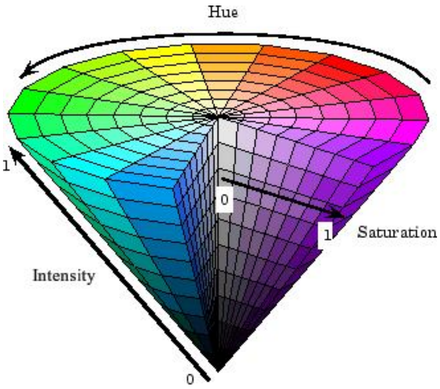


Figure 5.4 The HSI color model [23].

Method 1.

We use the hue value and the saturation value to describe the color of the curbstone as shown in Figure 5.5. The ground of the sidewalk looks unlike the red color, but as we classify the curbstone from the sidewalk using only the hue channel, the result is shown in Figure 5.5(b) which is not good. Fortunately, we found the saturations of the ground are much lower than those of the curbstone because of the different materials. Therefore, we use both hue and saturation values to extract the curbstone of the sidewalk. The classification rule for extracting the curbstone is designed to be in the following:

"if the input pattern x satisfied the following conditions:

$$thH \leqq h(x) \leqq thH_2,$$

$$thS \leqq s(x) \leqq thS_2,$$

then assign x to the class ω_c; else to the class ω_g, " (5.8)

where ω_c is the class of the curbstone, and ω_g is the class of the ground of the sidewalk; the functions h and s are used to translate the input pattern x to the hue value and saturation value respectively; thH_1 and thH_2 are the threshold values in the hue channel; and thS_1 and thS_2 are the threshold values in the saturation channel.

Method 2.

We extract the guide line using the ground features instead of the curbstone in this case. In the learning process, we use two-dimension features of the hue and saturation and design a single-class classification rule to extract the curbstone:

"if $d(X) < thre$, then assign the input pattern X to the class ω_g; else,
to the class ω_c," (5.9)

where X is the input pattern, ω_c is the class of the curbstone, ω_g is the class of the ground of the sidewalk; *thre* is a threshold value decided manually; and d is the *Mahalanobis* *distance* which is defined as follows:

$$d^2 = (X - M)^{\mathrm{T}}\Sigma^{-1}(X - M) \qquad . \quad \tilde{1}$$

where M is the *mean vector* of ω_g and Σ is the *covariance matrix* of ω_g. The resulting image of Method 2 is shown in Figure 5.7.

Method 2 is more general than Method 1 but the computation cost of it is higher. If the color of the curbstone is different, the system can still extract out the guide line by using Method 2. In order to adapt to the real environment, Method 2 is adopted in this study.

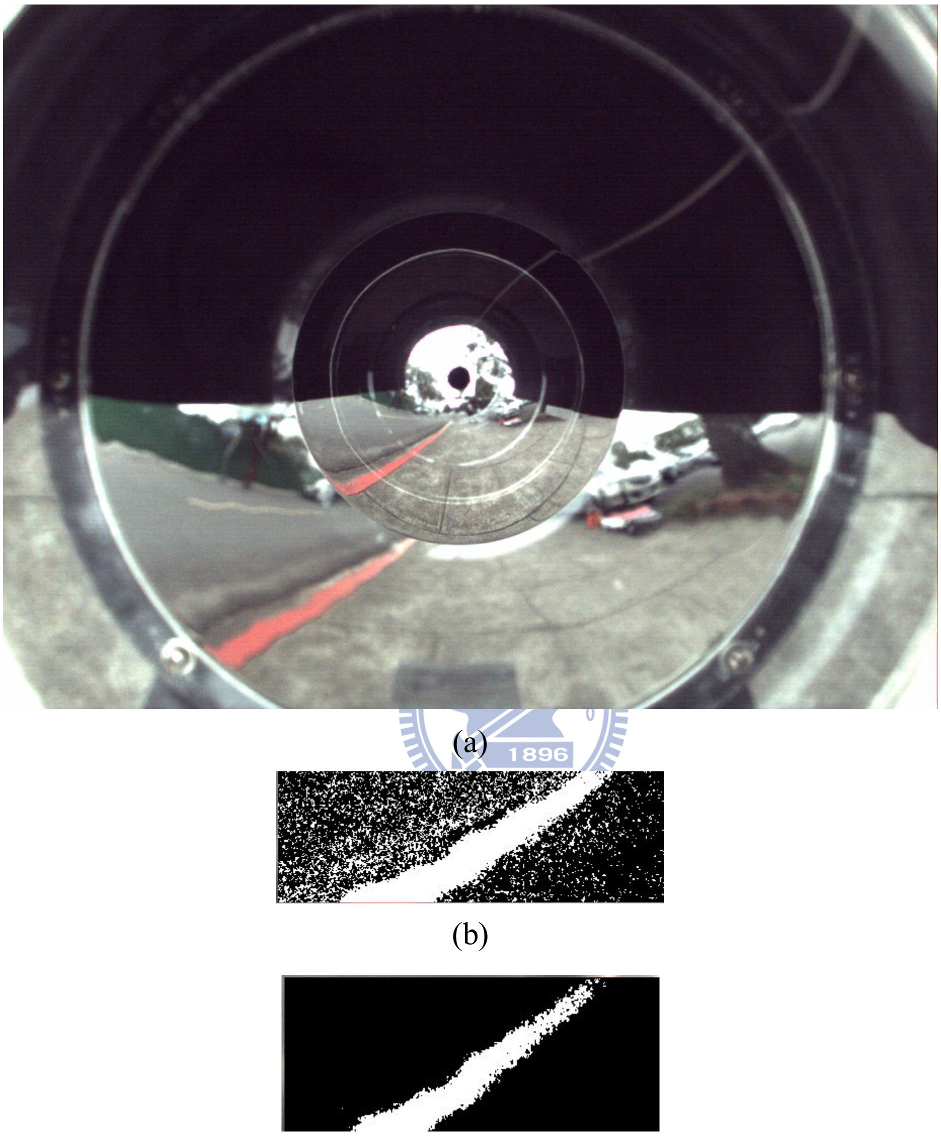
5.3.4 Proposed method for guide line detection

In this section, we will describe four steps for detecting the guide line features and finding the corresponding points in the following. We call this procedure as *guide line detection.*

1. Defining the extraction windows to detect the features.

We define two pairs of extraction windows, each pair has two extraction windows, to detect the guide line features of the curbstone, and one pair is used for the lateral direction and the other pair is used for the frontal direction. In order to increase the efficiency, the extraction windows in this study are defined as the rectangular shapes instead of them as the fan shapes.

In the lateral direction, the extraction windows are defined to cover a shape extending from 223° to 256° in the ICS. The extraction window W_{small} is located from the top-left point at coordinates (701, 713) to the bottom-right one at coordinates (755, 759) with the size of 46 pixels by 54 pixels and the other extraction window W_{big} is located from the top-left point at coordinates (463, 932) to the bottom-right point at coordinates (685, 1116) with the size of 184 pixels by 222 pixels. Their shapes appearing in the omni-image in accordance with the rotational invariance property is illustrated in Figure 5.6(a).



(a)

(b)

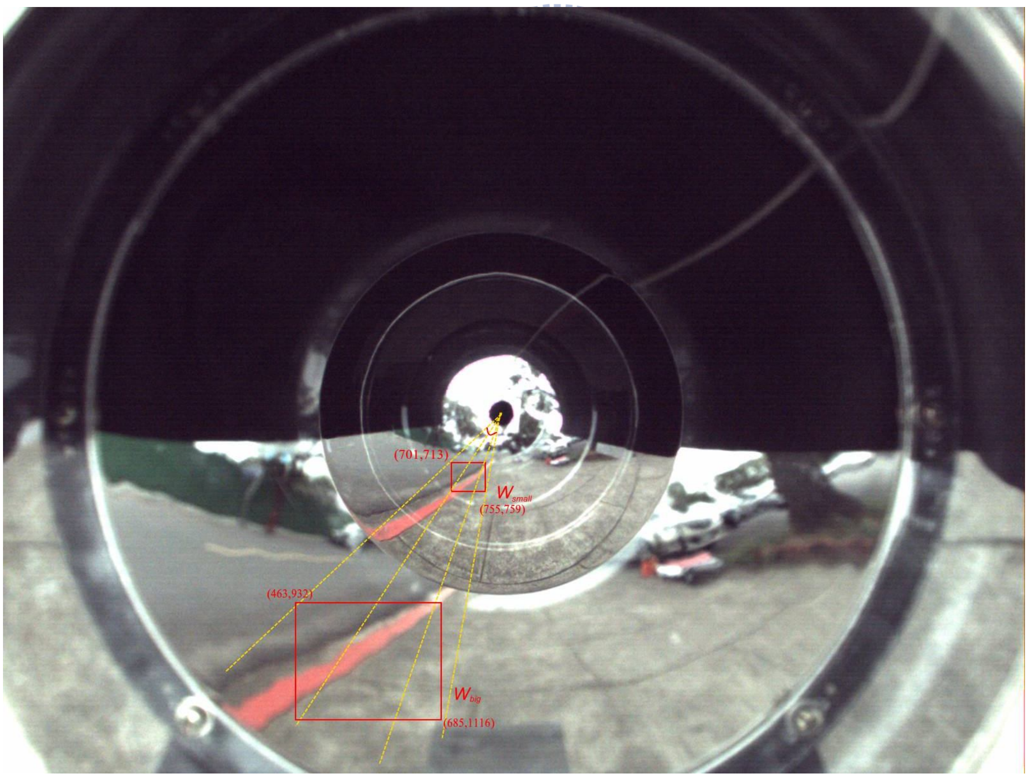
(c)

Figure 5.5 Experimental results of extracting curbstone features. (a) An input image. (b) The curbstone features extracted by the hue channel. (c) The curbstone features extracted by the hue and saturation channels.

In the frontal direction, the pair of the extraction windows is defined from 244° to 299° in the ICS. The extraction window W_{small} is located from the top-left point at coordinates (753, 730) to the bottom-right point at coordinates (811, 798) with the size of 68 pixels by 58 pixels and the other extraction window W_{big} is located from the top-left point at coordinates (692, 929) to the bottom-right point at coordinates (882, 1124) with the size of 195 pixels by 190 pixels by the rotational invariance property as illustrated in Figure 5.6(b). Each of the four corners of W_{big} and that of W_{small} are aligned in the same radial direction.

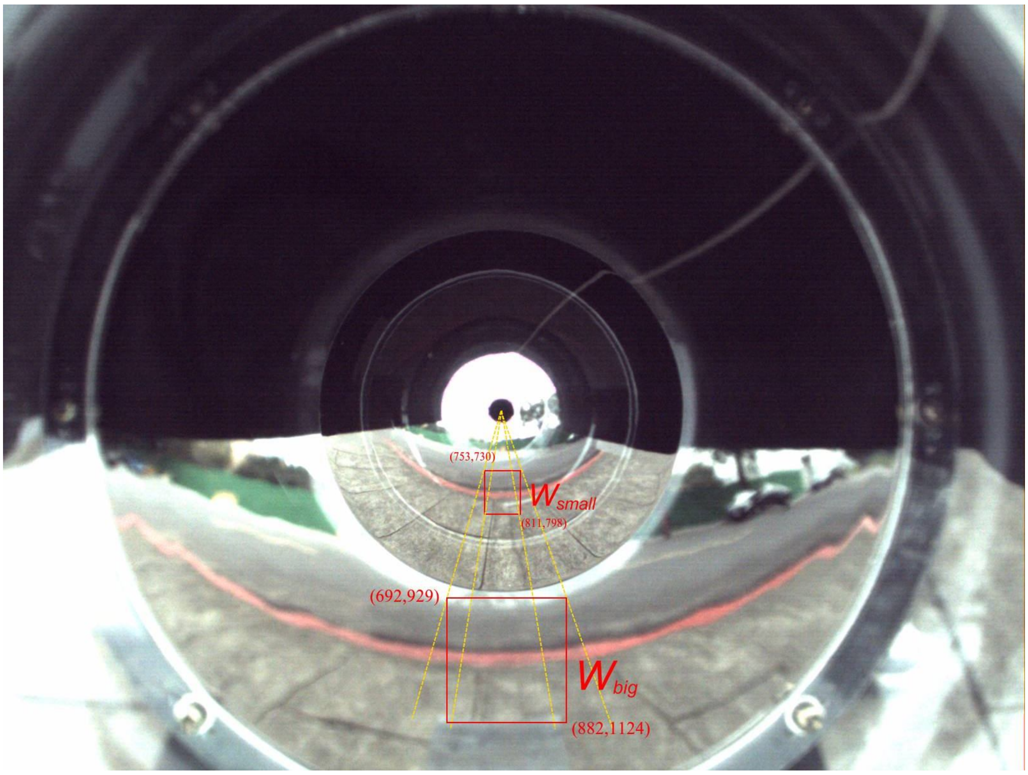
2. Detecting the features of the curbstones.

By the proposed classification rules, we can extract curbstone features well and we also use some techniques in imaging process such as *connected components*, *dilation* and *erosion* to eliminate noise in the image after detecting the features of the curbstone. The resulting image is shown in Figure 5.7.



(a)

Figure 5.6 Two extraction windows used to detect guide line. (a) A extraction window used for detection of lateral direction. (b) A extraction window used for detection of frontal direction.



(b)

Figure 5.7 Two extraction windows used to detect guide line. (a) A extraction window used for detection of lateral direction. (b) A extraction window used for detection of frontal direction. (cont'd)

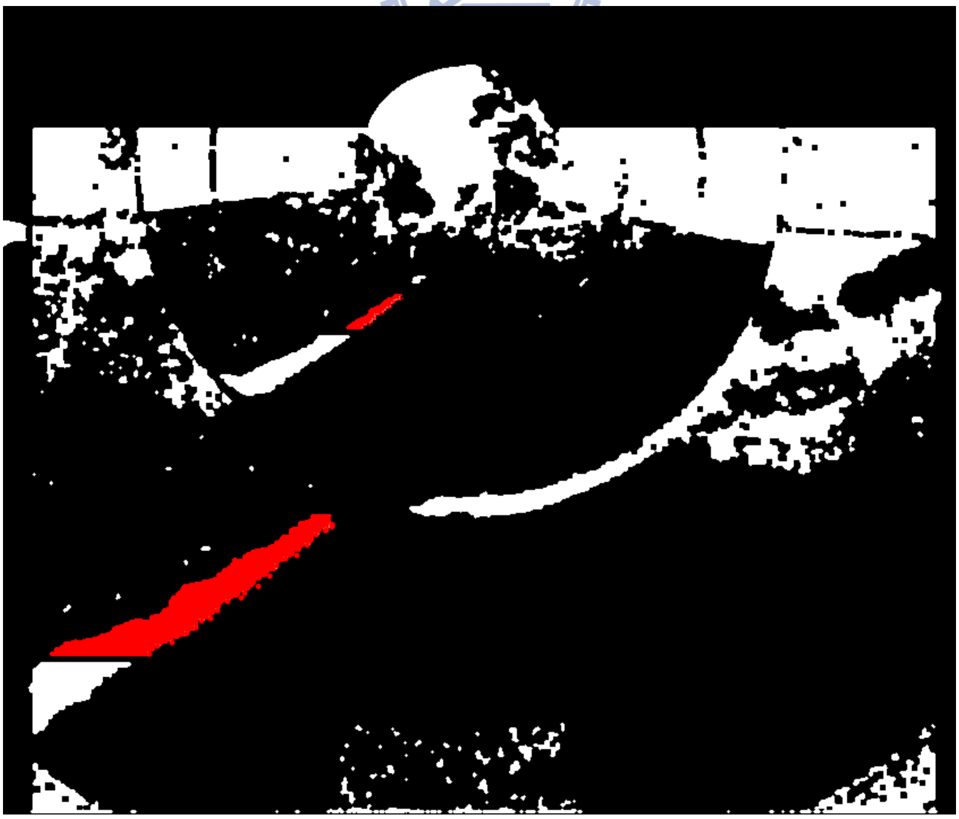


Figure 5.8 The result of detecting the curbstone features in the experimental environment.

3. **Extracting the guide line of the curbstones.**

Many curbstone features are hard to use to detect the corresponding points in W_{big}, so we propose a method to detect the guide line of the curbstone instead of total features. After detecting the curbstone features, we scan the image in W_{small} from right to left and from top to bottom to find the guide line at the inner position of the sidewalk. After detecting the first feature pixel with the scan order from right to left, label the pixel as a guide line feature point and then detect the next feature pixel in the next row. An illustration is shown in Figure 5.8.

4. **Finding the corresponding points.**

In [10] and [23], an omni-image is transformed into two panoramic images for two mirrors, then the *bilinear interpolation technique* is used to refine the panoramic images, and corresponding points are found by some matching algorithms, and range data are computed finally. A disadvantage is that the computation cost of this process is high.

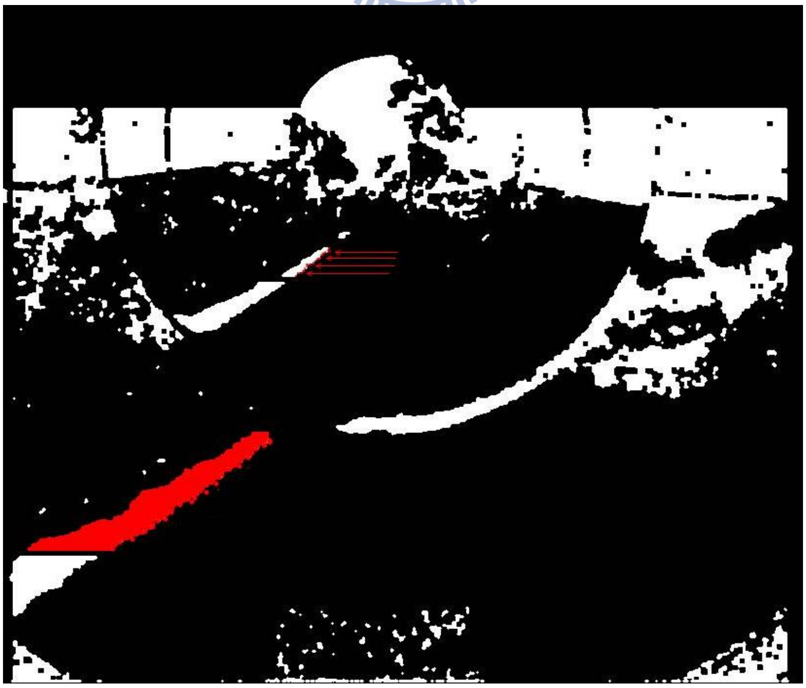


Figure 5.9 The guide line (the red dots) extracted by the proposed method.

We propose in this study a method based on the rotational invariance property of the omni-image without transforming the omni-image into two panoramic images, as illustrated in Figure 5.9. The corresponding point is in the same radial direction of the feature point, so we scan the feature points along the radial direction (the red dotted line in Figure 5.9(a)) and take the corresponding point to be the *last* feature pixel in the image portion corresponding to the large mirror, as illustrated by Figure 5.9(a).

Algorithm 5.3. Computation of corresponding feature points.

Input: an image I_{input}.

Output: an image with corresponding points marked.

Steps:

Step 1. Scan I_{input} in W_{small} and W_{big} to extract the curbstone features based on the proposed classification rules.

Step 2. Scan in W_{small} from right to left and from top to bottom to find the guide line at the inner sidewalk. If the first curbstone feature on a row is found, label the pixel as a guide line feature F_g, and then scan the next row until finished.

Step 3. For each F_g, compute the azimuth angle θ, and scan I_{input} accordingly on the same radial direction of F_g from near to far with respect to the center in the area of W_{big}.

Step 4. Label as a corresponding point of F_g the last feature pixel found in the from-near-to-far scanning with respect to the center of the omni-image.

5.3.5 Line fitting technique for sidewalk following

After detecting the guide line features and their corresponding points, we can compute the range data by Equations (3.15) and (3.16). And in this section, we use the line fitting technique described by Equations (5.3) through (5.5) to refine the range data. We also use the technique to compute the directions of the guide line. The proposed schemes are illustrated in Figure 5.10, and the procedure is listed in the following.

Algorithm 5.4. Refining the range data by the line fitting technique.

Input: a set of range data, R.

Output: a refined set of range data R'.

Steps:

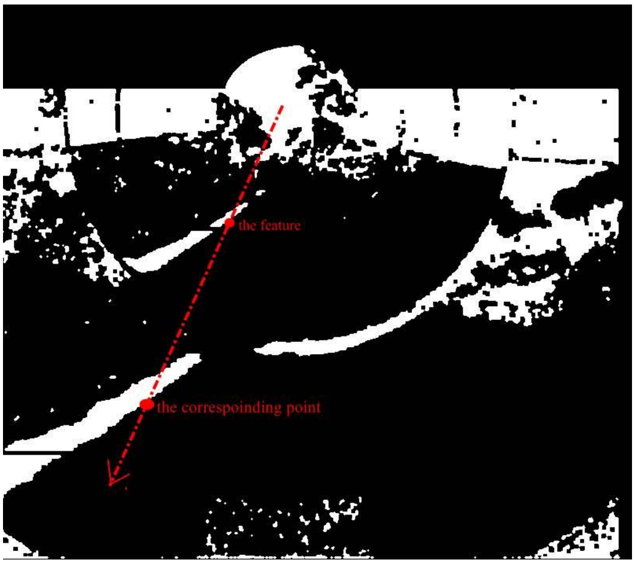
Step 1. Use a line equation L to fit all the range data by Equations (5.3) through (5.5).

Step 2. Calculate the distance d_i from each range data item r_i to L.

Step 3. Compute a threshold value d_{mean} which is the average value of all computed distances d_i.

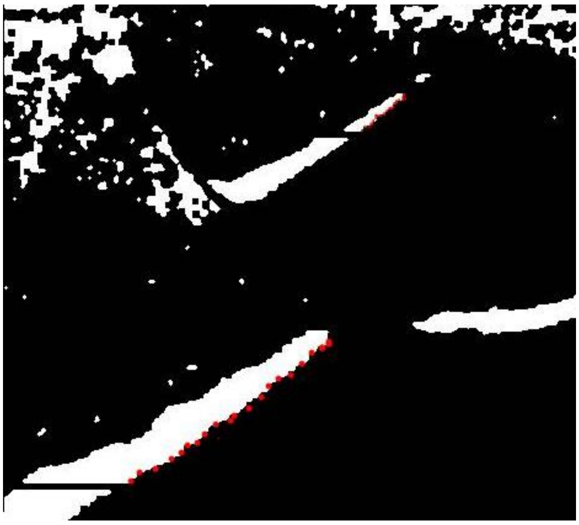
Step 4. Collect the range data item r_i into R' if the corresponding distance $d_i < d_{mean}$.

Step 5. Take the final R' as the output.



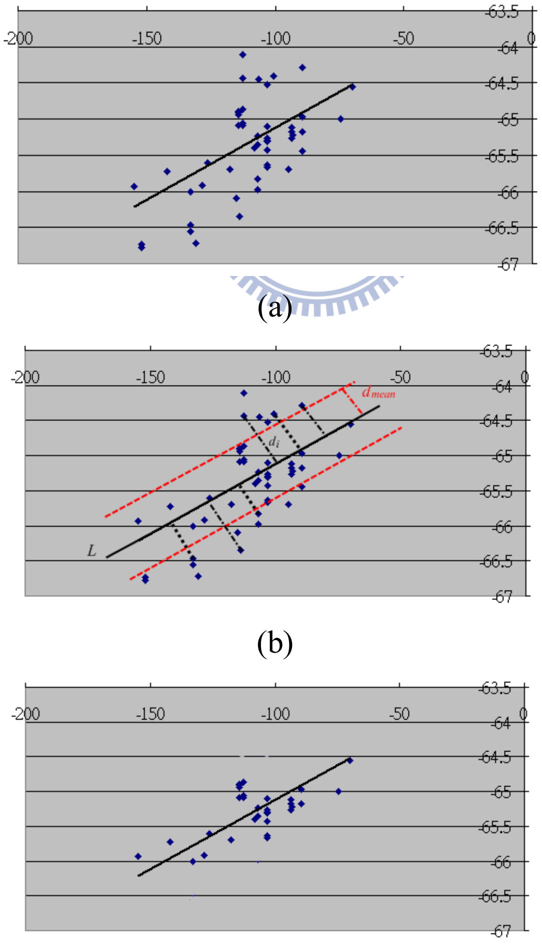
(a)

Figure 5.10 Finding the corresponding point for the feature point. (a) The scanning order. (b) The experimental results.



(b)

Figure 5.11 Finding the corresponding point for the feature point. (a) The scanning order. (b) The experimental results. (cont’d)



(a)

(b)

(c)

Figure 5.12 An illustration of the proposed scheme to refine the range data. (a) Fitting the range data. (b) Computing the distance from each data to the line. (c) Refined result.

After refining the range data, we obtain the turning angle γ of the vehicle and the distance d from the fitting line to the mirror center by

$$\gamma = \tan^{-1}(1/m), \tag{5.11}$$

$$d = \frac{|mx_0 + b - y_0|}{\sqrt{1+m^2}}, \tag{5.12}$$

where m is the slope of L and d is the distance from the mirror center (x_0, y_0) to L. And we use γ and d to instruct the vehicle to perform line-following actions (described later).

5.3.6 Proposed method for semi-automatic navigation

In the navigation procedure, we adopt a strategy to instruct the vehicle to follow a safer navigation route as illustrated in Figure 5.11. If the vehicle navigates in any of the damage zones as shown in the figure, its speed will be slowed down (and so to increase the cycle time of the navigation procedure).

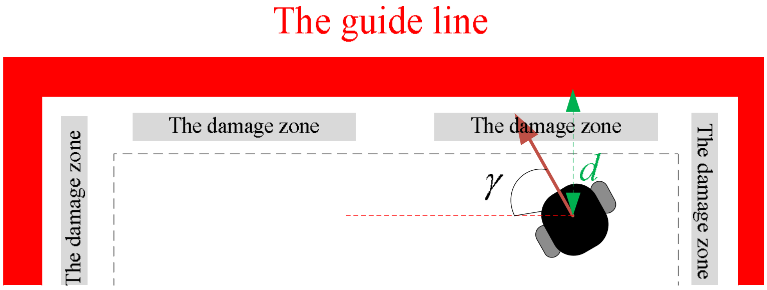


Figure 5.13 The dangerous (damage) zones along the guide line.

Algorithm 5.5. Process of semi-automatic navigation.

Input: current image I_{input} and the mapping function f_{exp} for exposure correction.

Output: the vehicle commands with γ and d.

Steps:

Step 1. Compute the average intensity of I_{input}.

Step 2. Adjust the exposure of the camera automatically by f_{exp} described in Section 5.3.2.

Step 3. Extract the curbstone features by the proposed techniques described in Section 5.3.3.

Step 4. Extract the guide line feature points by the techniques described in Section 5.3.4.

Step 5. Find the corresponding points of each guide line feature point.

Step 6. Calculate the 3D range data of the guide line by the technique described in Section 3.2.3.

Step 7. Refine the range data R' using the line fitting technique by Algorithm 5.4.

Step 8. Conduct line fitting for R' again and get the line equation L by Algorithm 5.4.

Step 9. Compute γ and d of L by Equations (5.11) and (5.12).

Step 10. Slow down the speed of the vehicle if it is in the damage zone detected by the use of d.

Step 11. Perform the turning command by the use of γ.

5.4 Detection of Hand Poses as Guidance Commands

5.4.1 Idea of proposed method for hand pose detection

In the previous section, we described the proposed scheme to extract the guide line feature points in the outdoor environment. If there exists no special curbstone feature point in the environment or if the extracted curbstone feature points are similar to the feature points of the ground, then it will become hard to extract out the guide line feature points. Therefore, we propose a *human interaction technique* using hand poses to solve this problem. The hand poses are detected by the technique as commands which are defined in advance. Then, the system will follow the commands to instruct the vehicle to navigate in the environment. Once a human hand pose is detected, the system terminates the procedure of semi-automatic navigation and starts up a procedure of *blind navigation* to let the user guide the vehicle by different hand poses without the assistance of the visual device (the camera).

Based on the use of different hand positions, we define four regions on the camera enclosure with five different commands in Table 5.1 and show the illustration in Figure 5.12. A counter is used to distinguish the different commands for the cases when the human hand is detected in the UP region, as shown in Table 5.2.

The hand pose detection procedure is divided into two parts in the following and it will be described in detail in the subsequent section:

1. Detection of human skins by the YC_bC_r color model.
2. Ellipse fitting for the human hand gestures.

5.4.2 Use of YC_bC_r colors for hand pose detection

The YC_bC_r color space is often used as features to detect human skins owing to its good properties for this purpose. Figure 5.13 shows how the YC_bC_r color space model is formed with Y representing the *luminance* component and C_b and C_r representing the *blue-difference* and *red-difference* components, respectively.

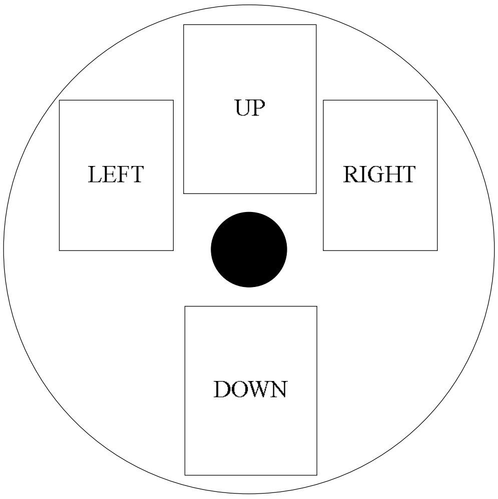


Figure 5.14 Four detection regions with different vehicle commands.

Table 5.1 The vehicle commands table.

Detecting region	The vehicle commands
UP	terminate the guide line detection procedure, stop the vehicle or go forwardly
LEFT	turn left
RIGHT	turn right
DOWN	terminate the human guidance procedure and restart the guide line detection procedure
LEFT and RIGHT	terminate the learning procedure and save the path nodes

Table 5.2 The definition of the counter used for different commands.

The counter C	The commands
$C=1$	Stop the vehicle and terminate the guide line detection procedure
C is even and bigger than 2	Stop the vehicle
C is odd and bigger than 2	Go forward

The proposed process of human skin detection is described in the following:

1. Detect a rough sketch of the human hand using the luminance component by

$$Y_i \leqq threY, \tag{5.13}$$

where Y_i is the luminance value of a pixel and $threY$ is a threshold value. An example of the results is shown in Figure 5.14(b).

2. Detect hand skin using the blue-difference and red-difference components by

$$threC_{b1} < C_{bi} < threC_{b2},$$

$$threC_{r1} < C_{ri} < threC_{r2}, \tag{5.14}$$

where C_{bi} is the C_b value of each pixel and $threC_{b1}$ and $threC_{b2}$ are two threshold values of the upper bound and the lower bound, respectively; C_{ri} is the C_r value of each pixel and $threC_{r1}$ and $threC_{r2}$ are two threshold values of the upper bound and the lower bound, respectively. The resulting image is created by the intersection of C_{bi} and C_{ri}. An example is shown in Figure 5.15.

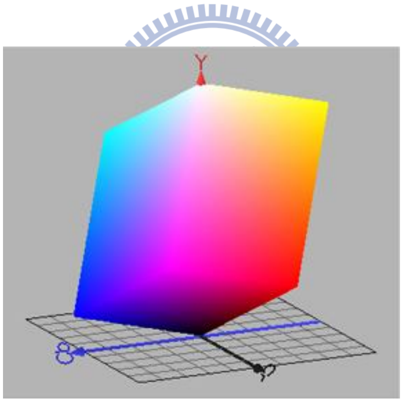
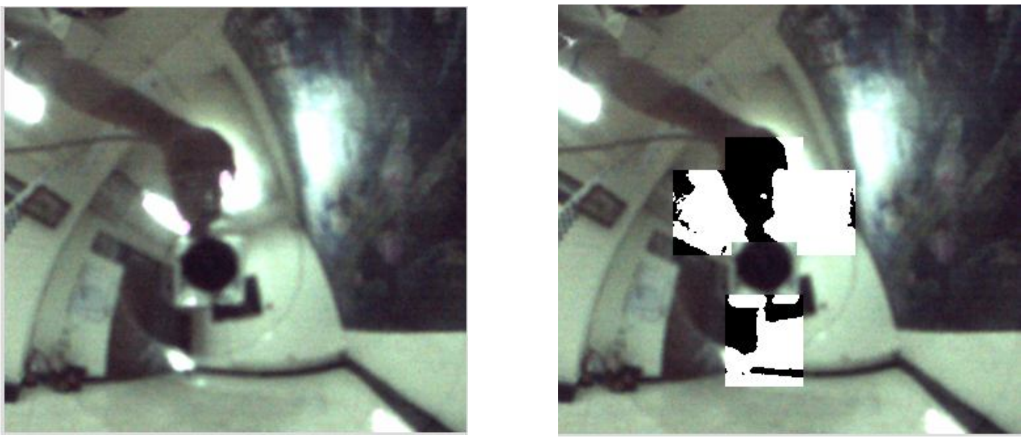
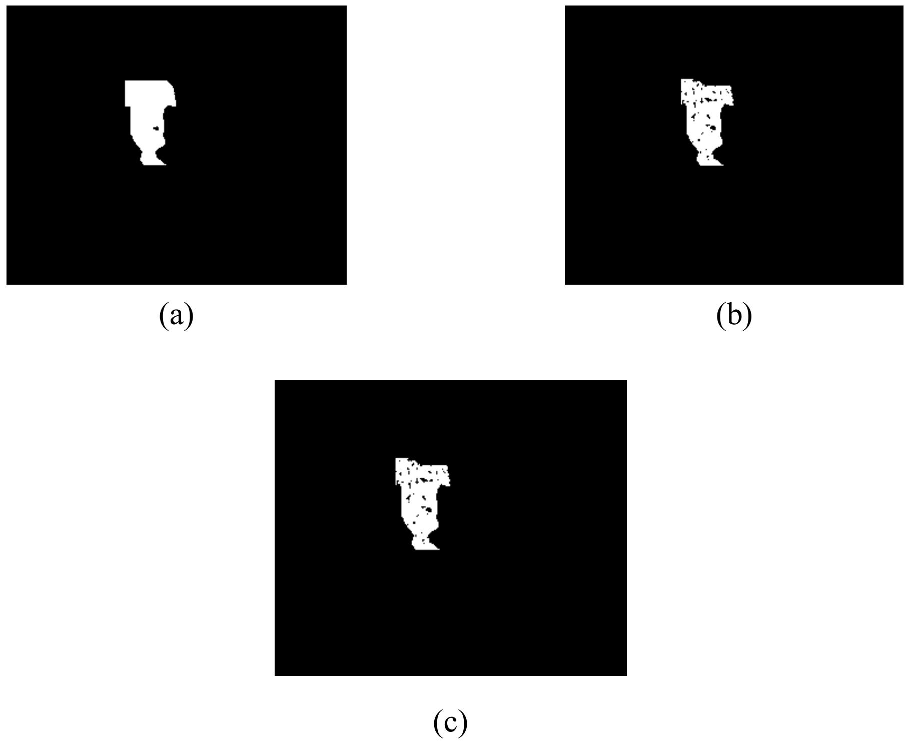


Figure 5.15 The YC_bC_r color space model [25].



(a) (b)

Figure 5.16 A rough sketch of the human hand using luminance component.



(a) (b)

(c)

Figure 5.17 The detection of the human hand using blue-difference and red-difference components. (a) The result of the C_b component. (b) The result of the C_r component. (c) The intersection of the C_b component and the C_r component.

5.4.3 Proposed hand shape fitting technique for hand pose detection

Although we can use the proposed technique described in the last section to extract the human hands from an omni-image using the YC_bC_r color model, noise may be detected because of noise features may be similar to those of the human hand. Therefore in this section, we will introduce the proposed *ellipse fitting technique* to correct the hand pose detection results. In advance, we assume that the allowed hand gesture is of only one kind in the study, the five fingers are in a “gathering” condition, and the fingers are pointing to the ground, such as the one shown in Figure 5.14(a). Two strategies are adopted in the proposed technique of hand shape fitting by elliptical shapes.

Strategy 1:

In the ellipse equation, $\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$, the two unknown variables *a* and *b* are the horizontal and the vertical axis length. We can distinguish different objects by different values of *a* and *b*. The two unknown variables *a* and *b* can be computed by the following algorithm.

Algorithm 5.6. Detection of the human hand shape and elliptical shape fitting.

Input: a set *S* of possible human hand feature points.

Output: a signal indicating the classification result of *S* as a human hand or noise.

Steps:

Step 1. Compute the center (u_c, v_c) of the possible hand feature points (u_i, v_i) of *S* in the ICS by

$$(u_c, v_c) = \frac{1}{n}\left(\sum_{i=1}^{n} u_i, \sum_{i=1}^{n} v_i\right), \tag{5.15}$$

where *n* is the number of the hand feature points.

Step 2. Take (u_c, v_c) to be the coordinates of the origin of the ellipse, and normalize the coordinates (u_i, v_i) of all feature points with respect to (u_c, v_c) to be (x_i, y_i) by

$$(x_i, y_i) = (u-u_c, v-v_c). \tag{5.16}$$

Step 3. Compute *a* and *b* of the ellipse according to the geometry of the ellipse shape by [26]:

$$a = \sqrt{\frac{\left(\sum_{i=1}^{n} x_i^2 y_i^2\right)^2 - \sum_{i=1}^{n} x_i^4 \sum_{i=1}^{n} y_i^4}{\sum_{i=1}^{n} x_i^2 y_i^2 \sum_{i=1}^{n} y_i^2 - \sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^4}},$$

$$b = \sqrt{\frac{(\sum_{i=1}^{n} x_i^2 y_i^2)^2 - \sum_{i=1}^{n} x_i^4 \sum_{i=1}^{n} y_i^4}{\sum_{i=1}^{n} x_i^2 y_i^2 \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i^4 \sum_{i=1}^{n} y_i^2}}, \tag{5.17}$$

where n is the hand feature items.

Step 4. Classify S as a human hand, if $(b-a) \geqq thre$, and then go to Algorithm 5.7 below; otherwise, regard S as noise.

Strategy 2:

To extract the human-hand shape, just using the ellipse fitting technique is not enough. Figure 5.16(a) shows a case that the noise is assigned as a human hand because the ellipse shape of the noise is similar to that of a human hand. Figure 5.16(b) shows the ellipse fitting result with correct human hand detection. Comparing the two pictures, the feature points of the correct human hand are more compact and neighboring, but those of the noise are separate. Therefore, we propose a second strategy for human hand detection as described in the following algorithm:

Algorithm 5.7. Computation of the compactness of the human hand feature points.

Input: a set S of possible human hand feature points.

Output: a signal indicating the classification result of S as a human hand or noise.

Steps:

Step 1. Compute $d_i[(u_i, v_i), (u_c, v_c)]$ for each hand feature point in S with coordinates (u_i, v_i), where the function $d(a, b)$ is computed as the Euclidean distance between a and b.

Step 2. Compute the average distance d_{mean} by

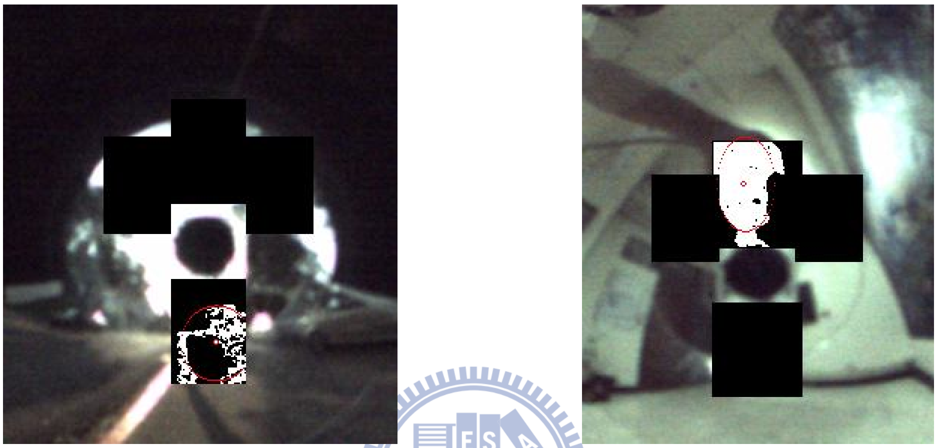
$$d_{mean} = \sum_{i=1}^{n} d_i, \tag{5.18}$$

where n is the number of feature points in S.

Step 3. Refine the classification of S by the following rule:

"If $thre_1 < d_{mean} < thre_2$, regard S as a human hand; else, as noise," (5.19)

where $thre_1$ and $thre_2$ are two pre-selected threshold values decided in the experiment.



(a) (b)

Figure 5.18 The result image of the human hand detection. (a) The noises are detected as the human hands. (b) The correct detection.

5.4.4 Proposed dynamically thresholding adjustment

Differing from dynamic exposure adjustment mentioned in Section 5.3.2, the proposed dynamic thresholding adjustment scheme mentioned previously adjusts the threshold value of luminance, $threY$, in Equation (5.13). In the study, we found that the threshold value $threY$ is positive relatively to the luminance of the image. Therefore, we can adjust $threY$ by a linear scaling technique. Similar to the dynamically exposure adjustment we use, we save the luminance value of the base image Y_{base} and the luminance threshold value $threY_{base}$ in advance and the process of dynamically thresholding adjustment is described in Algorithm 5.8. To adjust the threshold value automatically, the process is started up at the beginning of the procedure of the human hands detection in each cycle time.

Algorithm 5.8. Dynamic thresholding adjustment.

Input: the input image I_{input}, the luminance of the base image, Y_{base}, and the threshold value $threY_{base}$.

Output: the new threshold value $threY'$.

Steps:

Step 1. Compute the average luminance Y_{input} of I_{input}.

Step 2. Compute the new threshold value in the range $[Y_{input} - Y_{base}]$ by

$$threY' = threY + [(Y_{input} - Y_{base})/Y_{base}] \times threY. \tag{5.20}$$

5.4.5 Proposed hand pose detection process

In this section, we summarize different techniques for hand pose detection and describe them in the following scheme.

Algorithm 5.9. The hand pose detection process

Input: an image I_{input}.

Output: commands to instruct the vehicle to navigate.

Steps:

Step 1. Execute the procedure of dynamic thresholding adjustment by Algorithm 5.8.

Step 2. Detect possible human hand feature points in the pre-defined region in I_{input} mentioned in Section 5.4.2.

Step 3. Execute the ellipse fitting technique to decide whether the possible human hand feature points compose a human hand by Algorithms 5.6 and 5.7.

Step 4. If a human hand is detected, then use Table 5.1 to decide a command to instruct the vehicle to navigate.

5.5 Proposed Path Planning Method Using Learned Data

5.5.1 Idea of path planning

In the learning procedure, a semi-automatic process and human hand pose detection are used to let the autonomous vehicle learn the information in the environment. Figure 5.17 shows an example which is a map created in the experimental environment and the blue points are the path nodes recorded by odometer of the vehicle, and the purple line points and the red line points are the guide lines detected by the two-mirror omni-camera at different directions. In this study, we call the blue point as the *general point* in the procedure of path planning. The autonomous vehicle can navigate well with many path nodes, but the system will spend much time to correct its position as the vehicle arrives at a path node. Therefore, we design a strategy to decrease the numbers of the recorded path nodes

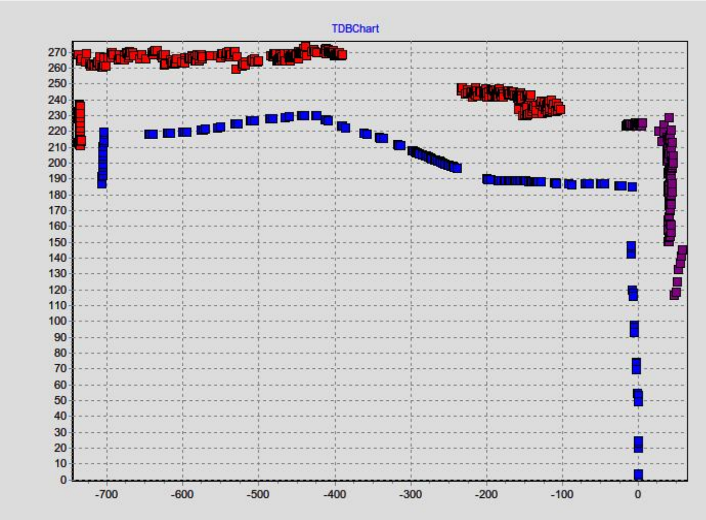


Figure 5.19 A map created in the experimental environment.

5.5.2 Proposed path planning process

Five types of the path nodes are defined in this study as shown in Table 5.3, and these types of nodes are recorded in the learning procedure automatically. In the procedure of guide line detection, if the vehicle detects the guide line to be in front of it, the system will record the current position as a turning point and put it into a set S. In the procedure of human hand pose detection, if the system terminates the procedure of guide line detection and execute the procedure of blind navigation, the system will record the current position as a blind navigation point and put it into the set S.

The mechanical error of the autonomous vehicle will be increased if the interval between two path nodes is too long. Therefore, we insert some points in the interval in the set S to increase the number of times for correcting the location of the vehicle. A new set T of the path nodes after executing the path planning process is obtained. Figure 5.18 shows a map with T of the path nodes after the path planning process.

Table 5.3 Different type of the path nodes.

Type of points	Colors of points on the map
General point	The blue points in Figure 5.17
Start point / End point	The red points
Turning point	The green points
Blind navigation point	The black points
Inserted point	The yellow points

Algorithm 5.11. Path planning.
Input: a set S of path nodes learned in the learning procedure.
Output: a set T of left path nodes after path planning.
Steps:

Step 1. Scan each node N_i in S. If N_i is not a general point, then mark N_i as an extreme point N_1^e.

Step 2. Repeat Step 1, and if a second extreme point N_2^e is found, then create an interval P_i, where N_1^e is the beginning point and N_2^e is the end point of P_i.

Step 3. Scan each node n_i in P_i, and if $d(n_i, N_1^e) \geqq thre$, then mark n_i as an inserted point, where the function value $d(a, b)$ is computed as the Euclidean distance between a and b and *thre* is a pre-selected threshold value decided in the experiment.

Step 4. Repeat Steps 1 through 3 if N_i is not the end of S; else, scan each node N_i in S and if N_i is not a general point, put it into the output set T.

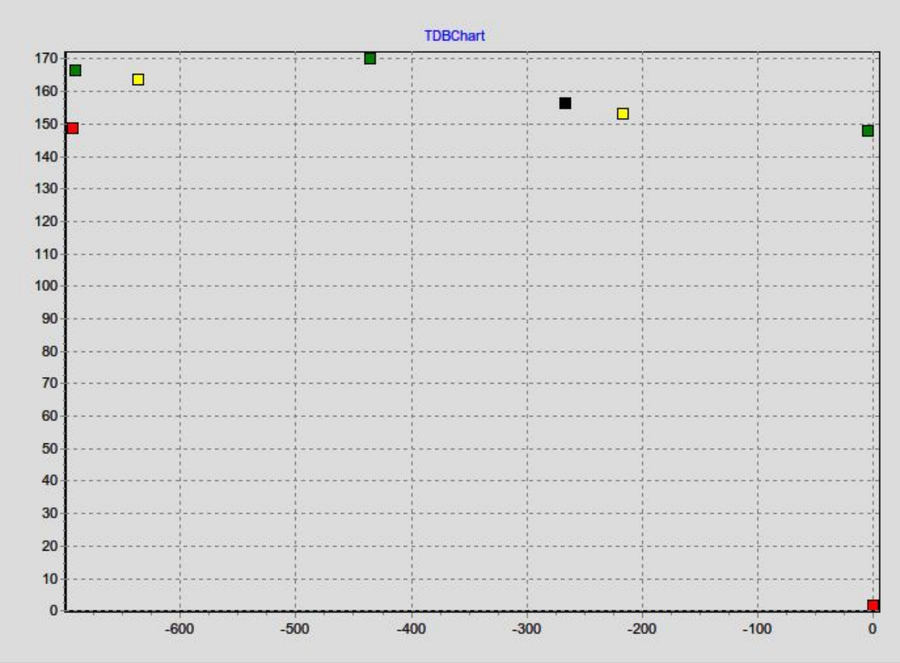


Figure 5.20 A map is created with the path planning process.

Chapter 6
Vehicle Guidance on Sidewalks by Curb Following

6.1 Idea of Proposed Guidance Method

In this study, the autonomous vehicle is designed to guide the blind people to walk safely on the sidewalk environment. The first difficulty is whether the vehicle knows the existence of the user who follows the autonomous vehicle. We propose a technique using ultrasonic sensors to synchronize the speed of the autonomous vehicle with that of the blind.

Another difficulty is that some dynamic obstacles may block the navigation path of the autonomous vehicle. We propose a method based on Chiang and Tsai's method [21] to avoid the dynamic obstacles by using the proposed two-mirror omni-camera. An illustration is shown in Figure 6.1 and the proposed schemes are described in the subsequent sections.

6.1.1 Proposed synchronization method of vehicle navigation and human walking speeds

In the proposed method for synchronization of vehicle navigation and human walking speeds, the aft sonar arrays are used only to detect foots of the user and they are no.9 through no.14 sonar sensors in Figure 6.2. We define four regions: near, middle, far and too far, for synchronizing the vehicle speed with that of the user, respectively. An illustration is shown in Figure 6.4 and their definitions are given in Table 6.1. An algorithm of the proposed method is described in the following.

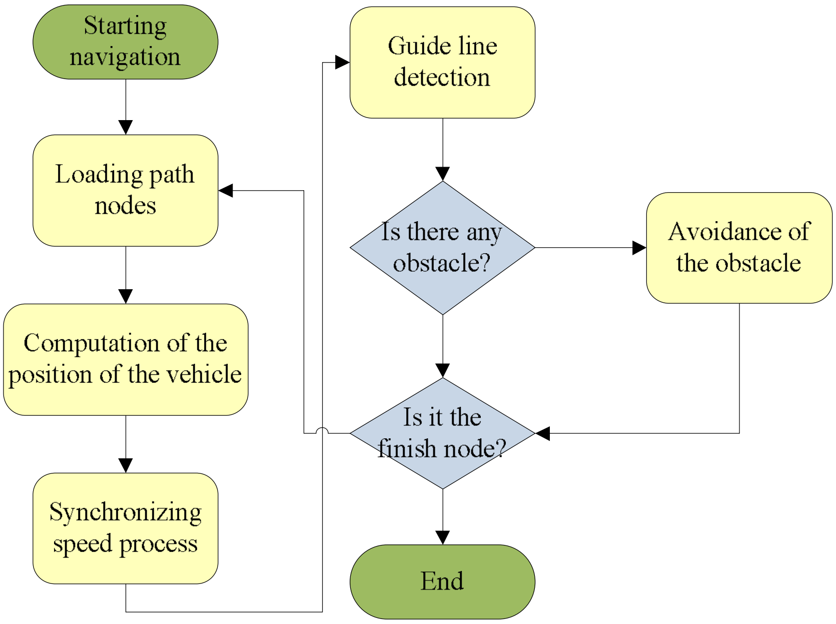
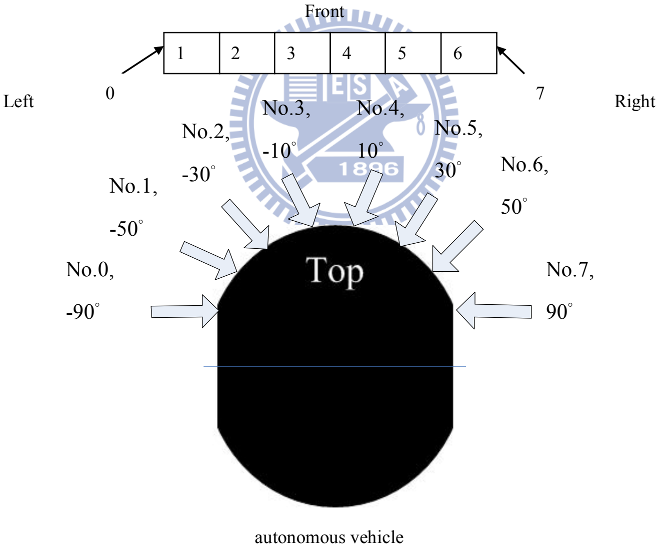


Figure 6.1 An illustration of the navigation procedure.



(a)

Figure 6.2 The illustration of sonar arrays. (a) The fore sonar array of the vehicle. (b) The fore and aft sonar arrays [19].

Algorithm 6.1. Synchronization between the vehicle speed and that of the user.

Input: none.

Output: the vehicle speed.

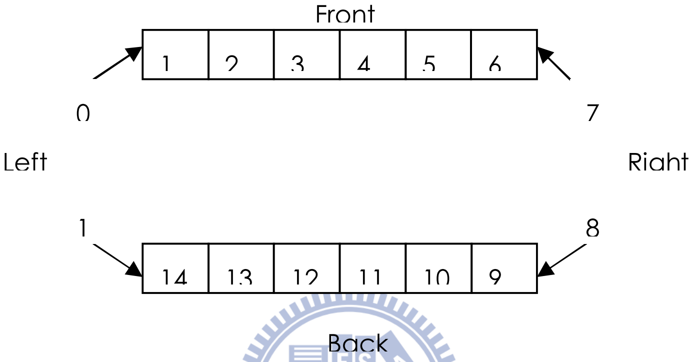
Steps:

Step 1. Compute the signal for each of the no. 9 through no. 14 sonar sensors by following rule:

> “if the sonar signal is in the detectable range, then label the sonar signal as *detected*, and record the value; else, label it as *un-detected*.”

Step 2. Compute the average value *d* of the signals which are labeled as detected.

Step 3. Decide the corresponding command by *d* according to Table 6.1.



(b)

Figure 6.3 The illustration of sonar arrays. (a) The fore sonar array of the vehicle. (b) The fore and aft sonar arrays [19] (cont’d).

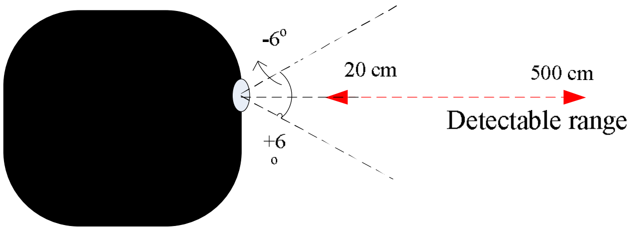


Figure 6.4 The range of a sonar sensor in detecting obstacles.

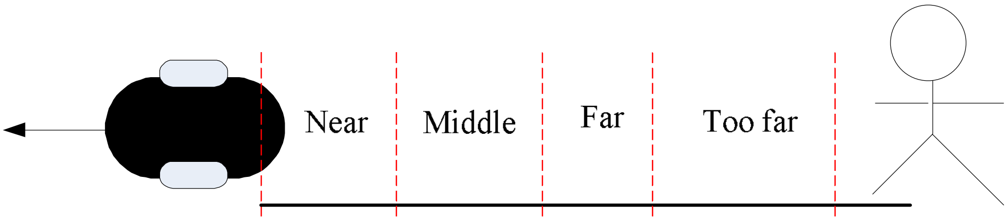


Figure 6.5 An illustration of the defined regions used for synchronizing the vehicle speed with that of the user.

Table 6.1 Different commands of the vehicle according to the sonar signal.

The region for the sonar signal detects	The vehicle commands
Near	Accelerate
Middle	Adjust the speed to be identical to the default speed which is already set in the initial
Far	Slow down the speed
Too far	Stop the vehicle

Algorithm 6.1. Synchronization between the vehicle speed and that of the user.

Input: none.

Output: the vehicle speed.

Steps:

Step 4. Compute the signal for each of the no. 9 through no. 14 sonar sensors by following rule:

> "if the sonar signal is in the detectable range, then label the sonar signal as *detected*, and record the value; else, label it as *un-detected*."

Step 5. Compute the average value *d* of the signals which are labeled as detected.

Step 6. Decide the corresponding command by *d* according to Table 6.1.

6.2 Proposed Obstacle Detection and Avoidance Process

6.2.1 Proposed method for computation of the vehicle position

In [15-18], the real world coordinates of a certain spot are computed in advance, and then used to decide a vehicle command, called goFront, to instruct the vehicle to move to different positions. Using this command provided by the MobileRobot to instruct the vehicle will lock all processes until the vehicle arrives the destination, as found in this study.

In order to use the sonar sensors to compute the distance between the vehicle and the user, we cannot use this vehicle command to instruct the vehicle. Therefore we propose an algorithm to compute the vehicle position and command accordingly the vehicle to navigate in the map.

We use a projection vector to compute the vehicle position and decide whether the vehicle has arrived at a path node or not. As illustrated in Figure 6.5, there are two path nodes, N_i and N_{i+1}, which are indicated as ith and (i+1)th ones of the path nodes. The vehicle is at the position P_i in Figure 6.5(a), and it moves to the position P_{i+1} after a time interval, as shown in Figure 6.5(b). An algorithm proposed to compute the vehicle position is described below.

Algorithm 6.2. Computation of the vehicle position.

Input: the path nodes learned in the learning procedure.

Output: none

Steps:

Step 1. Compute the length L_v of a direction vector $\overrightarrow{V_i}$ which is equal to $N_{i+1} - N_i$.

Step 2. Get the current position P_i of the vehicle from the odometer.

Step 3. Compute a direction vector $\overrightarrow{W_i}$ which is equal to $P_i - N_i$.

Step 4. Compute the length L_{wv} of a projection vector which is from $\overrightarrow{W_i}$ onto $\overrightarrow{V_i}$: if $L_{wv} \geqq L_v$, then regard the position as the one that the vehicle arrive at, i.e., the destination N_{i+1}; else, go to Step 2.

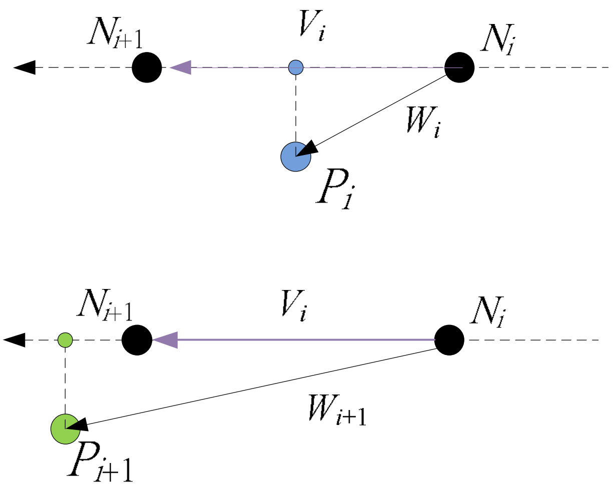


Figure 6.6 An illustration of the proposed algorithm. (a) The vehicle is at P_i. (b) The vehicle is at P_{i+1}.

6.2.2 Detection of obstacles

In this section, we will introduce the proposed scheme for detection of obstacles. Two extraction windows with rectangle shapes are defined and used to detect obstacles in front of the vehicle. The pair of the two windows is defined from 143° to 285° in the ICS. A window W_{small} is located from the top-left point at coordinates (727, 735) to the bottom-right point at coordinates (822, 810) with the size 95 pixels by 75 pixels. Another window W_{big} is located from the top-left point at coordinates (633, 936) to the bottom-right point at coordinates (903, 1113) with the size 270 pixels by 177 pixels, as shown in Figure 6.6.

Similar to the procedure described in Chapter 5, three stages are used to compute the 3D range data of obstacles.

1. **Detection of obstacle features.**

An obstacle is extracted by the use of Equation (5.8) using the HSI color model and some image processing techniques such as connected component labeling, dilation, and erosion. An example of the resulting image of detected obstacles is shown in Figure 6.7.

2. Detection of the outlines of obstacles.

After detecting the obstacle, we detect the outlines of the obstacles appearing in image portions corresponding to the big mirror and the small mirror, respectively. We scan the extraction window from top to bottom and from left to right to detect the outline feature points on the top of the obstacle, and from bottom to top and left to right to detect the outline feature points on the bottom of the obstacle. An example is shown in Figure 6.8.

3. Compute the 3D range data of the obstacle.

After we get the outline feature points of the obstacle, we can find the correspondences of the feature point pairs by Algorithm 5.3, and compute the range data by Equation (3.16). And then two portions of the range data are computed: one is of the outline feature points on the top of the obstacle, and the other is that of those on the bottom of the obstacle. We calculate the average values of the range data for the two portions with respect to the three axes in the CCS , respectively and express the them in the vector form $[d_x^{\mathrm{T}}, d_y^{\mathrm{T}}, d_z^{\mathrm{T}}]^{\mathrm{T}}$ and $[d_x^{\mathrm{B}}, d_y^{\mathrm{B}}, d_z^{\mathrm{B}}]^{\mathrm{T}}$ for the top portion and the bottom portion, respectively. The height distance H and the depth distance D and the width distance W of the obstacle can be obtained by

$$[H, D]^{\mathrm{T}} = [d_y^{\mathrm{T}}, d_z^{\mathrm{T}}]^{\mathrm{T}} - [d_y^{\mathrm{B}}, d_z^{\mathrm{B}}]^{\mathrm{T}}, \tag{6.1}$$

$$W = max(d_x^{\mathrm{T}}, d_x^{\mathrm{B}}) \tag{6.2}$$

where the function value of $max(a, b)$ is computed for the bigger one between a and b.

Finally we can use the 3D information to perform different collision avoidance actions. For example, if an obstacle is not flat and blocks the navigation path, then the vehicle will avoid it; else, the vehicle will keep going.

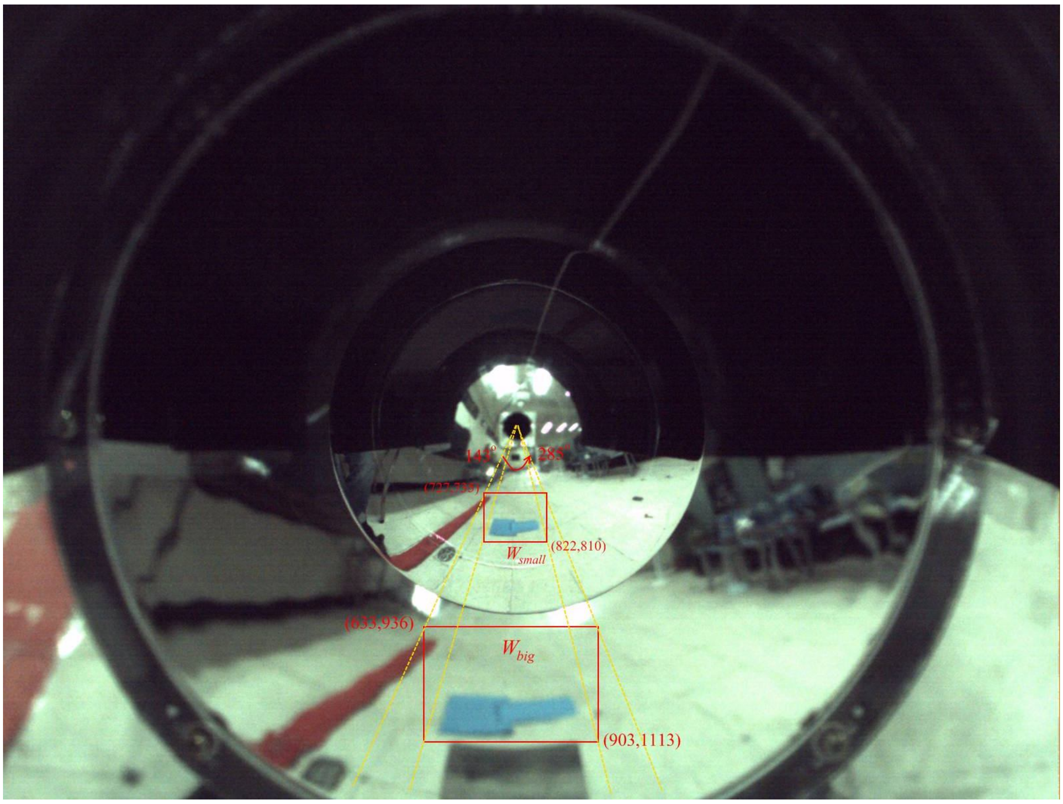
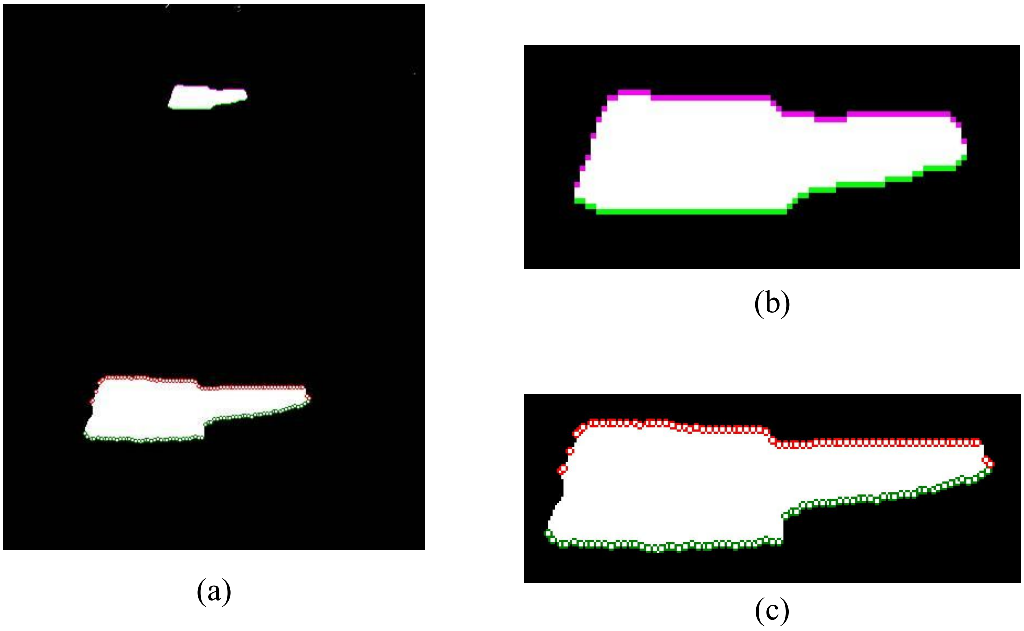


Figure 6.7 positions of extraction windows.

Figure 6.8 The obstacle features.



(a)

(b)

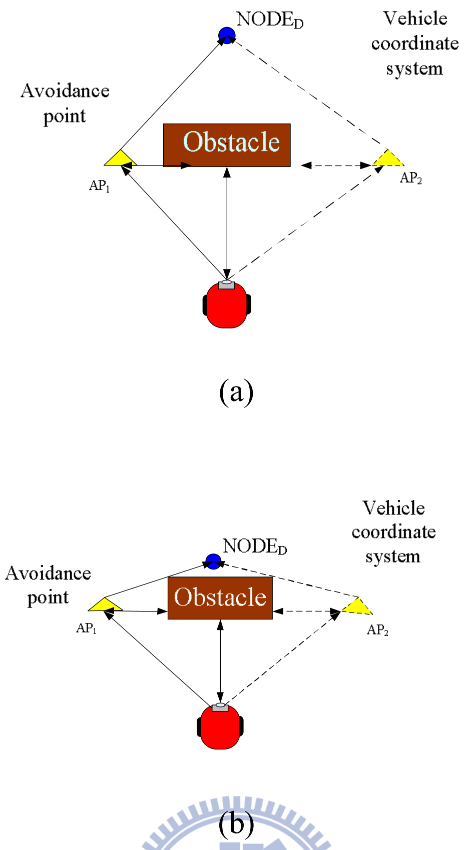
(c)

Figure 6.9 The outline feature points of the obstacle. (a) The result image with both portions of feature points. (b) The result of the upper feature points of (a). (c) The result of the lower feature points of (a).

6.2.3 Ideas of proposed method for obstacle avoidance

As an obstacle is detected, a new path for obstacle avoidance is necessary. In Chiang and Tsai's method [21], the depth distance and the width distance of an obstacle was computed by a 2D projection method. Then, the vehicle avoided the obstacle by inserting an avoidance node into the original node sequence and went to the next node through the avoidance node. The method is shown in Figure 6.9(a).

A problem is shown in Figure 6.9(b), that is, the obstacle is at a position which overlaps the position of the path node NODE_D or is very close to the node NODE_D that is the destination the vehicle will arrive at. As can be seen, the angle from the avoidance point to NODE_D becomes narrow and the vehicle will hit the obstacle along the avoidance path.



(a)

(b)

Figure 6.10 An illustration of the avoidance method. (a) The avoidance method proposed in Chiang and Tsai [21]. (b) A situation in an assumption.

A concept of a *virtual node* as well as Algorithm 6.2 is proposed to solve this problem, as illustrated in Figure 6.10. The vehicle is located in the interval from the path node, NODE$_F$, to the path node, NODE$_D$. After detecting the obstacle, several processes are executed by the system as described in the following:

- Calculate the distance D_0 between the vehicle and the obstacle, and the depth distance *DEPTH* of the obstacle by the description in Section 6.2.2.
- Compute the position of N_{VD} on a vector $\vec{V}$ by

$$\vec{V} = \text{NODE}_D - \text{NODE}_P, \tag{6.3}$$

$$d = 2\times D_0 + DEPTH, \tag{6.4}$$

where d is a distance in front of the vehicle.

- Compute the position of the vehicle by Algorithm 6.2 when the vehicle arrives at N_{VD}.

A problem occurs when the vehicle is at the guide line and trying to avoid obstacles. Figure 6.11(a) shows that the right avoidance point is located at a position out of the bound of the guide line. If the right path is the chosen path used to avoid the obstacle, the vehicle should change the avoidance path to the left one. Figure 6.11(b) shows that N_{VD} may be created at a position out of the bound of the guide line if NODE_D is a turning node. Before moving to N_VD from an avoidance point, the vehicle should change N_{VD} to $\mathrm{NODE}_{\mathrm{D+1}}$.

A procedure is performed to solve these problems mentioned previously before avoiding the obstacle and moving to N_{VD} from the current position through the avoidance point. A distance $d_{node2line}$ from each path node to the guide line should be recorded in the learning procedure and the proposed method is described in the following.

Algorithm 6.3. Avoidance of the vehicle out of the guide line.

Input: the distance $d_{node2line}$ from the path node to the guide line.

Output: none.

Steps:

Step 1. Compute the distance d_v from the vehicle to N_VD by Equation (6.4)$_{,}$ if NODE_D is a turning node;

Step 2. Set $\mathrm{NODE}_\mathrm{D} = \mathrm{NODE}_{\mathrm{D+1}}$ if $d_y > d_{node2line}$.

Step 3. Compute the distance d_{ap} from AP_2 to C in Figure 6.11(a).

Step 4. Choose the other avoidance path if $d_{ap} > d_{node2line} - \mathrm{W}_\mathrm{vehicle}$, where $\mathrm{W}_\mathrm{vehicle}$ is the width of the autonomous vehicle.

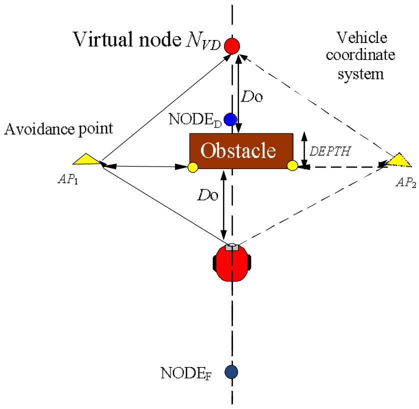
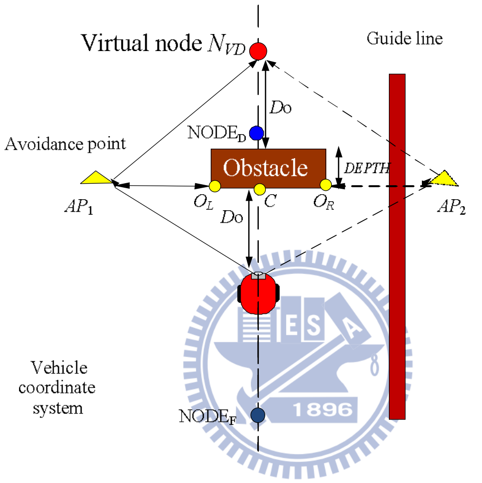


Figure 6.11 An illustration of a virtual node in the proposed method.



(a)

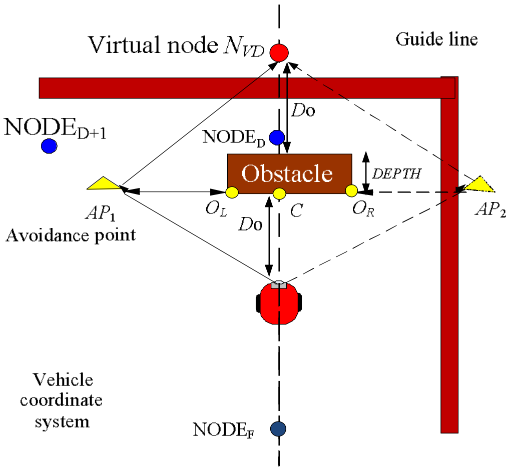


Figure 6.12 To avoid an obstacle at guide lines. (a) The obstacle is at the straight guide line. (b) The obstacle is at a corner.

6.2.4 Proposed method for obstacle avoidance

In this section, we summarize the different techniques mentioned previously into two algorithms for obstacle avoidance. Algorithm 6.4 is a decision procedure before avoidance of obstacles. Algorithm 6.5 is a procedure for avoidance of obstacles.

Algorithm 6.4. Preparation for obstacle avoidance.

Input: input image I_{input}.

Output:

Steps:

Step 1. Detect obstacles in I_{input} mentioned in Section 6.2.2.

Step 2. Compute the outlines in the top portion F_T and the bottom portion F_B of possible obstacle features mentioned in Section 6.2.2.

Step 3. Calculate the range data of F_T and F_B, respectively, as mentioned in Section 6.2.2.

Step 4. Compute D_0 from F_B as mentioned in Section 6.2.2.

Step 5. Go Algorithm 6.5 if D_0 is near enough (the vehicle is close to the obstacle).

Algorithm 6.5. Avoidance of obstacle.

Input: a set of the range data R of the obstacle and the path nodes.

Output: avoidance points AP_1, AP_2 and a virtual node N_{VD} for obstacle avoidance.

Steps:

Step 1. Compute the 3D information of the obstacle by subtracting F_B from F_T by Equations (6.1) and (6.2).

Step 2. Go to Step 4 without creating the avoidance points (Step 5 to Step 6) if the obstacle is too short to block the navigation path; Else, go to Step 3.

Step 3. Compute the avoidance points AP by the following steps:

(a) compute the vector of the obstacle perpendicular to the direction of the vehicle:

$$\vec{x} = O_L - O_R\,; \tag{6.3}$$

(b) compute the unit vector of $\vec{x}$ by:

$$\overrightarrow{x_u} = \vec{x} / \left\| \vec{x} \right\|; \tag{6.4}$$

(c) compute the avoidance point by:

$$AP_1 = O_L + \mathrm{W}_{\text{vehicle}} \times \overrightarrow{x_u}\,;$$

$$AP_2 = O_R - \mathrm{W}_{\text{vehicle}} \times \overrightarrow{x_u}\,. \tag{6.5}$$

Step 4. Compute the location of the virtual node N_{VD} by Equations (6.3) and (6.4).

Step 5. Compute the distances d_1 and d_2 of two avoidance paths from the current position to N_{VD} through AP_1 and AP_2, respectively.

Step 6. Choose an avoidance path from d_1 or d_2 with the smaller distance.

Step 7. Determine whether the avoidance path is available and whether N_{VD} should be changed by Algorithm 6.3.

Chapter 7 Experimental Results and Discussions

7.1 Experimental Results

In this chapter, we will show some experimental results of the proposed system in an outdoor environment. Figure 7.1 shows the experimental environment.

In the learning procedure, the system will follow the curbstone in the navigation. Figure 7.2(a) shows the curbstone in front of the vehicle and Figure 7.2(b) shows the resulting image of feature extraction. Figure 7.3(a) shows the curbstone at the lateral of the vehicle and Figure 7.3(b) shows the resulting image of feature extraction. If the system detects a human hand in the pre-defined region of the camera enclosure, it will change the guide line detection mode to the blind navigation mode. Figure 7.4(a) shows that the user instructs the vehicle by hand poses and Figure 7.4(b) shows the resulting image that was taken when the system was detecting a human hand. A map created in the learning procedure before executing the path planning process is shown in Figure 7.5(a) and Figure 7.5(b) shows the map modified after executing the path planning process.

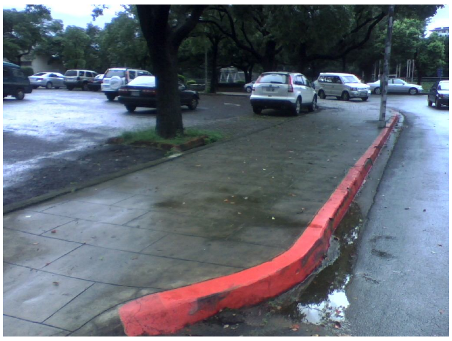
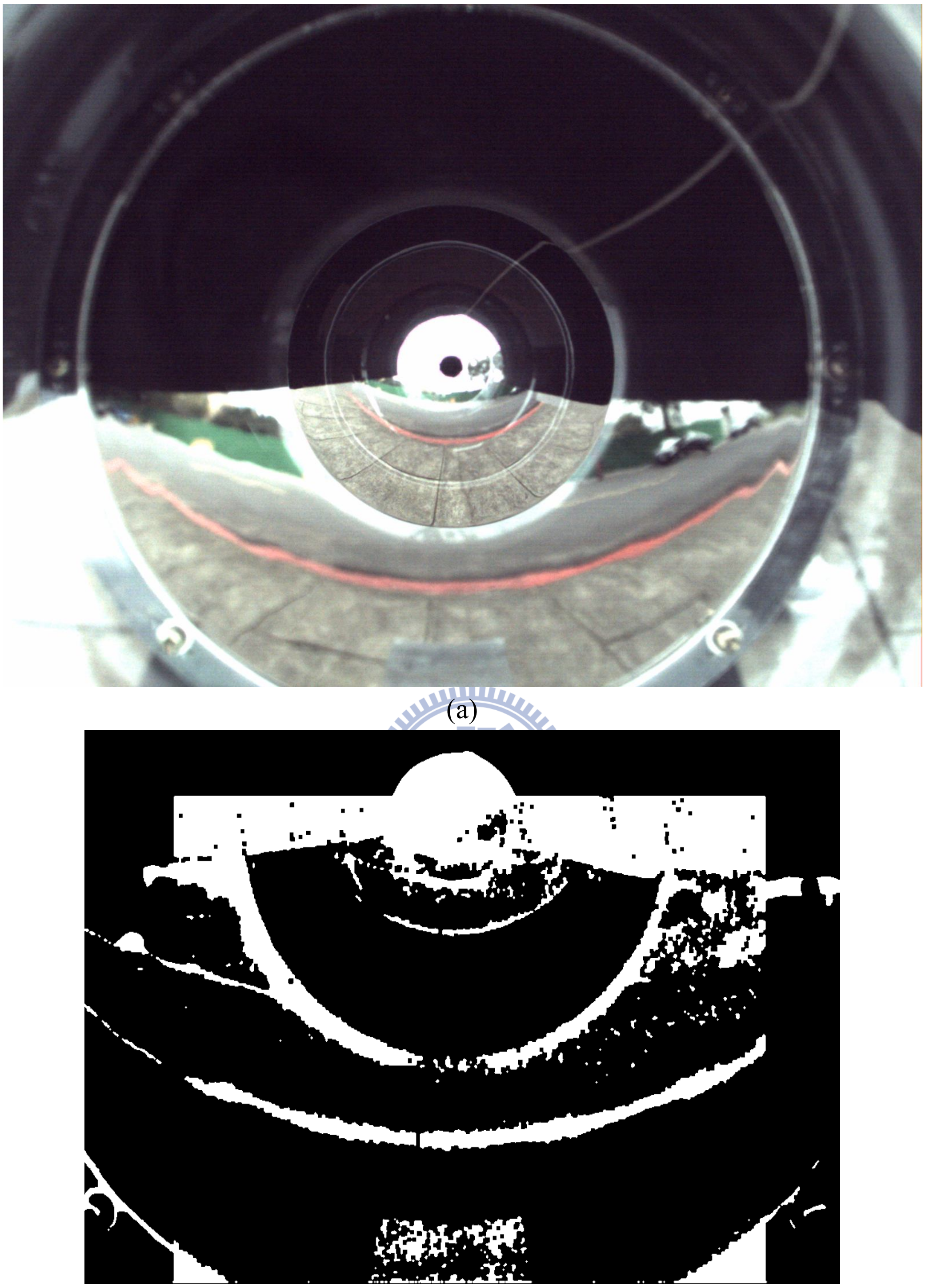


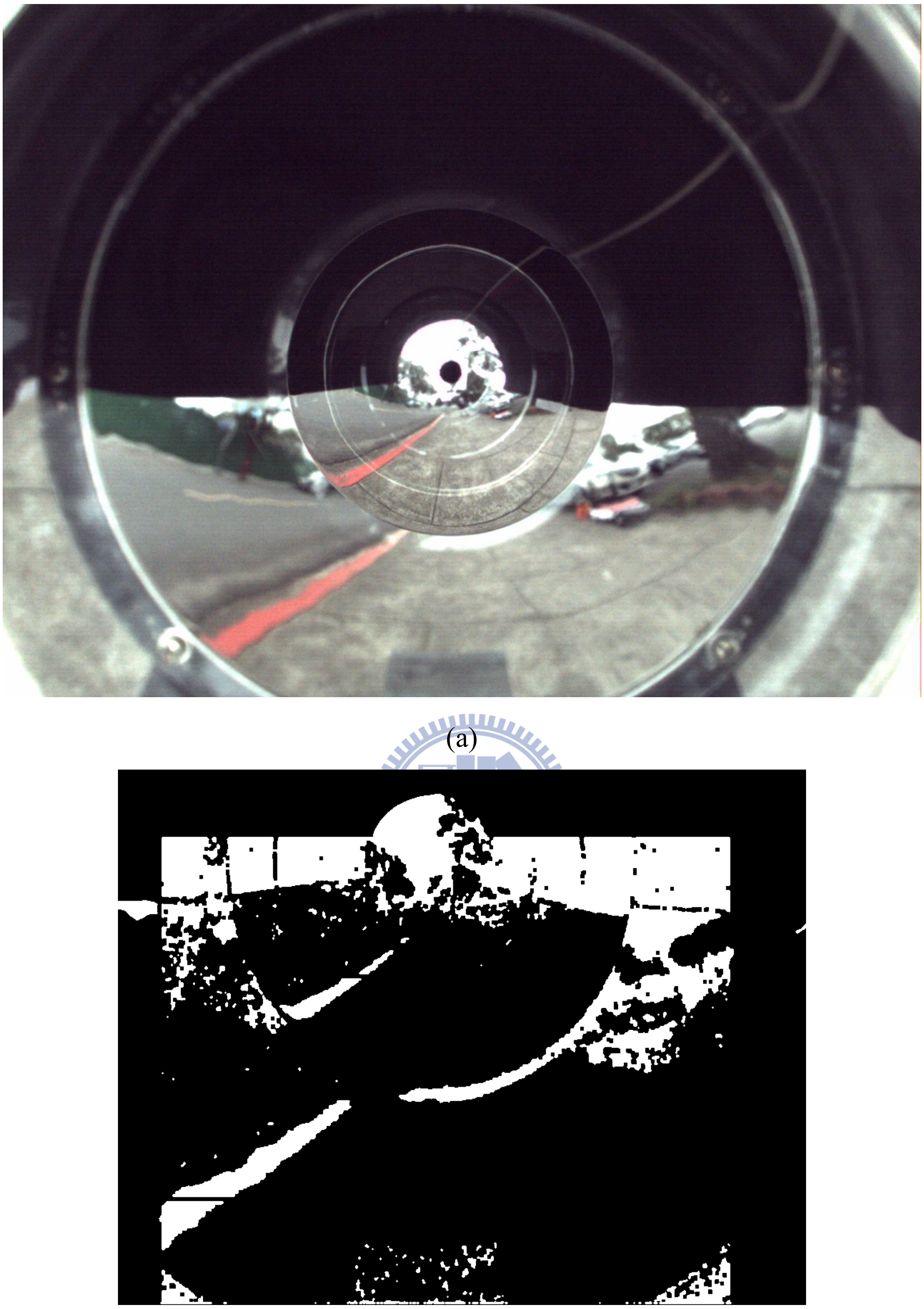
Figure 7.1 The experimental environment.



(a)

(b)

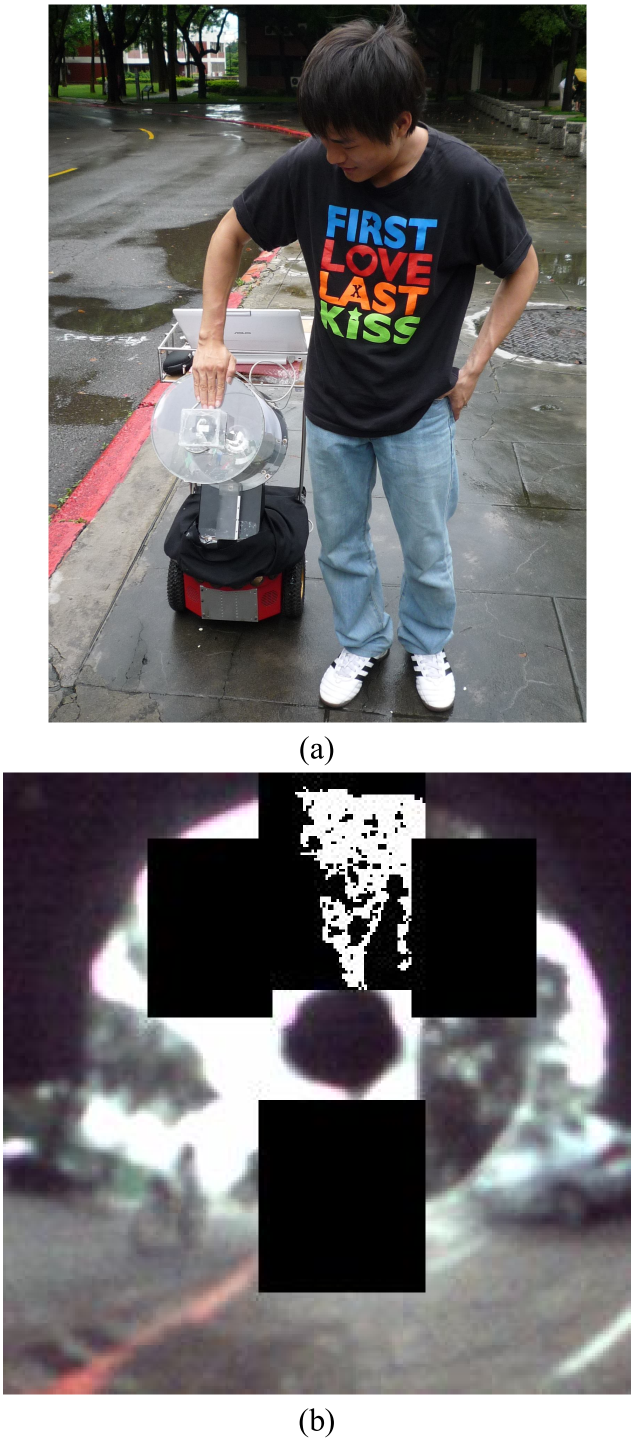
Figure 7.2 A curbstone appearing in front of the vehicle. (a) A captured image. (b) An image obtained from processing (a) with extracted feature points.



(a)

(b)

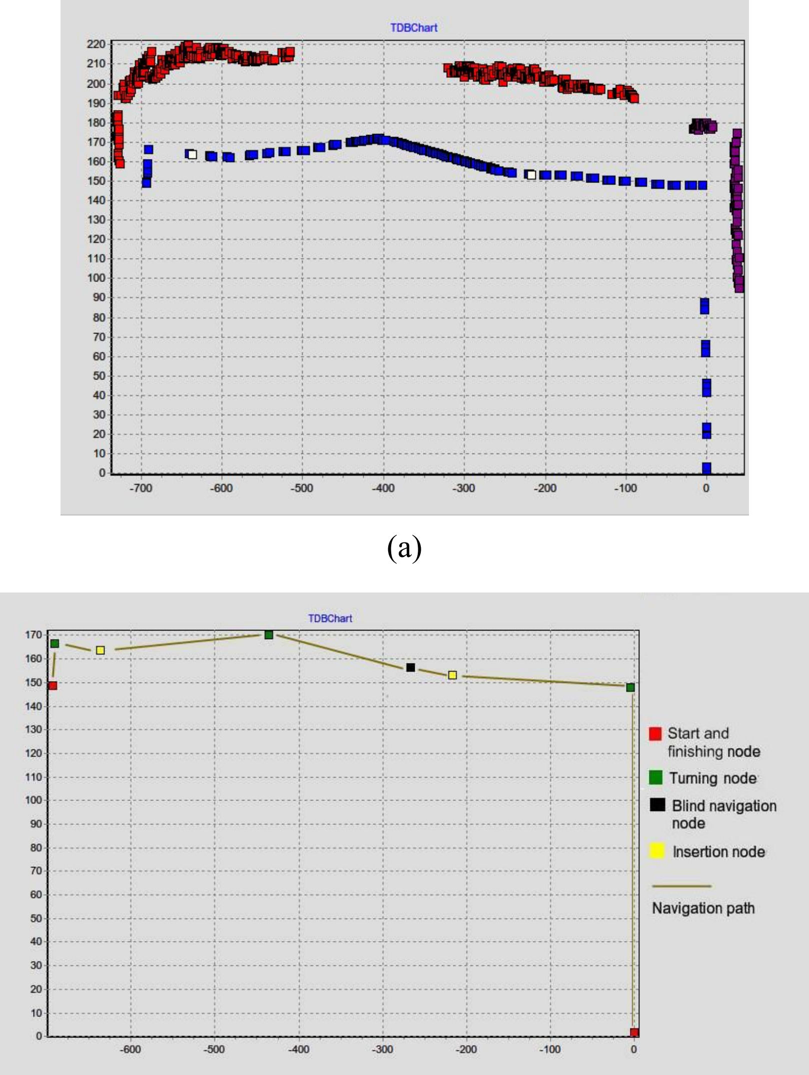
Figure 7.3 A curbstone appearing at the lateral of the vehicle. (a) A captured image. (b) An image obtained from processing (a) with extracted feature points.



(a)

(b)

Figure 7.4 A resulting image of hand pose detection. (a) A user instructing the vehicle by hand. (b) The human hand detected in a pre-defined region.



(a)

(b)

Figure 7.5 Two navigation maps. (a) A map created before path planning. (b) A map obtained from modifying (a) after path planning.

In the navigation procedure for guiding a blind person, the system synchronizes its speed with the person's speed. The signals captured from the six sonar sensors are shown in Figure 7.6. The synchronization method in the system used the sonar signals to compute the distance between the vehicle and the user. When an obstacle is detected, the system will avoid the obstacle if it is not flat and blocks the navigation path. An experimental result which shows that the system detected an obstacle is shown in Figure 7.7.

7.2 Discussions

By analyzing the experimental results of the learning procedure and the navigation procedure, we see some problems. First, in guide line detection, we use the sidewalk color to extract the guide line. If there are too many colors on the sidewalk, the system will be confused and cannot decide the features to use for guiding the vehicle. In this case, we have to learn all the colors as features. Also, the adjustment of the vehicle speed should be more quickly and smoothly. This can be done by using a faster CPU. Furthermore, because of the use of the plastic camera enclosure, light reflection yielded by the enclosure creates undesired lighting spots in the omni-image, as shown by as illustrated by Figure 7.8. This hopefully may be improved in future designs of the camera system. Finally, more experiments may be conducted to test the system in different environments.

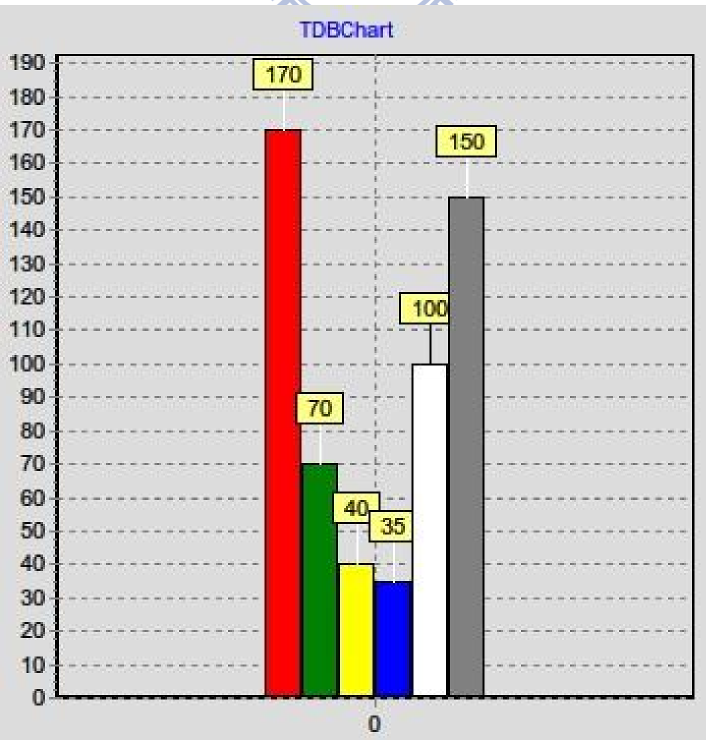
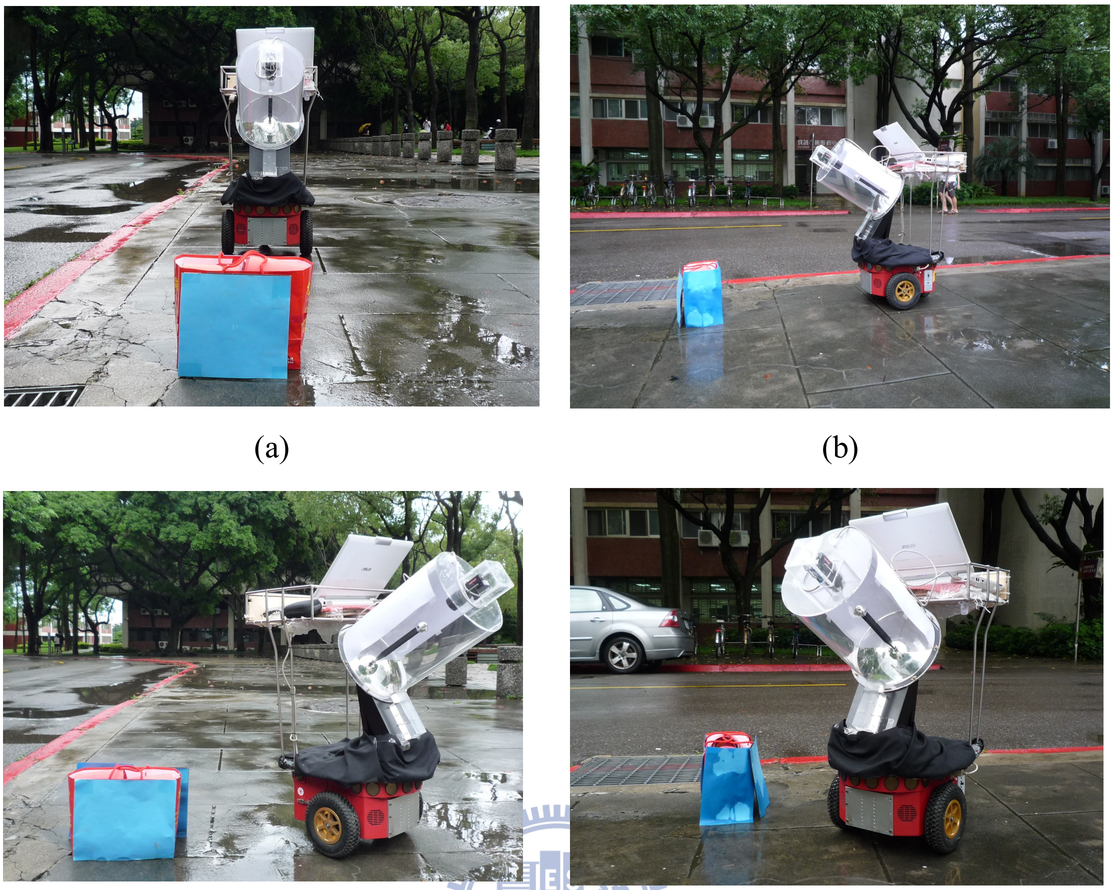


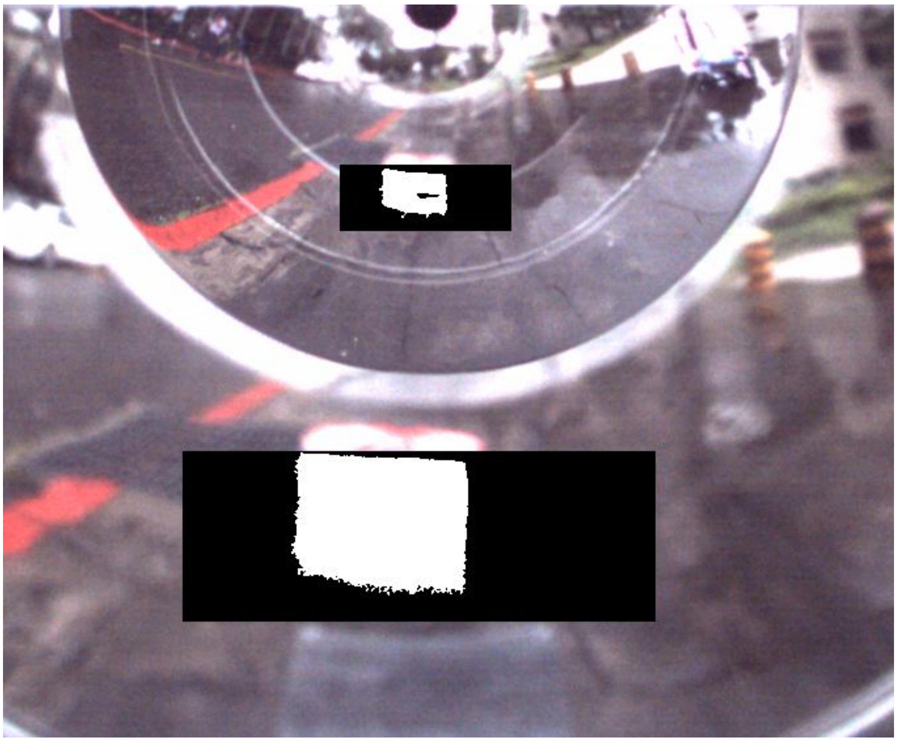
Figure 7.6 Sonar signals obtained when a person stands behind the vehicle.



(a) (b)

(c) (d)

Figure 7.7An experimental result of an obstacle avoidance process. (a) An obstacle in front of the vehicle. (b) A side view of (a). (c) The vehicle avoiding the obstacle. (d) A side view of (c). (e) Obstacle detection result using a captured image.



(e)

Fig. 7.7 An experimental result of an obstacle avoidance process. (a) An obstacle in front of the vehicle. (b) A side view of (a). (c) The vehicle avoiding the obstacle. (d) A side view of (c). (e) Obstacle detection result using a captured image (cont'd).

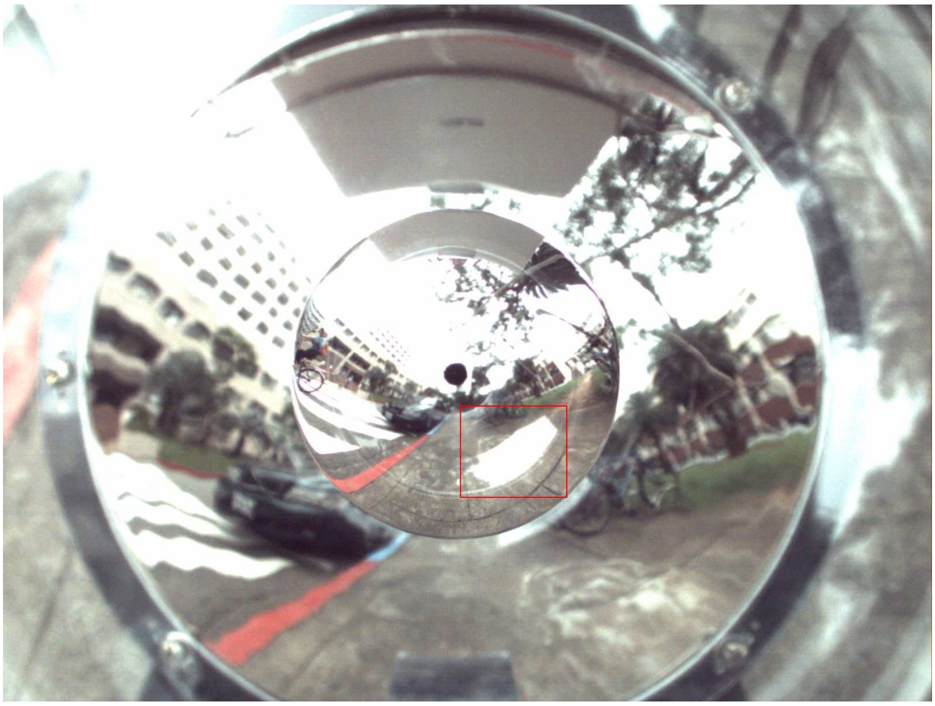


Figure 7.8 A light pollution on the omni-image.

Chapter 8
Conclusions and Suggestions for Future Works

8.1 Conclusions

In the study, we have designed an autonomous vehicle system equipped with a newly-designed two-mirror omni-camera as the visual sensor and navigating on sidewalks for use as a guide dog. Several techniques for implementing such a system have been proposed.

First, we have derived a new formula to design the two-mirror omni-camera which is composed of two reflection mirrors with hyperboloidal shapes and a traditional projective camera. The formula describes the relationship between the parameters of the mirror surface and the position of the mirror with respect to the camera. People can use the general formula to produce the two-mirror omni-camera with different parameters easily.

Next, we have proposed several new techniques for calibration of the camera and the mechanical error of the autonomous vehicle. The camera calibration technique is based on a pano-mapping technique proposed by Jeng and Tsai [22]. A mapping table which describes the relationship between the pixels and the elevation angles with respect to the hyperbolic mirror has been created and used in object localization and 3D data computation. To calibrate the mechanical error of the odometer equipped in the vehicle, a calibration model based on the curve fitting technique has been proposed. The mechanical error is reduced by the use of the proposed calibration model.

Furthermore, we proposed new techniques for vehicle guidance in the learning procedure and in the navigation procedure. To learn environment information, a semi-automatic method based on the line following technique has been proposed. The vehicle navigates on sidewalks by using the features of the curbstone and the technique of line following. If there exits no special feature or if the features are not easy to extract, a new human interaction technique based on human hand pose detection has been proposed to solve the problems. Different pre-defined hand poses on the camera enclosure are decoded to issue commands to guide the vehicle if it is necessarily.

To adapt to varying light intensities in outdoor environments, two techniques, called dynamic exposure adjustment and dynamic thresholding adjustment, have been proposed. Also, to create an environment map, a path planning technique has been proposed, which identifies path nodes at critical spots on a learned navigation route to create a navigation map.

To navigate in the environment with the path node map, a new technique is used to avoid dynamic obstacles on the navigation path without navigating out of the guide line on the sidewalk. To guide a person in the environment, a sonar signal processing method for synchronization between the speed of the vehicle and that of the person has been proposed. Also proposed is a technique for computing the location of the vehicle in the environment.

Good experimental results show the feasibility of the proposed system.

8.2 Suggestions for Future Works

According to our experience obtained in this study, several suggestions for future works are listed in the following.

1. The idea of the proposed calibration method can be used for other type of

the stereo camera.

2. Equipping the two-mirror omni-camera in the different directions, such as down-looking, on the autonomous vehicle may be tried.
3. Using different features, such as trees, road lights, and special signboards, to guide the autonomous vehicle navigating in the outdoor environment may be attempted.
4. Developing different human interface techniques by using different features such as human body, human motions, and human faces, etc. may be conducted.
5. Designing new algorithms to compute 3D range data of objects more quickly is worth study.

References

[1] J. Borenstein and I Ulrich, “The GuideCan — A Computerized Travel Aid for the Active Guidance of Blind Pedestrians,” *Proceedings of the IEEE International Conference on Robotics and Automation,* Albuquerque, NM, Apr. 21-27, 1997, pp. 1283-1288.

[2] C. C. Sun and M. C. Su, “A Low-Cost Travel-Aid for the Blind,” *M. S. Thesis*, Department of Computer Science and Information Engineering, National Central University, Jhongli, Taoyuan, Taiwan, June 2005.

[3] S. Tachi and K. Komority, “Guide dog robot,” *2nd Int. Congress on Robotics Research*, pp. 333-340. Kyoto, Japan, 1984.

[4] The Robot World.
http://www.robotworld.org.tw/

[5] National Yunlin University of Science and Technology.
http://www.swcenter.yuntech.edu.tw/

[6] J. Kannala and S. Brandt, “A Generic Camera Calibration Method for Fish-Eye Lenses,” *Proceedings of the 17th International Conference on Pattern Recognition,* Vol. 1, pp. 10-13, August 2004; Cambridge, U.K.

[7] C. J. Wu, “New Localization and Image Adjustment Techniques Using Omni-Cameras for Autonomous Vehicle Applications,” *Ph. D. Dissertation,* Institute of Computer Science and Engineering, National Chiao Tung University, Hsinchu, Taiwan, Republic of China, July 2009.

[8] S. K. Nayar, “Catadioptric Omni-directional Camera,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* pp. 482-488, June 1997, San-Juan, Puerto Rico.

[9] S. Baker and S. K. Nayar, “A Theory of Single-Viewpoint Catadioptric Image

Formation," *International Journal of Computer Vision,* Vol. 35, No. 2, pp. 175-196, November 1999.

[10] H. Ukida, N. Yamato, Y. Tanimoto, T. Sano and H. Yamamoto, "Omni-directional 3D Measurement by Hyperbolic Mirror Cameras and Pattern Projection," *Proceedings of IEEE International Instrumentation and Measurement Technology Conference,* Victoria, Vancouver Island, Canada, May12-15, 2008.

[11] Z. Zhu, "Omnidirectional Stereo Vision," *10th IEEE ICAR,* August 22-25, 2001, Budapest, Hungary.

[12] L. He, C. Luo, F. Zhu, Y. Hao, J. Ou and J. Zhou, "Depth Map Regeneration via Improved Graph Cuts Using a Novel Omnidirectional Stereo Sensor," *Proceedings of 11th IEEE International Conference on Computer Vision (ICCV 2007)*, Rio de Janeiro, Oct. 14-21, pp 1-8.

[13] S. Yi and N. Ahuja, "An Omnidirectional Stereo Vision System Using a Single Camera,“ *Proceedings of 18th International Conference on Pattern Recognition (ICPR'06)*, Hong Kong, Aug. 20-24, 2006.

[14] G. Jang, S. Kim and I. Kweon, "Single Camera Catadioptric Stereo System," *Proceeding of Workshop on Omnidirectional ,Vision, Camera Networks and Nonclassical cameras(OMNIVIS2005)*, 2005.

[15] K. C. Chen and W. H. Tsai, "A study on autonomous vehicle navigation by 3D object image matching and 3D computer vision analysis for indoor security patrolling applications," *Proceedings of 2007 Conference on Computer Vision, Graphics and Image Processing,* Miaoli, Taiwan, June 2007.

[16] J. Y. Wang and W. H. Tsai, "A Study on Indoor Security Surveillance by Vision-based Autonomous Vehicles with Omni-cameras on House Ceilings," *M. S. Thesis*, Institute of Computer Science and Engineering, National Chiao Tung University, Hsinchu, Taiwan, Republic of China, June. 2009.

[17] S. Y. Tsai and W. H. Tsai, "Simple automatic path learning for autonomous vehicle navigation by ultrasonic sensing and computer vision techniques," *Proceedings of 2008 International Computer Symposium*, vol. 2, pp. 207-212, Taipei, Taiwan, Republic of China.

[18] K. T. Chen and W. H. Tsai, "A study on autonomous vehicle guidance for person following by 2D human image analysis and 3D computer vision techniques," *Proceedings of 2007 Conference on Computer Vision, Graphics and Image Processing*, Miaoli, Taiwan, Republic of China.

[19] M. F. Chen and W. H. Tsai, "Automatic learning and guidance for indoor autonomous vehicle navigation by ultrasonic signal analysis and fuzzy control techniques," *Proceedings of 2009 Workshop on Image Processing, Computer Graphics, and Multimedia Technologies,* National Computer Symposium, pp. 473-482, Taipei, Taiwan, Republic of China.

[20] Y. T. Wang and W. H. Tsai, "Indoor security patrolling with intruding person detection and following capabilities by vision-based autonomous vehicle navigation," *Proceedings of 2006 International Computer Symposium (ICS 2006) – International Workshop on Image Processing, Computer Graphics, and Multimedia Technologies,* Taipei, Taiwan, Republic of China, December 2006.

[21] K. L. Chiang and W. H. Tsai, "Security Patrolling and Danger Condition Monitoring in Indoor Environments by Vision-based Autonomous Vehicle Navigation," *M. S. Thesis,* Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, Republic of China, June 2005.

[22] S. W. Jeng and W. H. Tsai, "Using pano-mapping tables to unwarping of omni-images into panoramic and perspective-view Images," *Proceeding of IET Image Processing*, Vol. 1, No. 2, pp. 149-155, June 2007.

[23] J. Gluckman, S. K. Nayar and K. J. Thoresz, "Real-Time Omnidirectional and

Panoramic Stereo," *Proceeding of Image Understanding Workshop*, vol. 1, pages 299–303, 1998.

[24] The MathWorks. http://www.mathworks.com/access/helpdesk/help/toolbox/images/f8-20792.html

[25] The Dimensions of Colour by David Briggs. http://www.huevaluechroma.com/093.php

[26] M. C. Chen and W. S. Tsai, "Vision-based security patrolling in indoor environments using autonomous vehicles," *M. S. Thesis*, Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, Republic of China, June 2005.