國立交通大學

多媒體工程研究所

碩士論文

利用多台 KINECT 裝置及自動車作園區安全巡邏之研究

Guidance of an Autonomous Vehicle for Security Patrolling in Park Areas Using Multiple Onboard KINECT Devices

研 究 生：何冠霖

指導教授：蔡文祥　教授

中 華 民 國 一 百 零 二 年 六 月

利用多台 KINECT 裝置及自動車作園區安全巡邏之研究

Guidance of an Autonomous Vehicle for Security Patrolling in Park Areas Using Multiple Onboard KINECT Devices

研 究 生：何冠霖　　Student：Kuan-Lin Ho

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 2013

Hsinchu, Taiwan, Republic of China

中 華 民 國 一 百 零 二 年 六 月

Guidance of an Autonomous Vehicle for Security Patrolling in Park Areas Using Multiple Onboard KINECT Devices

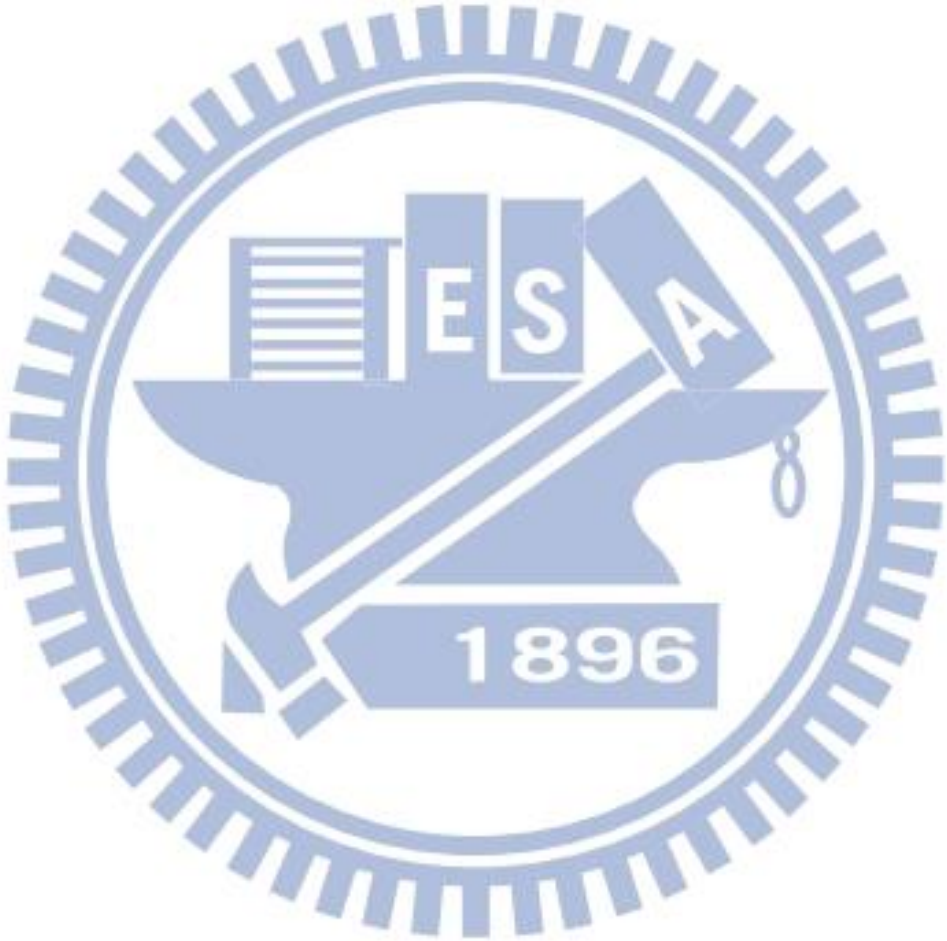
Student: Kuan-Lin Ho Advisor: Wen-Hsiang Tsai

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

ABSTRACT

A vision-based autonomous vehicle system equipped with KINECT devices for security patrolling on sidewalks in outdoor environments is proposed. A small-size vehicle with three onboard KINECT devices is designed to build the system. At first, a learning procedure is proposed for a trainer to guide the vehicle to extract specific features, including navigation path, color/depth information, pre-selected landmark objects, and vehicle location with respect to the landmark. Next, a strategy of vehicle navigation with a line-following capability is proposed, by which the vehicle may be guided to navigate according to the node data of the learned path, detect along-path landmarks using SURFs, and match the features with the learned data based on the measures of contrast difference and Euclidean distance. In addition, a vehicle location estimation technique for path correction utilizing the landmark matching result is proposed, which is based on the use of an ICP algorithm with the depth information as input according to the criterion of minimum MSE. Furthermore, techniques of ramp and curb-line detection have also been proposed, both for use to guide the vehicle safely on the path as well as to provide environment features and adjust the vehicle

orientation. Good experimental results show the flexibility and feasibility of the proposed methods for the application of security patrolling in outdoor environments.



利用多台 KINECT 裝置及自動車作園區安全巡邏之研究

研究生：何冠霖　　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

本研究提出了一個有電腦視覺之自動車系統，可應用於園區安全巡邏。該系統使用多台 KINECT 攝影機裝置做實驗平台。首先，使用者可利用一環境學習介面來控制自動車擷取環境中的特徵資訊，其中包括了航行地圖、特定路標的彩色及深度資訊，以及車輛與路標的相對距離。接著，提出了一個自動車航行的策略，讓車輛使用線段跟隨的技巧並根據學習的路徑節點依序航行，並且使用加速穩健特徵(speeded up robust feature, SURF)演算法，擷取沿路偵測到的物體之特徵點，再與學習的特徵資訊比較，計算其對比差異與歐氏距離，進行影像匹配。

此外，本研究也提出了一個自動車定位的技巧，可利用車輛與路標相對位置匹配的結果估計出車輛的位置，並且修正自動車航行的路線。此技巧是基於遞迴最近點(iterative closest point, ICP)演算法的概念，計算出深度影像之間的最小平方差做定位匹配之用。接著，還提出了坡道偵測與線段抽取的技術，這些技術可以使自動車安全地航行於所學習的路徑上，並且計算車輛航行中所遭遇環境的特徵與方向。

最後，依上述方法所做實驗的結果良好，顯示該等方法對室外安全巡邏的作用完整可行。

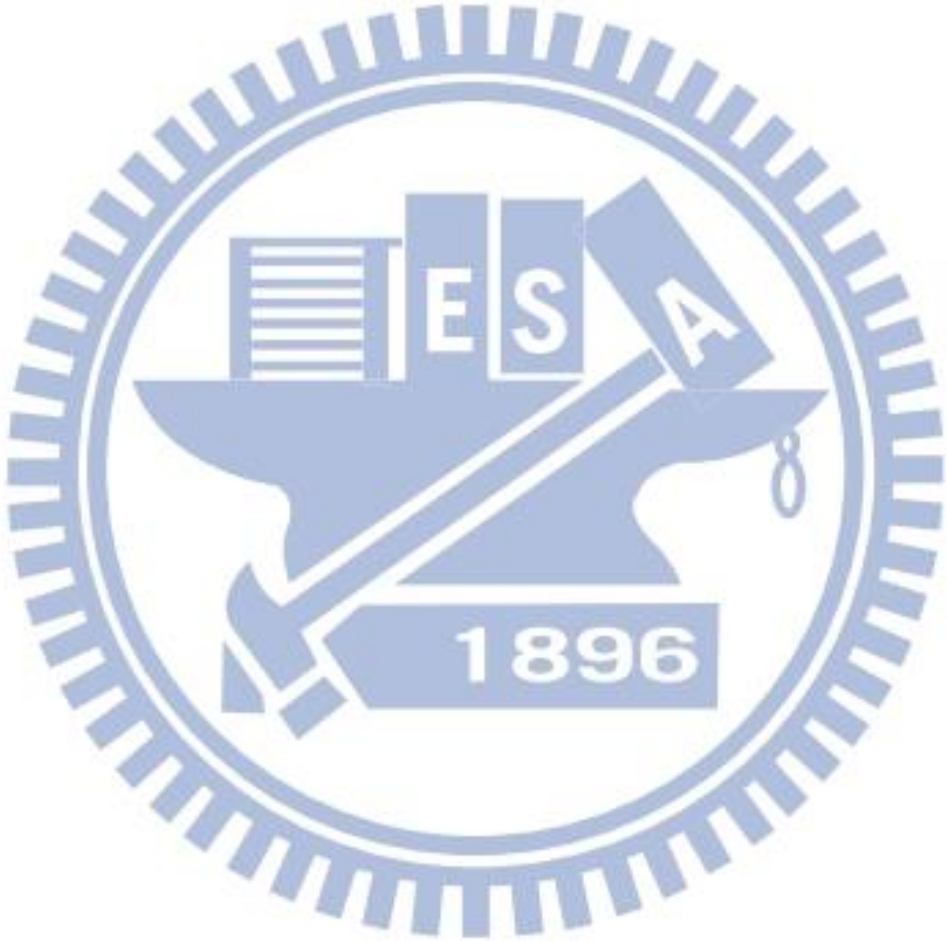
ACKNOWLEDGEMENTS

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

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

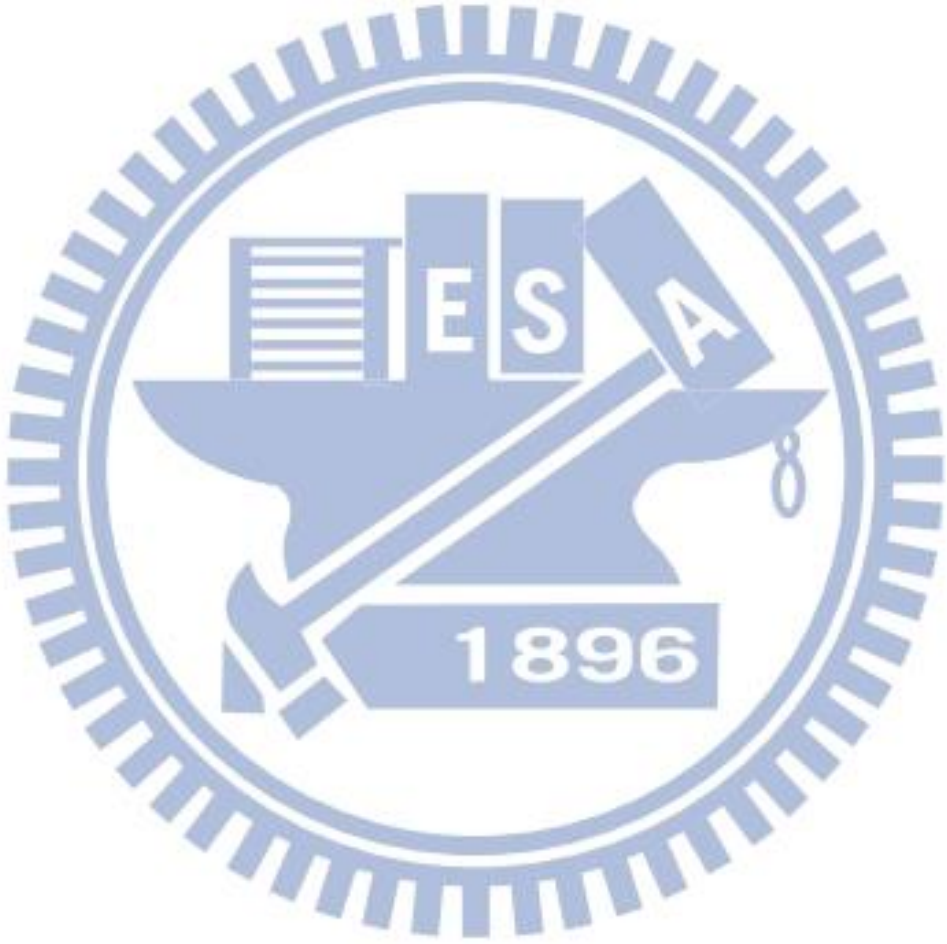
Finally, the author also extends his profound thanks to his dear mom and dad for their lasting love, care, and encouragement.

CONTENTS



LIST OF FIGURES

LIST OF TABLES



Chapter 1
Introduction

1.1 Motivation

Today, video surveillance systems are used widely in our life. Two examples are event data recorders installed on cars and public-space monitoring systems deployed almost everywhere in cities. But these systems usually are *fixed* at certain places or *affixed* to certain structures (walls, poles, etc.). This characteristic makes common video surveillance systems weak in their mobility for some application locations where no surveillance cameras are available. Consequently, in this study we try to design an autonomous vehicle system for monitoring interested areas in outdoor environments. The system may be regarded as a "mobile camera" movable to everywhere in the application environment.

In order to monitor environments with wider ranges, the autonomous vehicle must be designed to be equipped with video cameras or similar devices for image acquisition, as well as to include a built-in navigation system for controlling the vehicle. A common approach to autonomous vehicle navigation includes "training" the vehicle by use of environment features, followed by navigating the vehicle on a pre-planned path. In this study, we follow this concept to design the proposed autonomous vehicle system for video surveillance.

A major issue in designing such a system is how to navigate successfully in the environment. In this study, we concentrate on dealing with navigation on sidewalks in park areas for the purpose of security patrolling. Normally, an autonomous vehicle is

equipped with an odometer, which provides readings of the vehicle position and direction in every navigation cycle. Therefore, we can adjust the vehicle to correct its position according to such information. However, the vehicle position parameter which the odometer provides is often not sufficiently precise because the autonomous vehicle usually suffers from incremental mechanic errors in its navigation process due to manufacturing imprecision in its structure. One good solution to this problem is to continually estimate the vehicle position by use of pre-learned objects, called *landmarks*, in the surroundings along the navigation path, which is a sidewalk in this study as mentioned.

In recent years, using the Microsoft KINECT device for research is becoming popular. A reason is its convenience for acquiring 3D data of the real-world space. Some interesting applications have been proposed. The 3D data acquired by a KINECT device, often called RGB-D data, includes a depth image and a color image. The inherent 3D nature existing in the depth and color images makes it easier for people to conduct works of object detection and localization, although it has some restrictions on their uses for certain applications. We will describe the structure of the KINECT device in the following chapter. In this study, we try to design a vision-based autonomous vehicle equipped with more than one KINECT device which can navigate on sidewalks in park areas for security patrolling.

In summary, the goal of this study is to develop an outdoor navigation system with the following capabilities:

1. learning paths composed of nodes on sidewalks semi-automatically;
2. learning landmarks along sidewalks;
3. navigating automatically along sidewalks using learned landmarks for vehicle localization;
4. navigating to goals successfully on learned paths;

1.2 Survey of Related Works

In this section, we give a survey of previous works about outdoor navigation techniques, related applications using KINECT devices, landmark detection techniques, and landmark localization techniques in indoor or outdoor environments, .

In recent years, more and more research results of navigation systems using the KINECT device [1] have been reported. The emergence of the KINECT device facilitates captures of 3D image data and calculation of 3D space coordinates. But a problem arises when it is used for image acquisition during the day time, i.e., the infrared ray sensor equipped on the KINECT device is interfered by the infrared ray existing in the sunshine so that no depth data are provided by the KINECT device. Therefore, most KINECT devices are used in indoor environments. Correa and Sales et al. [2, 3] proposed an indoor-environment navigation system for video surveillance using the Kinect sensor. The system is based on a reactive navigation scheme, a finite-state machine, and an artificial neural network (ANN). Cunha et al. [4] proposed a robotic platform based on the use of a cooperative autonomous mobile robot with an advanced distributed architecture (CAMBADA) to navigate in indoor environments. Biswas and Veloso [5] proposed a fast sampling plane filtering (FSPF) algorithm to reduce the computation time required for indoor mobile robot localization and navigation.

In addition, some application systems using KINECT devices in outdoor environments have been proposed. Robledo, Cossell and Guivant [6] proposed the use of a KINECT device to plan a safe path to be followed by a bicycle rider. And Rasmussen [7] proposed a navigation system for tracking trails in outdoor environments with low or no sunlight.

When using any of the above-mentioned Kinect-based systems for navigation, it

is necessary to conduct the work of vehicle localization. This shows that vehicle localization is a core technique for use in implementing a navigation system. A variety of localization techniques are reviewed below. Willis and Helal [8] provided a navigation system which uses the radio frequency identification (RFID) technique to identify locations in buildings and rooms. Lisa et al. [9] utilized a DGPS (differential GPS) device to conduct localization in indoor and outdoor environments. Chen and Tsai [10] proposed an autonomous vehicle for indoor navigation using ultrasonic sensors. In [11], the GPS was used as a tool for vehicle localization as well.

In addition, vision-based devices are used widely for vehicle localization and navigation. Chen and Tsai [12] proposed a vehicle localization technique using perspective cameras, which adjusts the position of a vehicle by keeping watch over learned objects based on image matching using SIFT features. Another technique of vehicle localization in indoor environments by watching house corners was proposed by Chiang and Tsai [13]. Atiya and Hager [14] designed a vision-based system which computes the vehicle location in real-time. Moreover, in some other applications, cameras and other devices were combined together for use as an environment sensing device. Lopez et al. [15] combined a laser and a robot's camera together to compute the robot location. Tsai and Tsai [16] used a PTZ camera and an ultrasonic sensor to direct vehicle patrolling and people following. Agrawal and Konolige [17] proposed a system which uses stereo cameras and a low-cost GPS sensor.

1.3 Overview of Proposed System

The goal of this study is to design a vision-based autonomous vehicle system to navigate in outdoor environments. For this purpose, we use multiple Microsoft KINECT devices and an autonomous vehicle to build the system. An illustration of

the proposed system is shown in Fig. 1. We then propose some techniques for use on the system to navigate the vehicle on pre-learned paths. The major process of this system may be divided into two stages: the learning stage and the navigation stage.

Figure 1.1 Image of proposed autonomous vehicle system with three KINECT devices.

In the learning stage, we “learn” some image features provided by the KINECT devices, and some environment parameters along pre-selected paths on sidewalks before vehicle navigation. Then, in the navigation stage, we conduct vehicle navigation along the pre-learned path using the learned features and the images taken of the currently-visited landmarks. The details of the two stages are illustrated in Figs.

1.2 and 1.3, respectively, and discussed in the following.

A. The learning stage

At first, a prior work conducted before learning is to train the KINECT device system equipped on the vehicle. In this study, we deploy three KINECT devices on the vehicle, facing to the left, the front, and the right. Each KINECT device may be used to acquire a color image and a depth image simultaneously. To combine the two images into a 3D image, a problem of "shifting" between them due to the structural design of the KINECT device should be solved. Specifically, the two images not only should be "calibrated" into an identical image coordinate system, but also the relation between the identical image coordinate system and the 3D space coordinate system for the real world should be found out. For this purpose, we "calibrate" the depth and color image data using the KINECT SDK for Windows developed by Microsoft [18], and "calibrate" the coordinate systems according to the principle of the pinhole camera model. The details of these calibration techniques are described in subsequent chapters.

After the above image calibration work is done, the next work is to guide the autonomous vehicle to learn path information, including a sequence of landmarks, the vehicle poses in the path, the involved KINECT-device numbers, and some other environment information. After brining the vehicle to an area of interest, a path learning work is started. For this, we propose the use of two navigation modes: one being "navigation by following the curb line on the sidewalk"; the other being "manual control by the trainer". After being assigned the first mode, the vehicle starts to navigate toward a pre-selected goal. If a landmark has been selected in the path, the trainer may as well guide manually the vehicle using the second mode to an appropriate position to record the image features and the local position of the landmark. Furthermore, some information about the camera and the outdoor

environment are also recorded during the path learning process. Finally, when the learning process is over, all of the learned data are integrated into a set of path information and kept in a database.

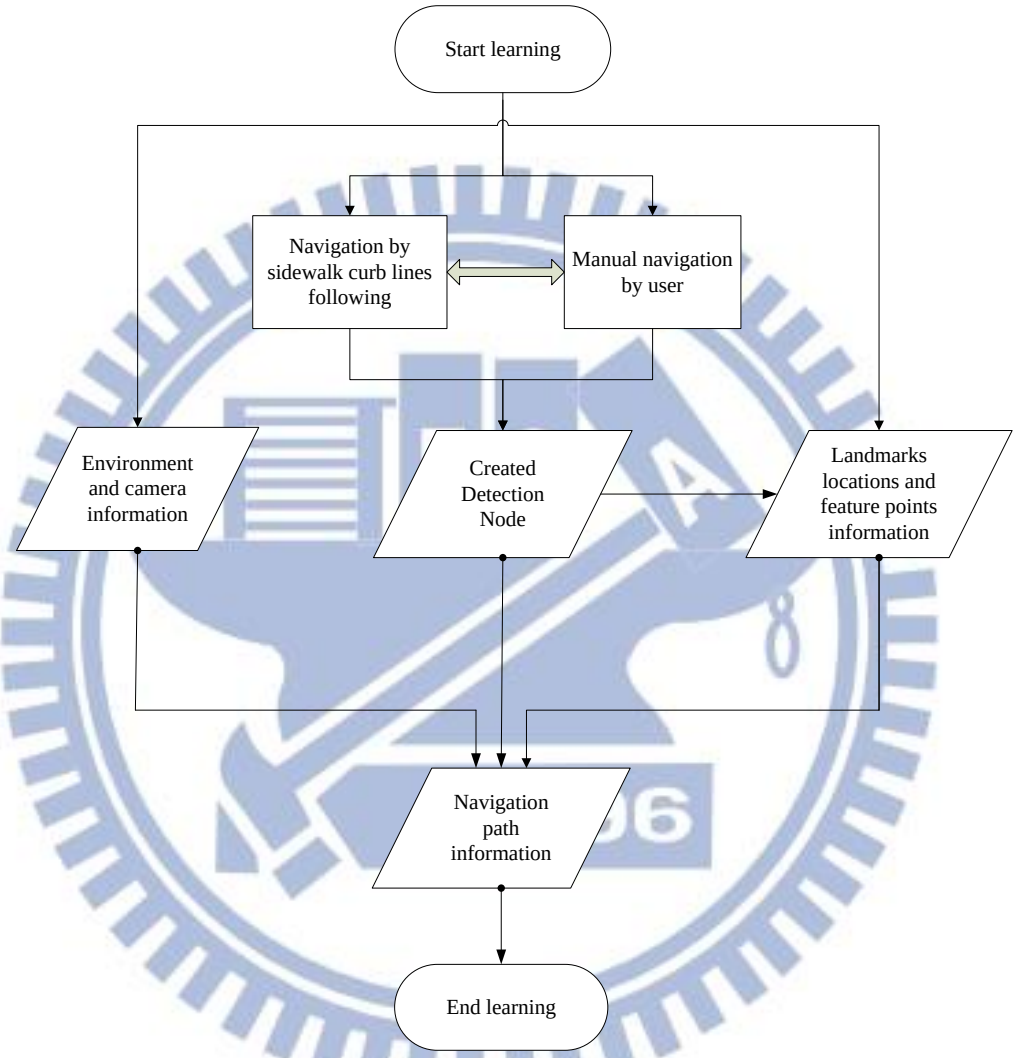


Figure 1.2 Flowchart of proposed learning stage.

B. The navigation stage

In the navigation stage, the path information which has been collected in the learning stage is utilized. The major tasks of this stage include landmark detection, vehicle location modification, and vehicle navigation. In principle, the autonomous vehicle moves constantly forward toward the goal according to the "node-based"

learned database on the learned path, where each node represents a spot on the path where a learning task as described previously is performed. When the autonomous vehicle navigates from a node to the next in the learned path, it can choose one of two pre-defined navigation modes. The first mode is "navigation by following the curb line on the sidewalk," and the other is "navigation by the odometer reading" provided by the vehicle system. The second mode will also be called the *blind navigation mode*. If the user chooses to use the first mode, the curb line on the sidewalk is detected continuously and the technique of line following is adopted to adjust the orientation of the vehicle, when necessary, during the navigation process. When the autonomous vehicle navigates to a non-curb position, the system will get into the second mode and move forward "blindly" according to the odometer readings as will be described in the subsequent chapters.

In addition, we use *fixed* along-path objects such as light poles, hydrants, and trees as landmarks for vehicle localization in this study. Specifically, we modify the vehicle position with respect to each detected landmark to eliminate accumulated mechanical or vision-processing errors during the navigation process. Finally, we propose an algorithm to combine SURFs and an iteration scheme to conduct vehicle localization in outdoor environments. And a technique of line following using the depth image only is proposed as well to guide the vehicle to navigate along the red-colored curb of the sidewalk. With the above-mentioned techniques, the autonomous vehicle can navigate safely to the end of the navigation stage.

1.4 Contribution of This Study

Some contributions of this study are described as follows.

1. A semi-automatic method for training an autonomous vehicle for outdoor

navigation using commonly-seen objects on sidewalks is proposed.

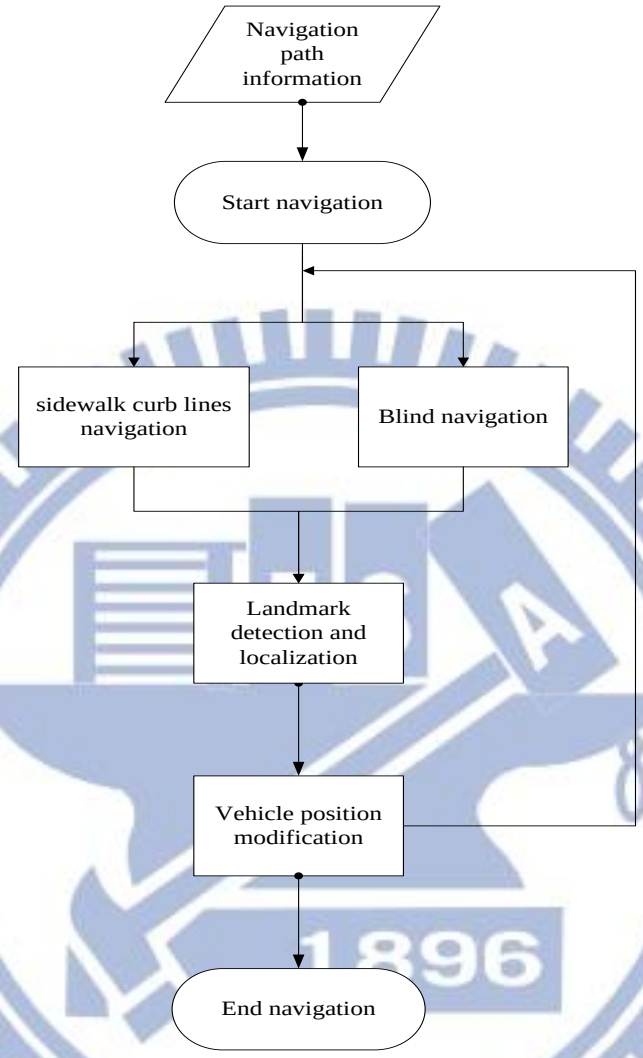


Figure 1.3 Flowchart of proposed navigation stage.

2. A vision-based method for estimating the vehicle location to reduce mechanical errors for vehicle navigation is proposed.
3. A vision-based method for vehicle navigation by following the curb line on the sidewalk is proposed.
4. Techniques for detecting landmarks of ramps and tree trunks are proposed.
5. A method for landmark detection and vehicle localization using depth data only is proposed.

1.5 Thesis Organization

The remainder of this thesis is organized as follows. In Chapter 2, we describe the configuration and the operation processes of the proposed system. In Chapter 3, the proposed learning technique for use in outdoor environments is described. In Chapter 4, the proposed navigation strategy for use in outdoor environments, which includes the ideas, localization techniques, and detailed navigation algorithms, is described. In Chapter 5, the proposed methods for landmark detection and localization using 3D image data are described. In Chapter 6, the proposed techniques for following the curb line, landmark detection, and vehicle localization using depth data only are described. Some experimental results are shown in Chapter 7. Finally, Conclusions and suggestions for future works are given in Chapter 8.

Chapter 2
System Configuration and Processes

2.1 Introduction

For video surveillance, we designed a vision-based autonomous vehicle system and trained it to monitor an area of interest in a park area. In this study, we choose a path on a sidewalk in National Chiao Tung University for the training. In order to conduct security patrolling along the path on the sidewalk quickly and stably, we installed three KINECT devices on an autonomous vehicle to construct a mobile security-monitoring system for use as experimental platform in this study. Acquisition of 3D data is made easier by the use of the KINECT devices, and the use of a small and flexible vehicle is a good choice, as done in this study. Also, we need to design processes to control the systems of the KINECT devices, the vehicle, and a communication mechanism for connecting the former two systems to analyze their data. The entire system configuration, including hardware and software, is introduced in Section 2.2, and the structure of the used KINECT devices is described in Section 2.3.

Furthermore, to navigate in an unknown environment, a learning strategy is needed to “teach” the vehicle where to navigate, what to monitor, and how to adjust its locations in each navigation session. Finally, a good navigation strategy which can lead the autonomous vehicles to the goal safely also need be designed. We will describe the learning and navigation processes for the adopted vehicle and the associated principles in Section 2.4.

2.2 System Configuration

In this study, we use the Pioneer 3-DX vehicle made by MobileRobots, Inc. as a platform for our experiments. The vehicle is equipped with three KINECT devices, facing to difference directions (facing the front, left-forward, and right-forward), as shown in Fig. 2.1. The KINECT devices are new and specially designed by Mirosoft in recent years, so we would like to describe it in detail in Section 2.3, which includes the structure of the sensor and the coordinate calibration process. The hardware architecture, and the software including the application programming interfaces and development tools we use, will be described in Sections 2.2.1 and 2.2.2, respectively.

2.2.1 Hardware Configuration

The hardware architecture of the proposed autonomous vehicle system is shown in Fig. 1.1. It can be divided into three major components: the vehicle system, the KINECT-device system, and the control system. We will describe these systems, respectively, in the following.

The vehicle has an aluminum body of the size of 44cm×38cm×22cm with two 19cm-sized wheels and a caster. The vehicle can climb a 25% grade and sills of 2.5cm. On flat floors, the vehicle can reach a forward speed of 160cm per second and a rotation speed of 300 degrees per second. Moreover, the vehicle has 16 ultrasonic sensors and three 12V rechargeable lead-acid batteries which supply the power for 18-24 hours if fully charged. The system can return its status parameters in each navigation cycle, which includes the position and orientation of the vehicle with respect to its initial pose. The system is shown in Fig. 2.1.

The second major component is the KINECT-device system which includes three KINECT devices, facing to three directions as mentioned previously, as shown in Fig.

2.2. In this study, we try to navigate the vehicle in outdoor environments quickly and stably. Therefore, we use multiple KINECT devices to reduce 3D data computation time and the hardware operation time. The structure of the KINECT-device system will be described in more detail in the next section.

(a) (b)

Figure 2.1 The vehicle Pioneer3-DX used in this study. (a) A front view. (b) A back view.

(a) (b) (c)

Figure 2.2 Three different directions of the hardware configuration, which includes a vehicle and three KINECT devices. (a) The right side. (b) The front side. (c) The left side.

Finally, in the third major component — the control system, we use a laptop computer as the control unit. It is of model R840 produced by TOSHIBA Computer, Inc. We use an RS-232 interface to connect the laptop computer to the autonomous

vehicle, and use USB's to connect the laptop computer to the KINECT devices.

2.2.2 Software Configuration

The MobileRobots Inc. provides an application process, called Advanced Robotics Interface for Applications (ARIA), which is an object-oriented programming interface written in the C^{++} language and may be used to control the vehicle. The lowest-level data and information of the vehicle are also retrieved easily by means of the ARIA. Therefore, we can use the ARIA as an interface to communicate with the embedded system of the vehicle. Besides, in this study we use the Borland C++ builder 6.0 as the development tool to control the vehicle.

For the KINECT devices to function under the Windows system, the Microsoft Inc. provides a development tool called *Kinect for Windows Software Developer Kit (SDK)*. We use this SDK to capture 3D images and calibrate the KINECT devices. But this SDK not only needs the operation system *Windows 7*, but also the two development tools of *.NET Framework 4.0* and *Microsoft Visual Studio 2010*. Therefore, to develop the KINECT-device system, we use the language of *Microsoft Visual C++ 2010* with *.NET Framework 4.0* under the *Windows 7* operating system.

2.3 Structure of Microsoft KINECT Device

At first, we introduce the hardware architecture of the KINECT device. The KINECT device includes a color VGA video camera, a depth sensor, a multi-array microphone, and a tilt motor for sensor operations and adjustments. The horizontal field of view of the KINECT device is 57 degrees, the vertical field of view is 43 degrees, and the physical tilt range is ± 27 degrees. The major difference between

common cameras and the KINECT device is that the KINECT device has a depth detection sensor. The depth detection sensor is composed of an infrared projector and a monochrome Complementary Metal-Oxide Semiconductor (CMOS) sensor, which work together to obtain the distance information between the depth sensor and the objects in front of the Kinect device. The resolution of the image acquired with this color VGA video camera is 1280×960, and the resolution of the image acquired with the depth sensor is 640×480. The depth range provided by the KINECT sensor using the *Kinect for Windows* SDK is from 800mm to 4000mm. But the effective range of distances between the KINECT device and the user is from 1200mm to 3600mm, which is advised by the KINECT development official website. And other specifications of the hardware are not described in detail here due to the page limit. An illustration of the KINECT device is shown in Fig.2.3.

Figure 2.3 Hardware of the KINECT device. (a) Structure of external. (b) Structure of internal.

Furthermore, calibration of the camera parameters before vehicle navigation is necessary. In this process, and a rotation problem and a shifting one will arise in the 3D space coordinates. A solution to the rotation problem is to calibrate the related parameters before vehicle navigation for each navigation session. The KINECT device can be calibrated with some calibration functions and parameters provided in

the Kinect-for-Windows SDK. In this study, we tilt the field of view of each KINECT device to the zero-angle position using the tilt motor in the sensor before vehicle navigation. And to solve the shifting problem between the color image and the depth image, we use certain functions provided by the Kinect-for-Windows SDK [19].

After the above two problems are solved, we can obtain a 3D image in which the original color and depth images are in the same image coordinate system. But these 3D image data are just the 3D depth coordinates combined with the 2D image coordinates, so they must be transformed into the 3D space coordinate system integrally. For this purpose, we apply the principle of the pinhole camera model to conduct the conversion of the 3D image coordinates into the 3D space coordinates.

As shown in Figure 2.4, a space point *G* at coordinates (*X*, *Y*, *Z*) in the 3D space is projected through the lens center of the camera onto the image plane, where the image plane may be the depth image or the color image. The depth value *d* is provided by the KINECT device, but we do not have its correct coordinates in the 3D space. Therefore, we compute the direction vector of the image plane to the lens center by using the focal length *f* of the depth image provided by the KINECT-for-Windows SDK [19] and image coordinates (*u*, *v*). Then, we can calculate the correct 3D space coordinates (*X*, *Y*, *Z*) of point according to the similar-triangle principle using the depth value *D*. Specifically, by the principle and following the direction vector starting from the image plane, going through the lens center of the camera, and projects finally onto the 3D space point *G*, we can compute the 3D space coordinates (*X*, *Y*, *Z*) as follows.

At first, apparently as can seen from Fig. 2.4, we can calculate the distance *d* between the image plane and the lens center by the following equation:

$$d = \sqrt{u^2 + v^2 + f^2}\ , \tag{2.1}$$

then, according to the similar-triangle principle, since the two triangles *OCI* and *GG'*O are similar, we can know the following equalities:

$$\frac{X}{u} = \frac{Y}{v} = \frac{Z}{f} = \frac{D}{d}, \tag{2.2}$$

from which we can derive the following equations to describe the relation between the image coordinates (*u*, *v*) and the corresponding space coordinates (*X*, *Y*, *Z*):

$$X = \frac{D \times u}{\sqrt{u^2 + v^2 + f^2}}; \; Y = \frac{D \times v}{\sqrt{u^2 + v^2 + f^2}}; \; Z = \frac{D \times f}{\sqrt{u^2 + v^2 + f^2}}. \tag{2.3}$$

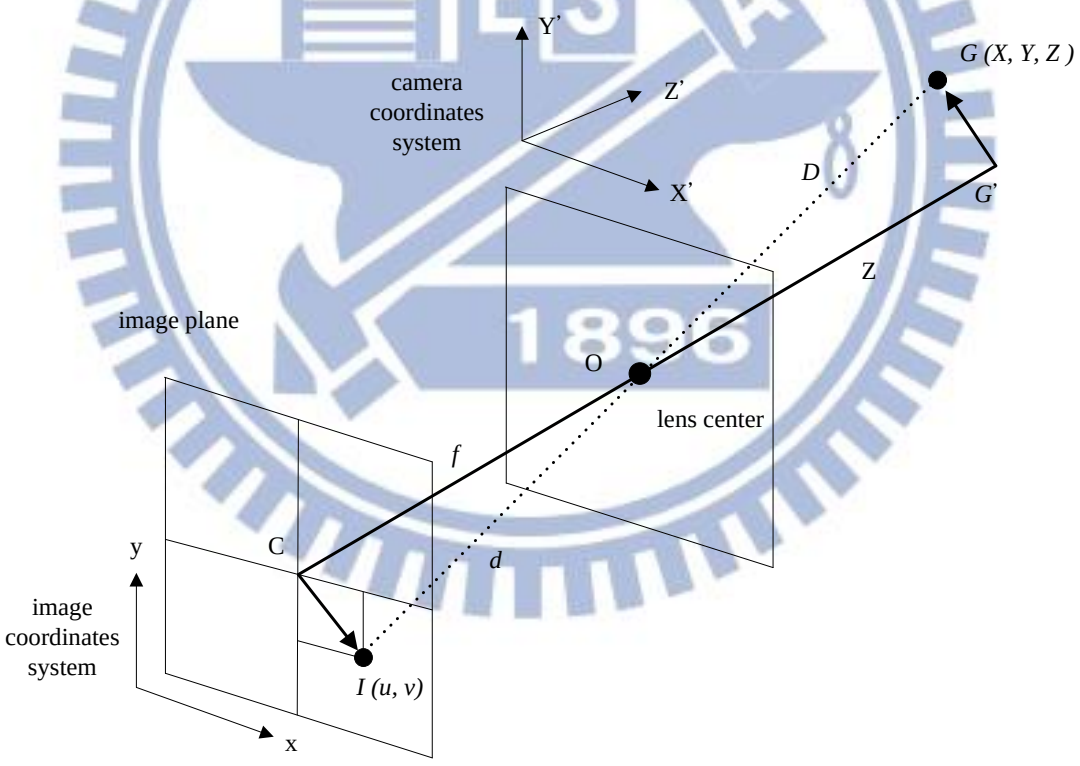


Figure 2.4 A pinhole camera model.

2.4 System Processes

2.4.1 Learning Process

To conduct security patrolling in an outdoor environment, a learning process is necessary. We describe the information which the vehicle should record in this process in detail now. At first, we bring the vehicle to a selected path in an outdoor environment, which is a part of the National Chiao Tung University campus. Because the goal is security patrolling, we use the vehicle to patrol along a path on a sidewalk in that part of the campus. Furthermore, we propose a "curb line following" technique in the proposed system for vehicle guidance. Finally, the environment information and camera parameters are recorded at difference positions on the path. The entire learning process is shown in Fig. 2.3.

In order to help users to guide the vehicle, a user interface has been designed for controlling the vehicle and selecting landmarks to be learned. Specifically, via the interface, the user controls the vehicle to navigate on the sidewalk, and move to an appropriate position with respect to each pre-selected landmark. Then, the features of the landmark are extracted from the 3D images acquired by one of the three KINECT devices using an SURF extraction algorithm. And the relative position between the vehicle and the landmark is computed by use of the depth image. Also, relevant information, including the camera number, the distance to the curb, the region of the detection window, and the vehicle parameters (the odometer readings), is recorded in the meantime.

Finally, we classify the recorded data into two categories, path-dependent data and landmark-dependent data. As soon as the learning process ends, the learned information is organized into a navigation path which is composed of several path

nodes with guidance parameters. All of the data are stored in the storages of the computer so that it can be modified and used repeatedly.

2.4.2 Navigation Process

Before the vehicle starts to navigate, the system reads the path and environment information created in the learning process as mentioned previously. In order to guide the vehicle to navigate along the learned path, the vehicle is instructed to move from a node to the next sequentially according to the learned path. A flowchart of the proposed vehicle navigation process is shown in Fig. 2.4.

In more detail, when the vehicle is navigating to the next node, it checks the navigation mode at first to ensure whether it has to detect the curb line and followed it. If the curb-line detection-and-following process fails, the system will enter *the blind navigation mode* and reconfirm the navigation mode in the next loop. Also, the navigation process detects the target landmark continually until the correct landmark appears in the omni-image. When the vehicle navigates to the desired node successfully, it can obtain the navigation information of the next node from the learned path kept in the system.

In addition, when the navigation process finds the target landmark successfully, the vehicle will adjust its position and load the relevant parameters for navigation to the next node. However, some nodes provide the navigation information only, which we call “tuning nodes.” This kind of node can help the vehicle to navigate to the terminal node successfully.

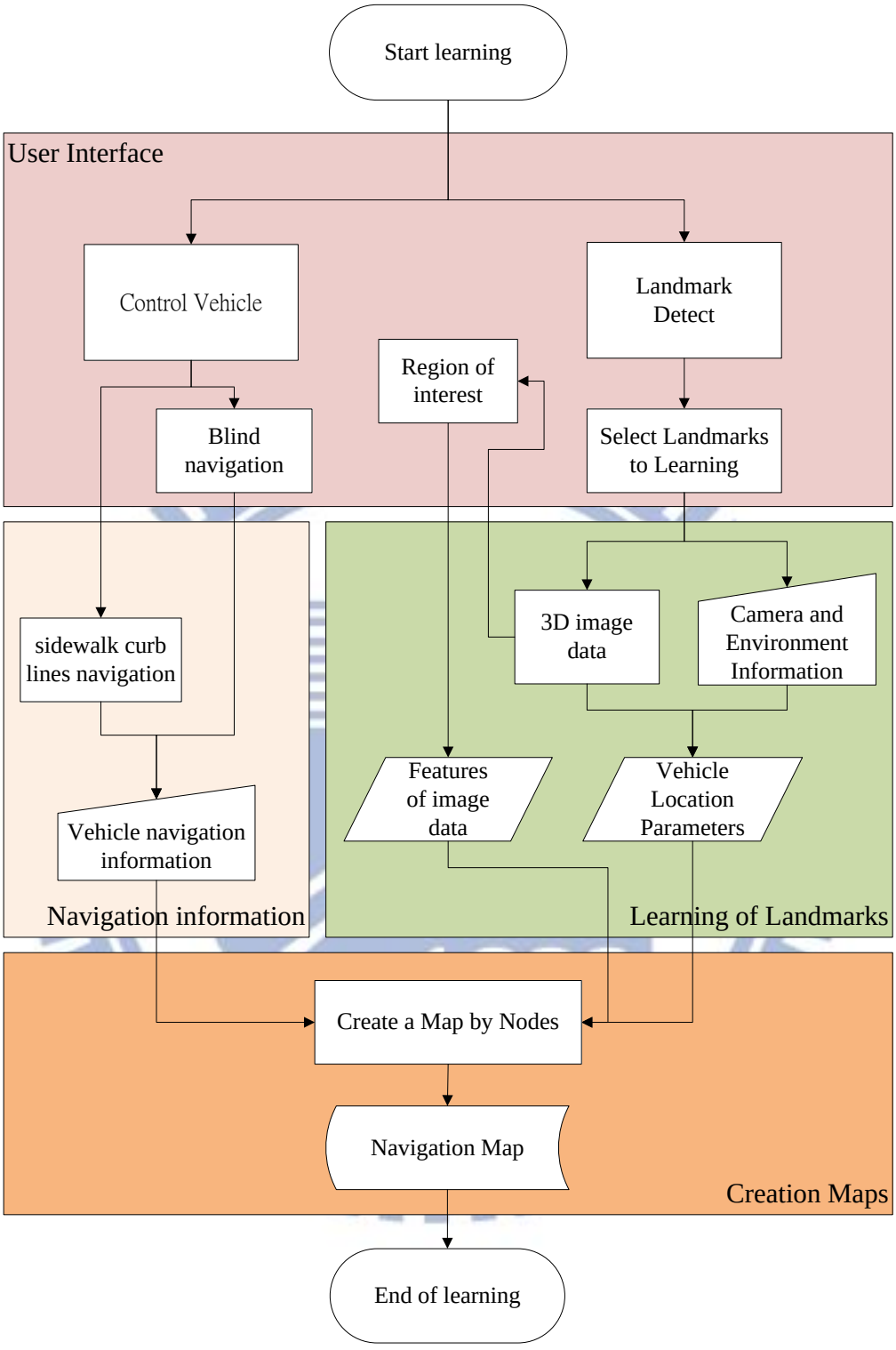


Figure 2.5 Flowchart of proposed learning process.

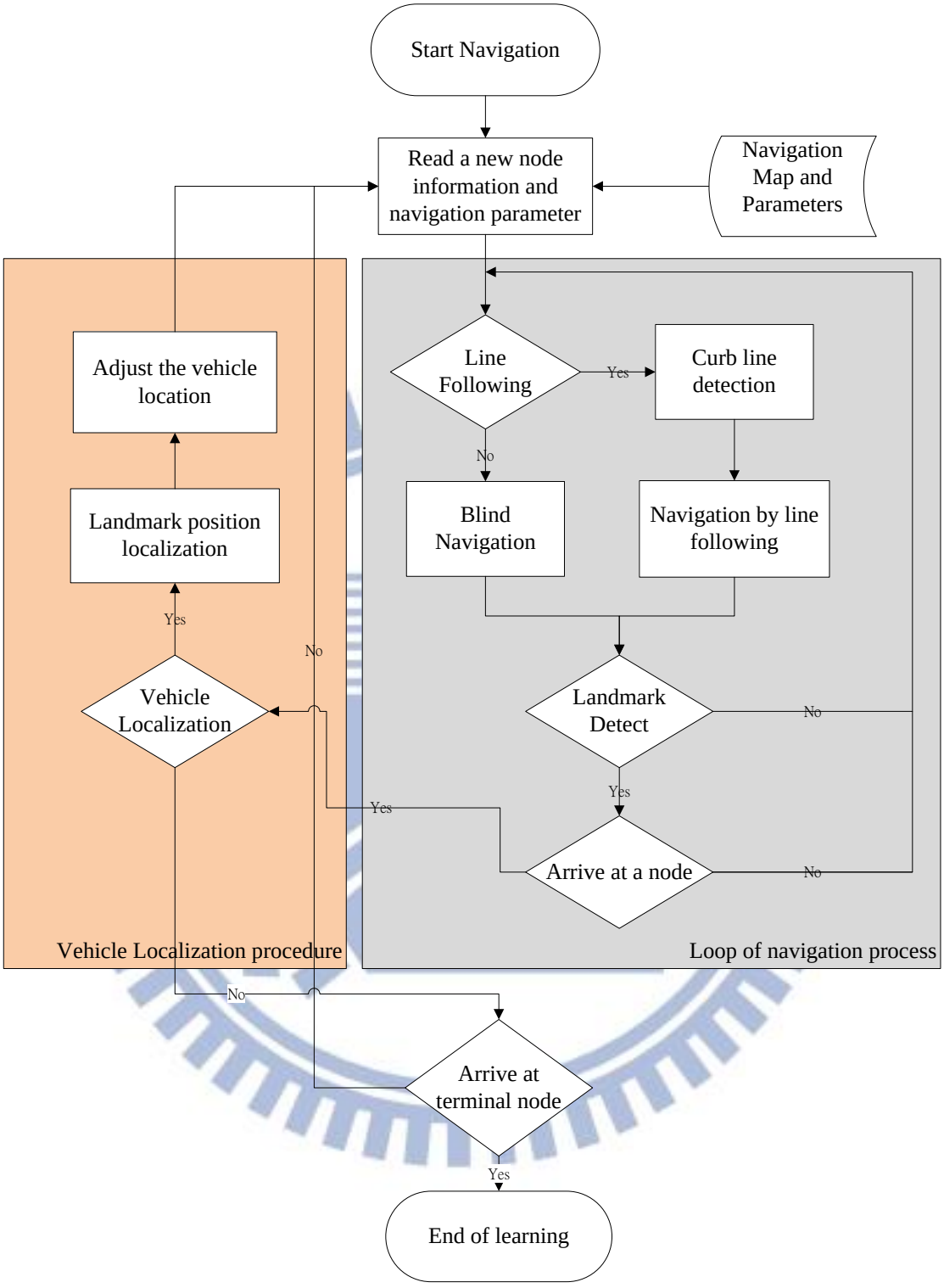


Figure 2.6 Flowchart of proposed navigation process.

Chapter 3
Learning of Outdoor Environment Features

3.1 Introduction

In order to use an autonomous vehicle to navigate in an outdoor environment, building complete path information to guide the vehicle is necessary. Therefore, creating a path map and selecting appropriate landmarks is a primary work for successful security patrolling by vehicle navigation. In this chapter, we will introduce our ideas of selection of landmarks and learning of guidance parameters in outdoor environments. Some coordinate systems, including the image coordinate system, the camera coordinate system, the vehicle coordinate system, and the global coordinate system, will be defined in Section 3.2. In addition, the learning techniques and strategies will be described in Section 3.3. At last, a detailed algorithm describing the learning process will be described in Section 3.4.

3.1.1 Selection of Sequential Landmarks for Learning

When we conduct the vehicle navigation process, mechanic errors will accumulate to affect to the readings of the odometer about the vehicle location and orientation. To solve such problems, we adopt an approach of "vehicle localization using landmarks." For this purpose, some objects should be selected as landmarks at first to conduct the vehicle localization task. In this study, we select some objects sequentially along the pre-selected path as landmarks. Because of this characteristic of

sequential selection, we can estimate the position of the vehicle on the sidewalk approximately without having to depend on using the odometer readings excessively. The main types of selected landmarks for localization in this study include light pole, hydrant, and tree trunk. Two other types of landmarks, namely, ramp and curb, which provide environment parameters for vehicle guidance are also selected.

With more and more categories of landmarks selected, we can utilize more information along the path for vehicle localization to reduce the chance of getting astray or falling out off the sidewalk, and guide the autonomous vehicle to the terminal point more reliably as well. The proposed methods of vehicle localization using landmarks will be described in detail later in Chapters 5 and 6.

3.1.2 Idea of Learning Guidance Parameters and Landmark Features in Outdoor Environments

To navigate in an unknown outdoor environment, some kinds of environment parameters or features should be learned for use in the navigation stage. The first feature learned in the proposed system is navigation path data. We can obtain the position of the vehicle by the odometer reading, but the mechanic errors usually cause imprecise readings of the vehicle location. Therefore, it becomes an important task to correct the position of the vehicle and the odometer reading. In Section 3.3.2, we will describe how to collect path data for vehicle localization by controlling the vehicle to navigate along a pre-selected path in an outdoor environment.

The features to be learned next are some camera and vehicle guidance parameters. Part of the parameters need manual measurements and are taken as inputs to the process of learning other features, and we refer to such types of feature data as "prior knowledge." More details of such parameters for learning will be introduced in

Sections 3.3.2 and 3.3.3.

The last feature to be learned is landmark. In order to use the landmarks to conduct vehicle localization, “training” the vehicle to “know” what to detect and how to recognize landmarks are necessary. That is, the vehicle must learn what features about each selected landmark should be detected, and then, it should be able to recognize each landmark by matching its features against those computed in the navigation phase. For this purpose, we adopt in this study a powerful approach — using the SURF [20] — to extract such features from selected-landmark images. In the mean time, we also record the vehicle location with respect to each selected landmark in terms of depth data. The detailed learning process is described in Section 3.3.4.

3.2 Coordinate Systems

In this section, we will introduce the coordinate systems used in this study, which describe the relations between the used devices and the selected landmarks in the navigation environment. The coordinate systems are illustrated in Figure 3.4 and defined 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 located at the center of the image plane.
2. Vehicle coordinate system (VCS) denoted as (V_X, V_Y): the origin O_V of the vehicle coordinate system is located at the center of the vehicle, and the V_X-V_Y plane coincides with the image plane.
3. Camera coordinate system (CCS), denoted as (X, Y, Z): the origin O_C of the CCS is located at the lens center of the KINECT device, the X-Z plane is parallel to the

ground, and the *Y*-axis is perpendicular to the ground.

4. Global coordinate system (GCS) denoted as (G_X, G_Y): the origin O_G of this system is always placed at the start position of the vehicle in the navigation path, and the G_X-G_Y plane coincides with the ground.

When we conduct the vehicle in the navigation phase, we have to know the relationships among the coordinate systems. At the beginning of each navigation process, the VCS and CCS follow the vehicle and the VCS coincides with the GCS. The coordinate systems are illustrated in Fig. 3.1.

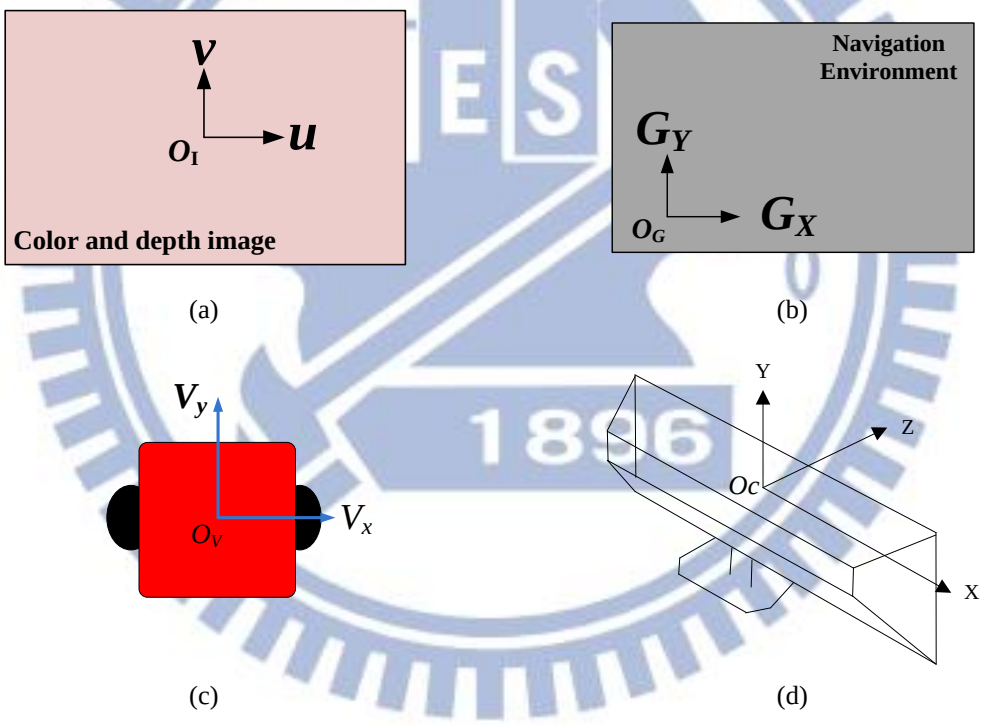


Figure 3.1 Four coordinate systems used in this study. (a) The ICS. (b) The GCS. (c) The VCS. (d) The CCS.

In this study, we use three KINECT devices equipped on the vehicle to sense the environment. When we bring the vehicle to a certain place where is a path node, the proposed system records the position data in the 3D space of the vehicle with respect to the selected landmarks. Then, when the vehicle moves on the path in the navigation

session, we can adjust the vehicle location according to the learned position at the currently-visited path node.

3.3 Learning of Outdoor Guidance Parameters and Landmark Features

3.3.1 Learning of Outdoor Guidance Parameters

For the vehicle to navigate in an outdoor environment, a trainer of the proposed vehicle system should guide the system to learn and record parameters or features of the environment. The parameters to be learned in this study include depth data, landmark feature, detection window, KINECT device number, and some other ground-truth parameters. The proposed techniques for learning these environment parameters are described in the following.

3.3.2 Learning of Navigation Paths Composed of Nodes

In general, the vehicle navigates in an outdoor environment under the control of a user. And at each visited path node, normally the proposed system will take the odometer reading as the position data of the vehicle. The position data consist of the vehicle coordinates (V_X, V_Y) and vehicle orientation θ in the VCS. We use these data to assist the vehicle system to conduct localization. Using the just-mentioned position data and the concept of sequential-node visiting to conduct vehicle localization is the main principle of vehicle guidance adopted in this study.

Specifically, we save the position data provided by the KINECT device, the

vehicle-turning parameters, and the vehicle coordinates (V_X, V_Y) as the data of a node Ni while the vehicle is in one of the following two situations:

1. the user controls the vehicle to learn a landmark object;
2. the user controls the vehicle to turn and record the turning parameters.

In addition to containing data items mentioned above, each node is labeled with a serial number. Such nodes then form a graph of the learned path. When the user controls the vehicle to move to a desired position, he/she will instruct the vehicle system to collect the node data semi-automatically. When the learning stage is finished, the system will have a set of nodes, denoted as $\boldsymbol{N_{path}}$. The process of recording the path data is described as an algorithm in the following.

Algorithm 3.1. ***Path node recording*.**

***Input*:** The 3D data provided by the KINECT device and the coordinates provided by the odometer.

***Output*:** A set of path nodes $\boldsymbol{N_{path}} = \{N_0, N_1, N_2, \ldots, N_t\}$.

Steps:

Step 1. Record the initial position of the vehicle $(x_0, y_0) = (0, 0)$ into the first node N_0 of the set $\boldsymbol{N_{path}}$ and mark the node as N_0.

Step 2. Create node N_i into the set $\boldsymbol{N_{path}}$, record the reading values of the odometer, (x_i, y_i), into N_i when the vehicle is in either of the following situations:

Step 2.1. the user controls the vehicle to learn an object of landmarks;

Step 2.2. the user controls the vehicle to turn and input the turning parameters.

Step 3. Repeat Step 2 until the learning process is finished.

Step 4. Create the terminal node N_t into the set $\boldsymbol{N_{path}}$.

Step 5. Save all the nodes of the set $\boldsymbol{N_{path}}$ into the computer and create a path map.

We show an illustration of the path in our experimental environment for this study in Fig. 3.2, which is part of the sidewalk of National Chiao Tung University. All of the nodes shown are recorded by the user. Each node is labeled with an index number according to the order of learning. The index numbers are useful for path map creation and landmark detection.

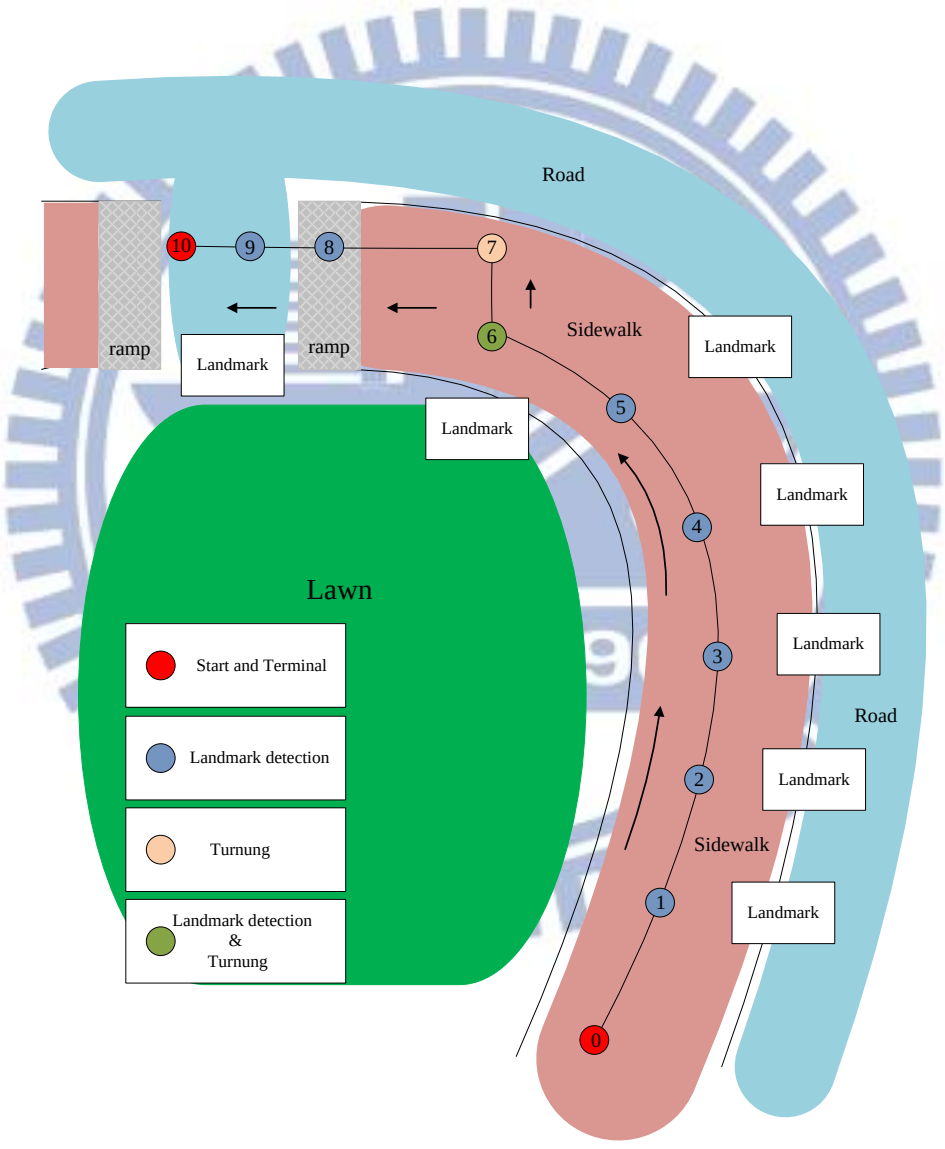


Figure 3.2 An illustration of the learned path nodes in the experimental environment for this study (part of the sidewalk in National Chiao Tung University).

3.3.3 Learning of Landmark Detection and Ground-truth Parameters

In this study, we use a technique of line following to navigate the vehicle along a path on the sidewalk which has a curb line along the path. Therefore, we can find that a line-segment landmark is usually projected in a fixed region in the image. For this characteristic, we only need to detect a part of the region in the image to reduce the computation time. Accordingly, we can define a region of interest (ROI) in the image as shown in Fig. 3.3, which is also called a *detection window*.

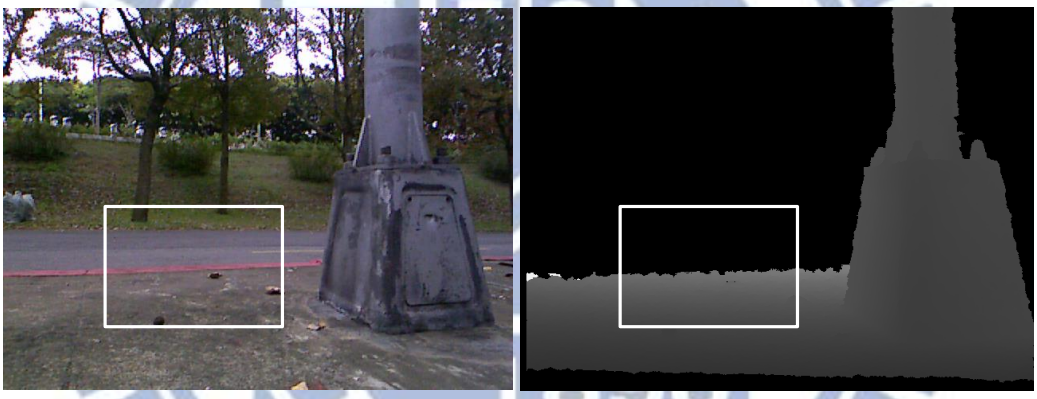
By this property, we also record which KINECT device is used to detect a certain landmark along the path. The KINECT devices are labeled with a serial number as shown in Fig. 3.4. When the vehicle moves on in the navigation stage, the recorded serial number in a path node can be retrieved to decide which KINECT device should be used to detect the target landmark *continuously* until the target landmark is detected. This means that the computation load in the navigation process is considerable. But after relevant parameters are learned, we can handle less data acquired by the specified KINECT device, and so do not have to use more than one KINECT device to detect the landmark at the same time unless we want. In this way, we can speed up the computation and so increase the navigation speed.

In addition, some ground-truth data are measured in the learning process, such as the angle of any ramp and the distance of the vehicle to the curb along the sidewalk. We will describe in Chapter 6 how these parameters are used in this study.

3.3.4 Learning of Landmark Features in Color and Depth Images

In order to learn selected landmarks, we design a user interface to help users to

specify the landmark which they want to use. While a user controls the vehicle to a position beside a landmark to be learned, he/she can select one of the KINECT devices to acquire the color and depth images, and then drag manually a rectangle as an ROI to segment out the landmark which appears in the color image. Next, a SURF extraction algorithm [20], which is described in Chapter 5 in detail, is applied to obtain the feature set of the ROI. Then, the depth data provided by the KINECT device, the feature set of the landmark, the KINECT device number, and the ROI are saved into the learned data set. A flowchart is illustrated in Fig. 3.5, and the details of the process are described in the following as an algorithm.



(a) (b)

Figure 3.3 Curb line in the detection window. (a) Color image. (b) Depth image.

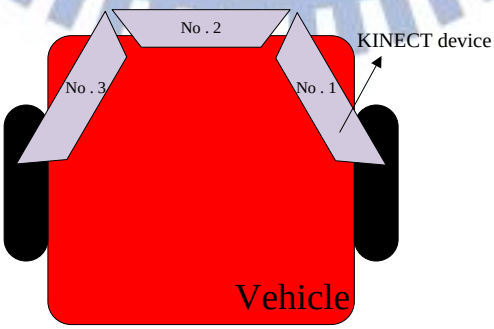


Figure 3.4 An illustration of KINECT device numbers.

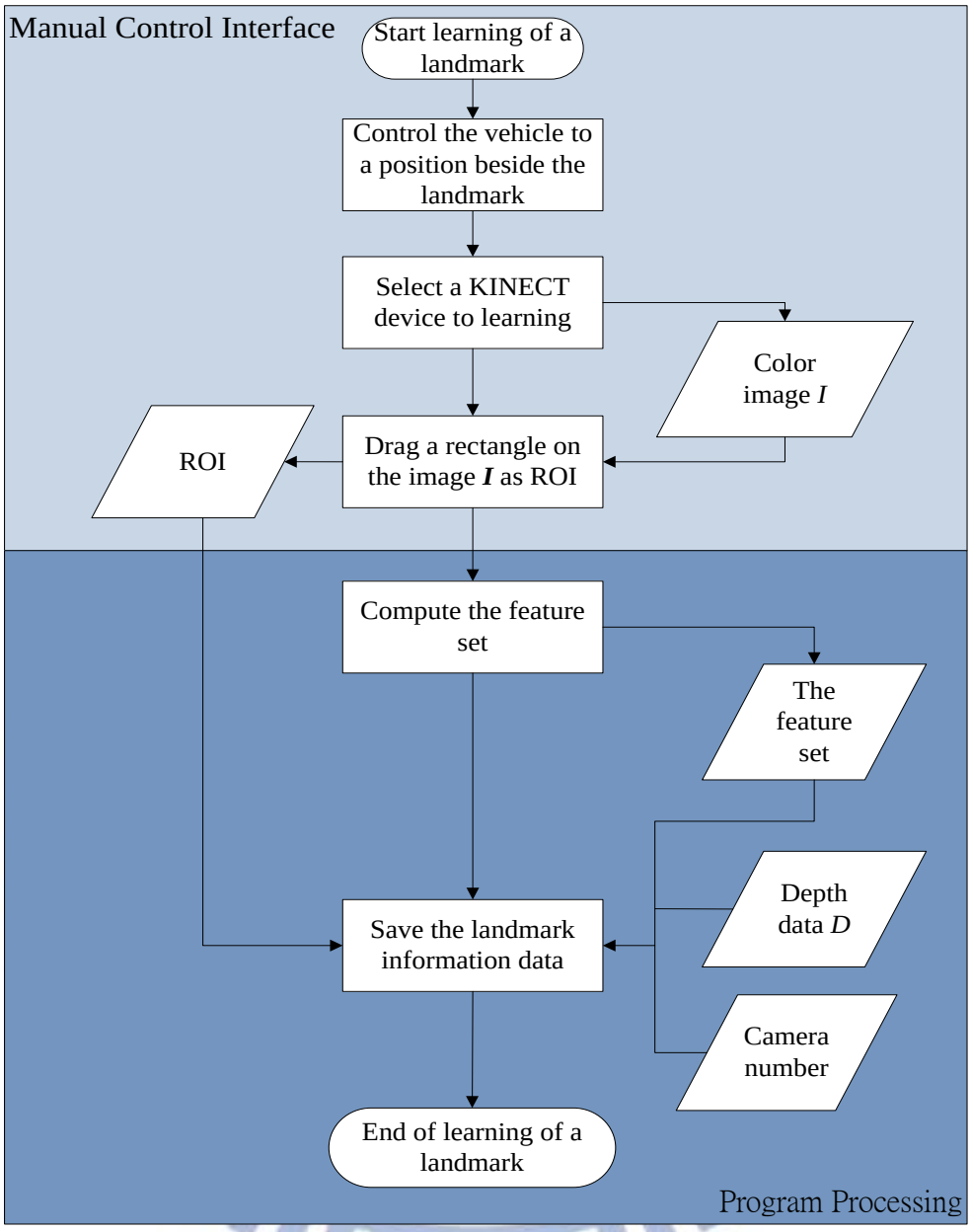


Figure 3.5 A flowchart of the landmark learning process.

Algorithm 3.2. *Learning of a selected landmark.*

***Input*:** the position *P* of a selected landmark *M*.

***Output*:** information data of landmark *M*.

***Steps*.**

Step 1. Control the vehicle to position *P* beside the landmark *M*.

Step 2. Select one of the KINECT device as specified in the path to acquire a color image *I* and a depth image *D*.

Step 3. Drag a rectangle on image *I* as an ROI *R*.

Step 4. Apply the SURF extraction algorithm on the ROI to extract a feature set *S*.

Step 5. Save the depth image *D*, the KINECT device number, the feature set *S*, and the ROI *R* manually in the record of the current path node corresponding to landmark *M*.

In this study, gray-level depth images composed of depth data provided by the KINECT device are used as inputs to the SURF extraction algorithm. An example of such depth images is shown in Fig. 3.6. Actually, the above algorithm of learning of a landmark is not suitable for such a type of depth image because the feature points in a gray-level image are much less than those in a color image. However, our experimental experience of using the depth image to extract SURF's for landmark localization shows that the effect of using the depth image alone is acceptable. More detailed experimental results and vehicle navigation schemes will be described in Chapter 6.

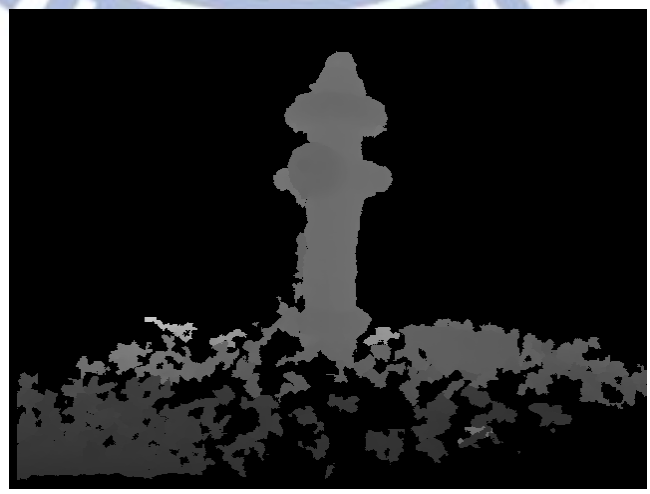


Figure 3.6 A hydrant landmark in a depth image.

Chapter 4
Navigation in Outdoor Environments

4.1 Introduction

When the learning process is finished, we can obtain the learned environment information, including a set of landmark features, ground-truth data, images of ROI, and a navigation path. In this chapter, we introduce our idea for vehicle navigation by this information in outdoor environments, and describe how we implement them. Some strategies for conducting the navigation work will be described in Section 4.2.1. In Section 4.3, the detailed algorithm for the proposed navigation process will be introduced after two main ideas to guide the vehicle to navigate on the learned path are described.

4.1.1 Strategy of Vehicle Guidance on Learned Paths

In the task of vehicle navigation, a navigation path like that shown Fig. 3.2 is established in advance. There are a starting point and an end one in the path, and also some spots of interest to us that the vehicle will go through between the starting point and the end one. In this study, we have chosen a starting point and an end one on an interesting path in a part of the sidewalk in National Chiao Tung University as our experimental environment, and record the features and positions of some pre-selected landmarks along the path. We have also “learned” some environment parameters, like the speed of the vehicle, the angle of each path turning, and the ground-truth data of a ramp and a curb segment, to assist the vehicle to navigate along the path successfully, as shown in Fig. 4.1. When the above-mentioned tasks are finished, the vehicle will

be said to be able to navigate along the *learned path*.

However, besides guiding the vehicle to learn the above-mentioned parameters, a vehicle navigation strategy is also important in this study. The strategy proposed in this study to conduct the navigation work is introduced in Section 4.2. The detailed algorithm for the proposed navigation process is introduced in Section 4.3.

(a) (b)

Figure 4.1 Two types of landmarks selected for use in this study. (a) Curb line. (b) Ramp.

4.1.2 Localization by Sequential Landmarks

As mentioned previously, the vehicle navigation process usually generates mechanic errors, resulting in imprecise computations of vehicle positions. To solve the problem, a strategy adopted in this study is to guide the vehicle to *constantly* localize its position based on the sequentially learned landmarks. Specifically after detecting and localizing a landmark in the acquired KINECT images by the use of the proposed methods (introduced later in Chapter 5) and obtaining the relative vehicle position with respect to the landmark, we can adjust the vehicle' position and orientation to the status as that learned in the learning phase at the current spot.

In addition, because the learned path is along a sidewalk and we use the concept of *sequential-node visiting* to conduct vehicle localization, the use of a curb line

feature on the sidewalk is practical in this study. We use the learned curb-line parameter to achieve line following to correct the vehicle's orientation for navigation along the learned path on the sidewalk.

4.2 Proposed Navigation Process

4.2.1 Strategies for Proposed Navigation Process

In this section, we introduce the strategies proposed in this study for vehicle navigation on the learned path. At first, the navigation process reads a learned navigation path and related guidance parameters which were recorded in the storage of the laptop computer. The navigation path consists of several nodes which were labeled in a *sequential* order in the learning process. The vehicle is guided according to the concept of sequential-node visiting to visit each node sequentially to conduct vehicle localization. Some strategies are proposed for use to guide the vehicle to navigate to the pre-selected destination successfully. They are described as follows.

1. The vehicle always follows the curb line on the sidewalk if possible. After detecting the curb line, the vehicle modifies its orientation to keep a safe distance with respect to the curb line on the sidewalk.
2. The vehicle localizes its position according to the learned sequential landmarks along the path. We adjust the vehicle's position in the GCS according to the *learned* landmark position and the *current* landmark position which are computed using the acquired images at the vehicle's current location.
3. An object detection process is conducted continuously to detect objects around. When an object of suspicion appears in the *detection window*, the vehicle will stop going forward, and match it against the recorded landmark.

By the above strategies, the vehicle can be expected to navigate to the desired destination successfully. A flowchart in accordance with the above three strategies is shown in Fig. 4.2.

4.2.2 Idea of Vehicle Localization by Learned Sequential Landmarks

Although the odometer readings provide the vehicle's position and direction for vehicle navigation in the navigation phase, they are usually imprecise to guide the vehicle to the next position correctly. Therefore, using the learned landmarks, which include light poles, hydrants, sidewalk curb lines, and tree trunks in this study, to localize the vehicle' position and orientation becomes the main task, as shown in Fig. 4.3. The sequential landmarks and the characteristic of the curb along a path on a sidewalk can be used to obtain the vehicle position on the path, and the learned odometer readings can assist judging whether the vehicle has arrived at a correct position or not to detect the landmark. The process of vehicle localization is illustrated in Fig. 4.4. Two different positions of the vehicle at a node in the navigation path and the relation between the vehicle, the curb, and the light pole are illustrated in Fig. 4.5.

The proposed vehicle localization technique consists of two major steps. Firstly, an object detection process detects the existence of the next-to-visit landmark continuously after the start of the navigation process. When the detection process detects a landmark at a correct node, we can acquire the landmark's depth data by the KINECT device with respect to the vehicle. Second, from the learned environment parameters, we can obtain the recorded depth data, and then we match, using the ICP algorithm, the two different sets of depth data to estimate the correct position and orientation of the vehicle according to the MSE criterion as illustrated in Fig. 4.6,

resulting in a set of 3D space parameters, including a pair of translation parameters (X_{mse}, Z_{mse}) and a rotation angle θ_{mse} in the CCS. The adopted technique to adjust the vehicle to a correct position is described in the following algorithm.

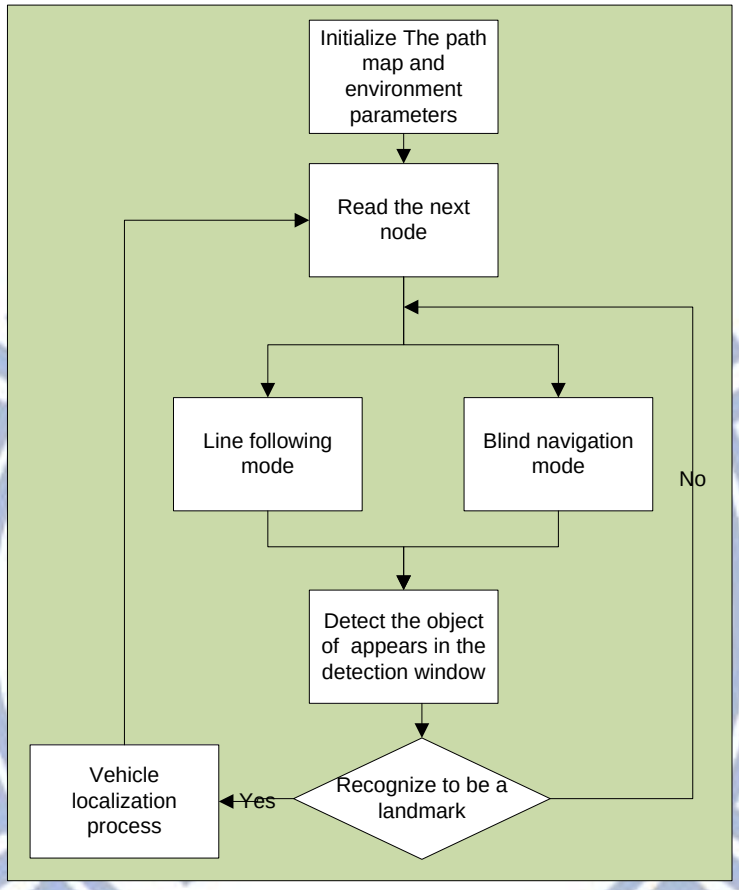


Figure 4.2 Flowchart of navigation process.

Algorithm 4.1 ***Vehicle localization and position adjustment by learned landmarks.***

Input: a color/depth image and a recoded landmark depth data *D*.

Output: None.

Step.

Step 1. Use the SURF extraction algorithm (described in Chapter 5) to recognize a learned landmark from the input color/depth image; and if an object of the learned landmark recognized successfully by system, take out the

corresponding node data from the learned path information; else, go to Step 1.

Step 2. Obtain at the vehicle's current position new depth data D' of the landmark by one of the three KINECT devices as specified in the learned path information.

Step 3. Convert the obtained depth data D' into 3D space coordinates (X, Y, Z) in the CCS using Equation (2.3).

Step 4. Compute the above-mentioned MSE estimation of the rotation angle θ_{mse} and the translation parameters (X_{mse}, Z_{mse}) between the converted coordinates and the recorded one in the path data in the CCS using the ICP technique (the detailed method will be described in Chapter 5).

Step 5. Convert the coordinates (X, Y, Z) in the CCS into the coordinates (V_X, V_Y) in the VCS by the following way:

$$V_X = X\,; \quad V_Y = Z\,. \tag{4.1}$$

Step 6. Use the estimated rotation angle θ_{mse} and the translation parameters (X_{mse}, Z_{mse}) to adjust the current vehicle position (V_X, V_Y) to the correct position (X_{adj}, Y_{adj}) in the GCS by the following equations.

$$\begin{bmatrix} X_{adj} \\ Y_{adj} \end{bmatrix} = \begin{bmatrix} X_{mse} \\ Z_{mse} \end{bmatrix} + \begin{bmatrix} \cos\theta_{mse} & \sin\theta_{mse} \\ -\sin\theta_{mse} & \cos\theta_{mse} \end{bmatrix} \begin{bmatrix} V_X \\ V_Y \end{bmatrix}. \tag{4.2}$$

At first, we define a region as the *detection window* and a threshold value *thr* for detecting landmark objects in the depth image, which are selected in the learning stage. Then, after the navigation process is started, the detection process will detect a region of *detection window* in the acquired depth image and decide whether there exists any object of concern. The criterion for this decision is to check if the distance between the detected object with respect to the vehicle is smaller than a pre-selected threshold *thr*. If this condition is satisfied, the SURF extraction algorithm is then

applied to extract the object's feature points in the *color* image, which then are matched with the learned feature set to recognize the learned landmark. The detection process is illustrated in Fig. 4.7.

(a) (b) (c)

Figure 4.3 Three types of landmarks selected for vehicle localization in this study. (a) Light pole. (b) Hydrant. (c) Curb line.

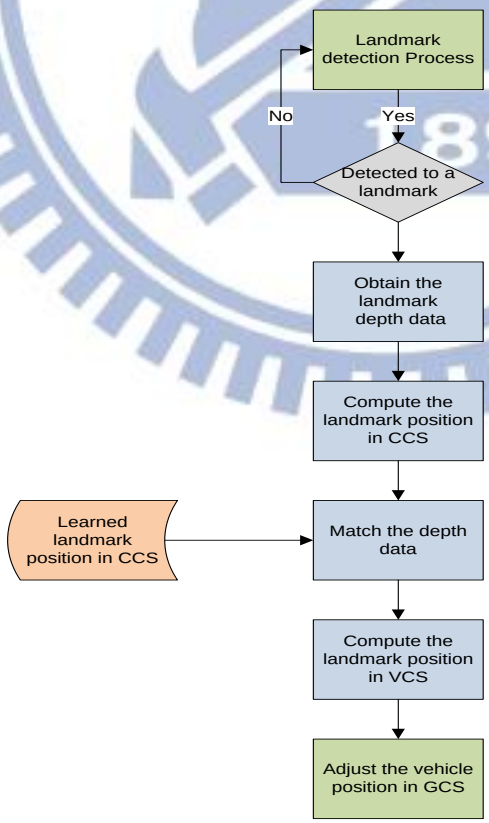


Figure 4.4 The vehicle localization process.

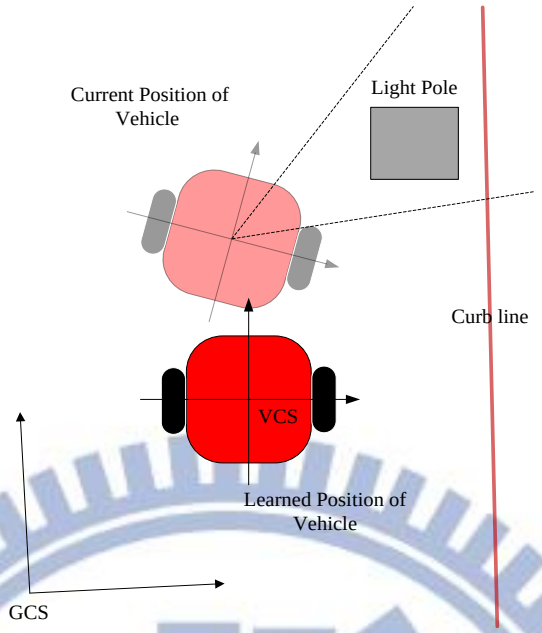
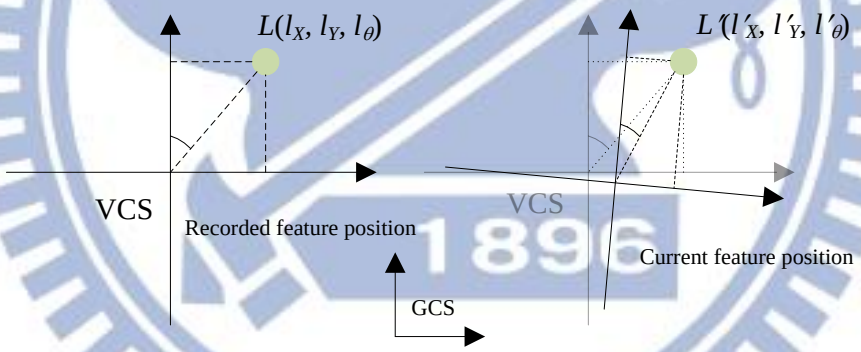


Figure 4.5 Illustration of learned position of the vehicle and current position of the vehicle in the GCS.



(a) (b)

Figure 4.6 The depth data of light pole recorded at position L are matched with newly-acquired depth data in navigation process at position L'.(a) A recorded feature position with respect to the vehicle. (b) A current feature position with respect to the vehicle.

4.3 Algorithm of Navigation in Outdoor Environments

In this section, we describe the detailed process for vehicle navigation in the

outdoor environment. With the learned information, the vehicle navigates along the learned path by the concept of *sequential-node visiting* to visit each recorded node consecutively and conducts specified works at the learned positions until reaching the end point of the learned path. The entire navigation process is described in the following algorithm, and a flowchart of the complete navigation process is shown in Fig. 4.8.

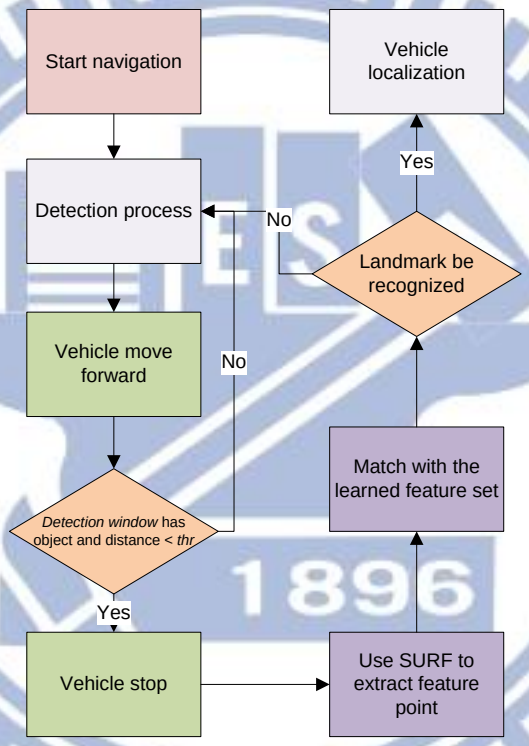


Figure 4.7 Flowchart of proposed detection process.

Algorithm 4.2 ***Navigation Process***.

Input: a learned navigation path N_{path} with relevant guidance parameters, and learned data of environment parameters.

Output: none.

Step.

Step 1. Choose a start node N_{start} and an end node N_{end} from the learned navigation

path N_{path}, and initialize vehicle navigation from N_{start}.

Step 2. Read from N_{path} a navigation node N_{next} and related guidance parameters.

Step 3. Move the vehicle forward to node N_{next} and detect the learned landmark.

Step 4. If a sidewalk following mode is adopted, detect the curb line by the curb line detection process (the detailed method will be described in Chapter 6). If successful, modify the vehicle direction accordingly; otherwise, conduct the vehicle in the blind navigation mode.

Step 5. If the detection process detects an object of concern in the *detection window* and its distance with respect to the vehicle is smaller than a threshold *thr*, then stop the vehicle and go to the next step; else, go to Step 7.

Step 6. If there exist a light pole or hydrant landmark in the current node N_{next}, capture a color/depth image by KINECT device, use the color/depth image and the learned landmark depth data *D* as inputs, perform the algorithm 4.1 to do vehicle localization, and then go to Step 2.

Step 7. If there exists a ramp landmark in the current node N_{next}, adopt the blind navigation mode, adjust the vehicle direction, and then go to Step 2.

Step 8. If there exists a tree trunk landmark in the current node N_{next}, compute the position of the landmark center in the image in terms of 3D space coordinates, use the coordinates to localize the vehicle, and then go to Step 2.

Step 9. If there exists a landmark which is pre-selected as the terminal node N_{end} (recognized to be so by its landmark type and its number), stop the vehicle, and finish the navigation.

Step 10. Repeat Steps 3 through 9.

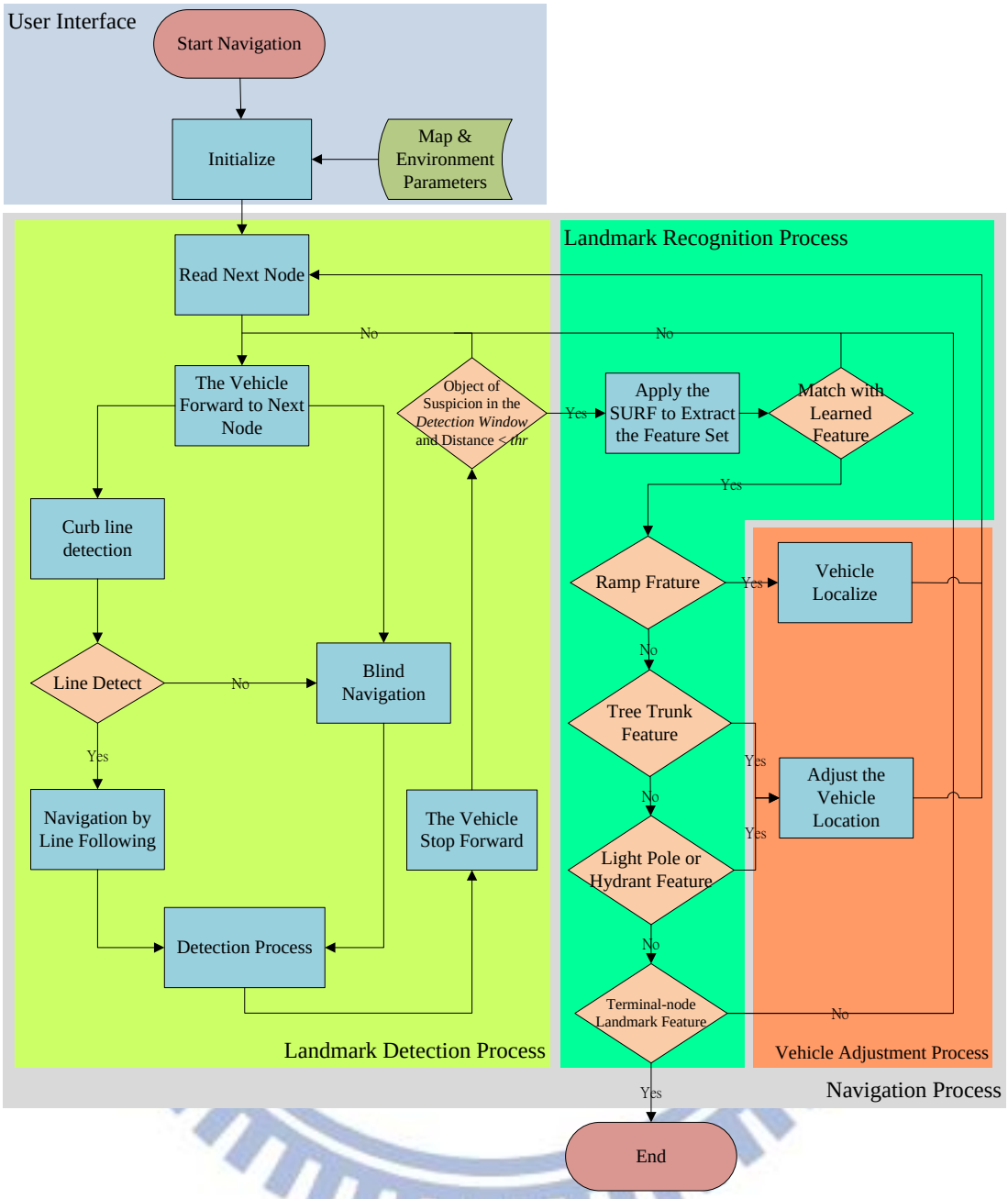
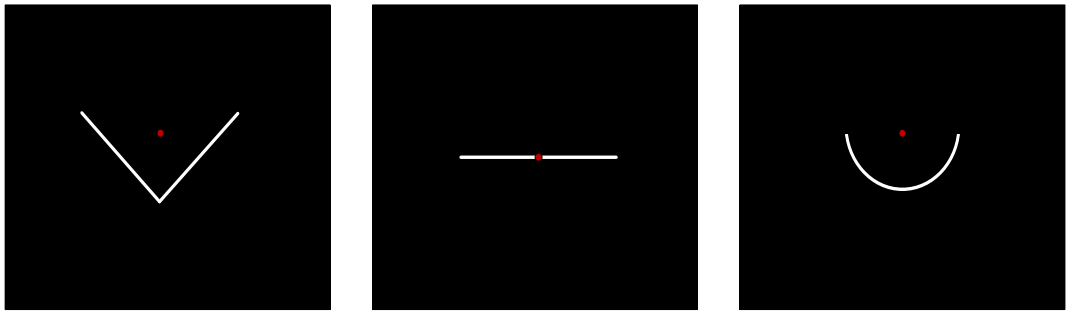


Figure 4.8 Flowchart of detailed proposed navigation process.

Chapter 5
Landmark Detection and Localization Using Depth and Color Images

5.1 Introduction

Vehicle localization is an important task for building the autonomous vehicle navigation system in this study. It can guide the vehicle move to a pre-selected destination successfully. For this purpose, we use pre-selected landmarks to provide the current vehicle position in the learned map when navigating. However, to decide which landmarks should be used is also an issue. Utilizing the characteristics of the KINECT device which can provide 3D space by depth images, we select objects which have prominent 3D shape information as landmarks for localization, as illustrated in Fig. 5.1. We choose rectangular-like objects as landmarks as Fig. 5.1(a), because it can provide translation and rotation information simultaneously. A method for feature extraction and matching for recognizing the landmark is described in Section 5.2. Unfortunately, not all of the landmarks on the learned path can provide 3D information. Therefore, some other techniques of object detection and localization will be introduced in Chapter 6. In addition, how we convert depth images of landmarks into 3D space coordinates to localize the vehicle will be described in Section 5.3. And a series of algorithms for landmark detection and localization will be described in details in Section 5.4.



(a) (b) (c)

Figure 5.1 The top views of three difference types of objects in the depth image which is captured from the front of the KINECT device. (a) A rectangle. (b) A plane. (c) A cylinder.

5.2 Review of Method of Matching by Speeded Up Robust Features (SURFs)

The SURF extraction method proposed by Herbert Bay et al. [20] in 2006 includes four major stages of computation to generate a set of features, and part of the idea is based on the similar concept of the SIFT [21]. In this section, we will give a brief review of the SURF, which is divided into two parts as follows: detection of feature points of interest and description and matching of such points.

5.2.1 Detection of Feature Points of Interest

A main difference between the SURF and the SIFT is that the SURF is based on the use of Hessian matrix approximation and integral images, which reduce the computation time drastically because the integral image allows fast computation of box-type convolution filters and the Hessian matrix has a good performance in accuracy.

In more detail, the theory of the SURF one pixel in an image $I(x, y)$ can be

represented by a Hessian matrix as follows:

$$H(I(x,y)) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix}, \tag{5.1}$$

and using the convolutions of the Gaussian second-order derivatives, the Hessian matrix $H(x, \sigma)$ at x at scale σ is defined as follows:

$$H(x,\sigma) = \begin{bmatrix} Lxx(x,\sigma) & Lxy(x,\sigma) \\ Lxy(x,\sigma) & Lyy(x,\sigma) \end{bmatrix}, \tag{5.2}$$

where $Lxx(x, \sigma)$ is the convolution of the Gaussian second-order derivative $\partial^2 / \partial x^2$ with of the image I at point x, and $Lxy(x, \sigma)$ and $Lyy(x, \sigma)$ are interpreted similarly.

In the choice of the filter, the author thinks that the filters are non-ideal in any case, so he chose to approximate the Hessian matrix with box filters. The 9×9 box filters in Fig. 5.2 for computing the blob response maps are denoted by *Dxx*, *Dyy*, and *Dxy*. Therefore, the determinant of the Hessian matrix can be written as:

$$Det(H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2. \tag{5.3}$$

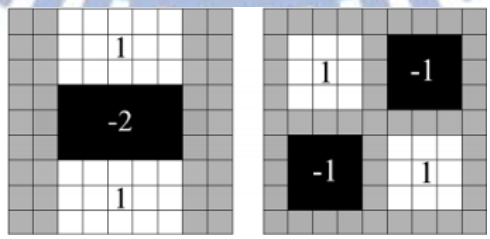


Figure 5.2 Left to right: The SURF used the approximation of the second-order Gaussian partial derivative in the *y*-direction (*Dyy*) and the *xy*-direction (*Dxy*). The grey regions are equal to zero.

The scale spaces are usually implemented as an image pyramid. The images are repeatedly smoothed with a Gaussian filter, and then sub-sampling in order to achieve

a high level of the pyramid. In the SIFT, the author subtracts these pyramid layers in order to get the DoG (Difference of Gaussians) images. In the SURF, the scale space *s* is analyzed by up-scaling the filter size rather than iteratively reducing the image size as shown in Fig. 5.3. Therefore, the SURF can reduce the sampling time to speed up the overall computation time.

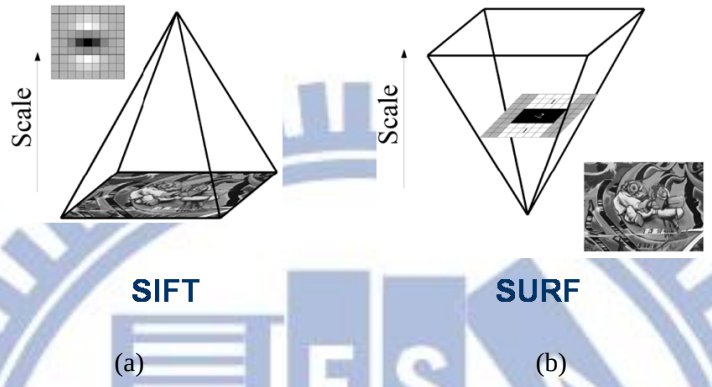


Figure 5.3 Illustration of SIFT and SURF. (a) Iteratively reducing the image size. (b) According to the scale space *s* to up-scaling the filter size.

A similar technique of the SIFT to localize the points of interest in the image is also used in the SURF extraction algorithm. Each point in the images is compared with its 8 neighbors in the same scale image, and the 9 corresponding neighbors in neighboring scale images, as shown in Fig. 5.4. If the point is a local maximum, it is selected as a candidate feature point. And the found candidate points of the determinant of the Hessian matrix are computed by 3D linear interpolation in the scale and image space.

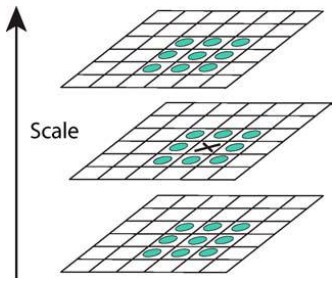


Figure 5.4 Maxima values are detected by comparing a pixel, as marked with **X**, with its 26 neighbors, as marked with the green circles, in 3×3 regions of the current and adjacent scales.

5.2.2 Description and Matching of Feature Points of Interest

In this section, we will introduce how the SURF feature extractor generates the feature descriptor and matches these features. The descriptor describes the distribution of the intensity content within the neighborhood of the point of interest, and is similar to the SIFT. The author builds the distribution of the first-order *Haar wavelet* responses in the *x* and *y* directions, rather than the gradient; exploits the integral image for speeding up; and uses only 64D. This reduces the time for feature computation and matching, and has proven to simultaneously increase the robustness. Furthermore, the author presents a new indexing step based on the sign of the *Laplacian*.

The first step consists of fixing a reproducible orientation based on information from a circular region around the feature point of interest. Then, a square region aligned to the selected orientation is constructed and the SURF descriptor is extracted from it. Finally, features are matched between two images. And the three steps are described in more detail in the following.

1. *Orientation assignment*

In order to be invariant to image rotation, the Gaussian weighted Haar wavelet responses in the *x* and *y* directions within a circular neighborhood of radius 6*s* around the interest point, with *s* the scale at which the interest point was detected. The responses are represented as points in a space with the horizontal response strength along the abscissa and the vertical response strength along the ordinate, and the dominant orientation is estimated by calculating the sum of the horizontal and vertical responses within a sliding orientation window of size $\pi/3$ as shown in Fig. 5.5. The

two summed responses then yield a local orientation vector.

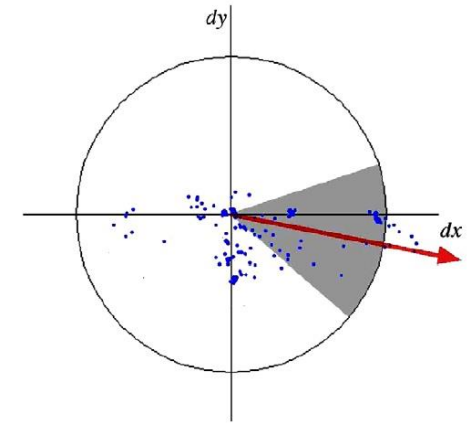


Figure 5.5 The dominant orientation of the Gaussian weighted Haar wavelet responses detected by the sliding orientation window.

2. Descriptor based on the sum of Haar wavelet responses

For the extraction of the descriptor, the first step consists of constructing a square-region centered around the points of interest and oriented along the orientation selected in the above-mentioned scheme. The region is split up regularly into smaller 4×4 square sub-regions. For each sub-region, we compute *Haar wavelet* responses at 5×5 spaced sample points. And the derivatives d_x and d_y are defined as *Haar wavelet* responses in the horizontal and vertical directions, respectively. Then, the wavelet responses d_x and d_y are summed up over each sub-region to form a first set of entries in the feature vector. In order to express the polarity of the intensity changes, it also extracts the sum of the absolute values of the responses $|d_x|$and $|d_y|$. Hence, each sub-region has a 4D descriptor vector $v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$ as illustrated in Fig. 5.6. Concatenating all of the 4×4 square sub-regions results in a descriptor vector of length 64.

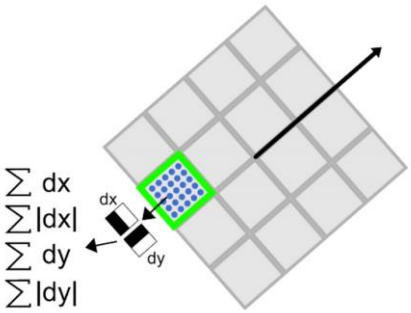


Figure 5.6 To build the descriptor, an oriented quadratic grid with 4×4 square sub-regions is laid over the point of interest. For each square, the wavelet responses are computed from 5×5 samples. For each field, the sums d_x, $|d_x|$, d_y and $|d_y|$, computed relatively to the orientation of the grid, are collected.

3. Fast indexing for matching

The matching technique in the SURF is only to compare features to see if they have the same type of contrast. Because the sign of the Laplacian distinguishes bright blobs on dark backgrounds from the reverse situation, as illustrated in Fig. 5.7, this feature is available at no extra computational cost as it was already computed during the detection phase.

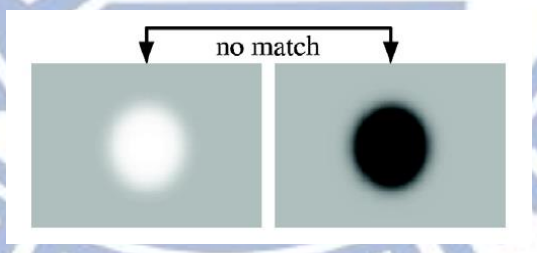


Figure 5.7 If the two types of contrasts between the two points of interest are different, it means that the candidate points do not match each other.

5.3 Vehicle Localization Using an Iterative Method

5.3.1 Conversion of Depth Information into 3D Space Coordinates

The depth data d_v provided by the infrared ray sensor equipped on the KINECT device are usually displayed as a depth image D on the monitor. The original depth data obtained by using the Kinect-for-Windows SDK are distance values and the range of them is from 800 to 4000 in the unit of millimeter. In order to display these values on a monitor, it is usually converted into a gray-level image using the following equation:

$$D(x, y) = \frac{d_v(x, y)}{16}, \tag{5.4}$$

where $d_v(x, y)$ is the distance value of a pixel at coordinates (x, y) and $D(x, y)$ is the computed result for d_v.

By Equation (5.4), the distance range will become 50 through 250. This shows that the depth values in the depth image have been compressed. Although the influence of this resolution reduction on the calculation result is not so much, yet we use the original depth data in this study and save them into a *2D array*. A depth image obtained from converting the depth data is shown in Fig. 5.8

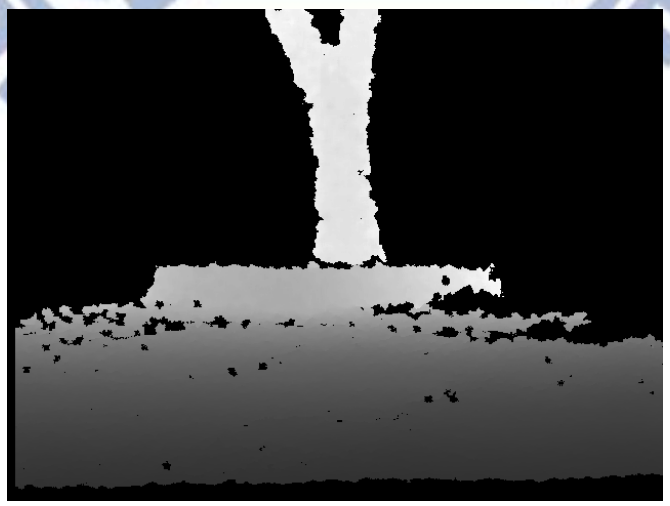


Figure 5.8 A landmark of tree appearing in a depth image.

In order to adjust the vehicle to the its correct position by the estimated rotation

and translation parameters in the GCS, we have to convert the depth data d_v into 3D coordinates (X, Y, Z) in the CCS, and then convert the coordinates (X, Y, Z) in the CCS into the VCS. For this, based on Equation (2.3), we convert all the depth data d_v by the following equations:

$$X(u,v) = \frac{d_v(u,v) \times (u - width/2)}{\sqrt{(u - width/2)^2 + (v - height/2)^2 + f^2}};$$

$$Y(u,v) = \frac{d_v(u,v) \times (v - height/2)}{\sqrt{(u - width/2)^2 + (v - height/2)^2 + f^2}}; \quad (5.5)$$

$$Z(u,v) = \frac{d_v(u,v) \times f}{\sqrt{(u - width/2)^2 + (v - height/2)^2 + f^2}},$$

where the values *width* and *height* specify the 2D array size; and the units of u and v are both pixel and those of $X(u, v)$, $Y(u, v)$, and $Z(u, v)$ are all millimeter. By Equations (5.5) above, we can obtain the 3D space coordinates in the CCS with respect to the center of the 2D array.

5.3.2 Localization by an Iterative Algorithm Using 3D Space Coordinates

According to the above discussions, we can obtain the 3D space coordinates of the landmark in the CCS. In order to localize the vehicle's position on the path, at the vehicle's current position, we capture new depth data d_v' of the landmark and match them against the learned data d_v to compute the MSE estimation of the vehicle *location*, including the rotation angle θ_{mse} and the translation parameters (X_{mse}, Z_{mse}), as described in Section 4.2.2 using the concept of ICP [23] technique.

Accordingly, we want to rotate and translate the depth data dv' of the landmark continuously by the concept of the ICP technique to find the correct vehicle location.

For this, we derive some formulas for use in the process. At first, assume that we have calibrated the tilt angle of the used KINECT device by the Kinect-for-Windows SDK before the navigation starts. Then, we want to rotate the depth data *dv'* of the landmark through the pan angle θ by a rotation matrix described as follows:

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.6}$$

also, we want to translate the depth data *dv'* of the landmark through a vector $[v_x, v_y, v_z]^T$ by a translation matrix as follows after representing the 3D space coordinates in the CCS by homogeneous coordinates (with an extra item of 1 in the fourth dimension):

$$T_v = \begin{bmatrix} 1 & 0 & 0 & v_x \\ 0 & 1 & 0 & v_y \\ 0 & 0 & 1 & v_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.7}$$

specifically, for a pixel *p* with 3D coordinates (p_x, p_y, p_z), the above translation results in:

$$T_v p = \begin{bmatrix} 1 & 0 & 0 & v_x \\ 0 & 1 & 0 & v_y \\ 0 & 0 & 1 & v_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} p_x + v_x \\ p_y + v_y \\ p_z + v_z \\ 1 \end{bmatrix}, \tag{5.8}$$

therefore, the translation result is $p + v$ if we represent $[v_x, v_y, v_z]^T$ by *v* and $[p_x, p_y, p_z]^T$ by *p*. Moreover, if we concatenate the rotation and the translation together, then if *P* is any point in the 3D space (say, on a landmark) with learned coordinates (*X*, *Y*, *Z*) and its current coordinates obtained from the KINECT images are (*X'*, *Y'*, *Z'*), then the

latter may be transformed into the former (i.e., from the current version to the learned one) by the following equation:

$$P(X,\ Y,\ Z,\ 1\ ) = P(X',\ Y',\ Z',\ 1) \cdot R_y(\theta) \cdot T_v p\ , \tag{5.9}$$

or equivalently, by the following equation:

$$P(X,\ Y,\ Z,\ 1) = P(X',\ Y',\ Z',\ 1) \begin{bmatrix} \cos\theta & 0 & \sin\theta & p_x + v_x \\ 0 & 1 & 0 & p_y + v_x \\ -\sin\theta & 0 & \cos\theta & p_z + v_x \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.9}$$

with

$$\begin{aligned} P(X) &= X'\cos\theta + Z'\sin\theta + p_x + v_x; \\ P(Y) &= Y' + p_y + v_y; \\ P(Z) &= -X'\sin\theta + Z'\cos\theta + p_z + v_z. \end{aligned} \tag{5.10}$$

With the above derived formulas, we now can use the concept of the ICP technique to obtain an angle θ_{mse} and a translation vector (X_{mse}, Z_{mse}) which minimize the mean square error (MSE), where the MSE value is computed by the following equation:

$$MSE = \frac{1}{N}\sum_{i=0}^{N} \left\| m_i - R_y(P_i) - T_v p_i \right\|^2, \tag{5.11}$$

where P_i (i = 0, 1, …, N) is a point of the depth data d_v', and m_i is a point in the learned data d_v *corresponding to* P_i which is computed in the sense of minimum Euclidean distance. Finally, a detailed description of the minimum MSE estimation using the ICP technique is as follows.

Algorithm 5.1: ***Minimum MSE estimation of the parameters for vehicle localization using the ICP technique.***

Input: the depth data d_v' of a landmark and the corresponding learned depth data d_v; a range for rotations between θ_{thl} to θ_{thr}, and a 2D range for translations between (X_{thl}, Z_{thl}) to (X_{thr}, Z_{thr}), all retrieved from the database.

Output: minimum MSE estimations of the rotation angle θ_{mse} and the translation vector (X_{mse}, Z_{mse}) of the vehicle in the CCS with respect to the landmark.

Steps:

Step 1. Convert the depth data d_v and d_v' respectively into the *learned* position coordinates (X, Y, Z) and the *current* position coordinates (X', Y', Z') in the CCS by Equation (5.5).

Step 2. For each point P_i with coordinate (X, Y, Z) in a 3D region A with θ from θ_{thl} to θ_{thr}, X from X_{thl} to X_{thr} and Z from Z_{thl} to Z_{thr}, do the following steps.

Step 2.1 Compute the Euclidean distances from P_i of dv' to every point P_j of d_v, and put them into a set M

Step 2.2 Find out a point P_{j_i} with the minimum Euclidean distance in M, take it as the one corresponding to P_i, and record the current data (θ_{j_i}, X_{j_i}, Z_{j_i}).

Step 2.3 Repeat Steps 2.1 and 2.2 until all points P_i in A are processed.

Step 3. Substitute all points P_i in A and their corresponding points P_{j_i} together with the recorded data (θ_{j_i}, X_{j_i}, Z_{j_i}) into Equation (5.11) to compute the MSE values and record them into a set K.

Step 4. Repeat Steps 2 and 3 until the end of the region A.

Step 5. Find the minimum of the MSE values, $k_i \in K$, and take out the corresponding values of θ, X, and Z in region A as the desired rotation angle θ_{mse} and translation vector (X_{mse}, Z_{mse}).

Step 6. Take θ_{mse} and (X_{mse}, Z_{mse}) as output.

5.4 Proposed Method for Light Pole Detection and Localization

5.4.1 Light Pole Detection Using SURFs

The use of a light pole as a landmark is a good choice for vehicle navigation because it is a commonly-seen object along the sidewalk. In addition, the light pole base is a solid rectangular-shaped object which is appropriate for object matching using the SURF extraction and matching algorithm.

To use the light pole as a landmark for vehicle localization, during the learning phase, we record a part in the color image as an ROI, and the feature set of the landmark as the learned data for matching. And in the navigation phase, at first, we apply the landmark detection process superimpose a region as a *detection window* on the acquired depth image and decide whether there exists any object of concern. If so and the distance with respect to the vehicle is smaller than a threshold *thr*, then we use the SURF extraction algorithm to extract the feature point of the object and match them with the recorded feature set. A detailed description of the landmark detection process is as follows.

Algorithm 5.2 ***landmark detection process*****.**

Input: a color image *I*, and a learned feature set read from database.

Output: a set of coordinates of feature points, *I*(*x*, *y*).

Steps:

Step 1. Conduct the landmark detection process.

Step 2. If an object appears in the *detection window* and the distance with respect to

the vehicle is smaller than a threshold *thr*, go to Step 3; else, go to Step 1.

Step 3. Use the SURF extraction algorithm to extract the feature points of the object in the color image *I*.

Step 4. Match the extracted feature points with the recorded ones using the computed contrast differences as described in Section 5.2.2.

Step 5. If the number of correct matching points is larger than a pre-selected threshold thr_2, go to Step 6; else, go to Step 1.

Step 6. Take the coordinates of the matching feature points $I(x, y)$ as output.

The above algorithm is a general one, and can be applied to detect and match light pole images. An example of feature extraction from landmarks is shown in Fig. 5.9(a), where the landmark is a light pole. We drag manually a rectangle as an ROI to segment out the landmark which appears in the color image as shown in Fig. 5.9(b). Next, we extracted feature points from the ROI image using the SURF extraction algorithm and recorded the descriptor of the feature points. The feature points in the ROI image shown in Fig. 5.9(b) are shown in Fig. 5.9(c), where the size of the circle specifies the degree of scaling and the line in the circle indicates the orientation of the feature point.

5.4.2 Light Pole Localization Using 3D Space Coordinates

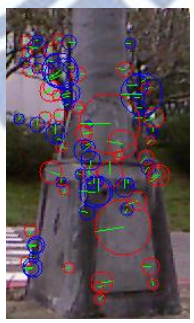
In this section, we will introduce the proposed method to localize the vehicle using the landmark of light pole. In order to show the advantage of the KINECT device, a series of schemes to combine color and depth images to achieve landmark detection and vehicle localization will be described.

The method of vehicle localization using the light pole landmark is similar to the

one described in Section 5.3.2, but the more detailed description about how to combine the coordinates of feature points in the ICS with the depth data to decide a range of threshold values for speeding up the computation of the ICP algorithm will described here. In the vehicle localization process, the ICP algorithm has a problem about the computation time. A large number of depth data used as input to the iteration algorithm will spend a considerable amount of computation time. Therefore, we use a range of threshold values and image sampling to reduce the number of calculated points for enhancing the computing speed. The detailed of deciding the threshold range is described as an algorithm as follows. An image resulting from using the range of threshold values to reduce the number of points in the depth image is shown in Fig. 5.10.

(a)

(b)



(c)

Figure 5.9 Feature extraction of a landmark of light pole image. (a) A used landmark in color image. (b) A ROI image of light pole base. (c) Feature points of ROI images.

Algorithm 5.3 ***decision of threshold range for reduction of image data.***

Input: the depth data d_v' and the coordinates of a set of feature points, $I(x, y)$, in the color image computed by SURF algorithm.

Output: a range of threshold values from thr_h to thr_l

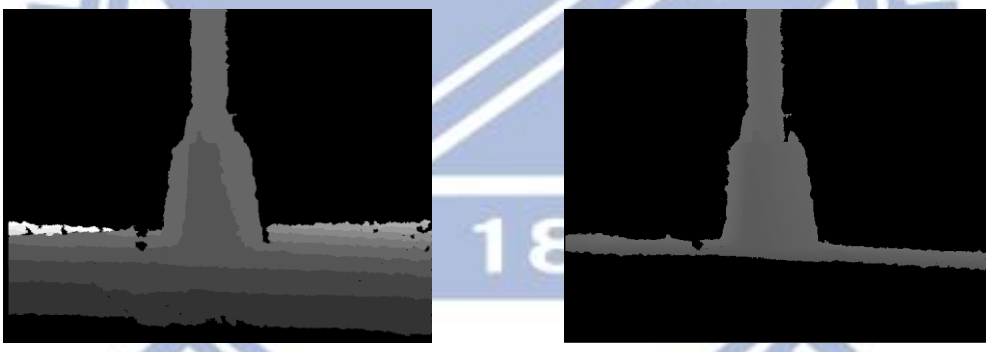
Steps:

Step 1. Take out all the coordinates of feature points $I(x, y)$.

Step 2. Use the technique provided by the SDK [19] to solve the shifting problem as shown in Fig. 5.11 by mapping the coordinates of the feature points $I(x, y)$ in the color image into the coordinates $d_v'(x, y)$ in the depth data d_v', and record the results.

Step 3. Compute the average value V_{avg} of the recorded depth data d_v'.

Step 4. Assign V_{avg} + 100 to thr_h and V_{avg} − 100 to thr_l and take the range from thr_l to thr_r as output.



(a) (b)

Figure 5.10 A landmark of light pole. (a) The landmark in the depth image. (b) Result of using a range of threshold values to reduce the number of data in the image.

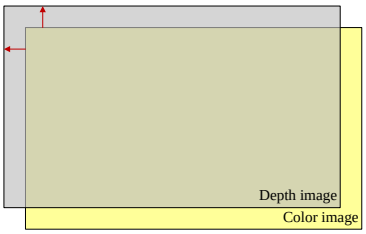


Figure 5.11 An illustration of the shifting problem of KINECT device (described in Chapter 2).

Then, we combine above-mentioned algorithm and the previously-described algorithm to form a complete algorithm for vehicle localization by the landmark of light pole.

Algorithm 5.4 ***vehicle localization by the detected landmark of light pole.***

Input: a captured color image I.

Output: none

Step 1. Using the color image I as input, perform Algorithm 5.2 to detect any object of concern in I with the output of a set of coordinates of feature points $I(x, y)$.

Step 2. Capture depth data d_v' using the KINECT device and use them together with $I(x, y)$ as inputs, perform Algorithm 5.3 to find a range R_{thr} of threshold values from thr_h to thr_l.

Step 3. Use R_{thr} to reduce the number of calculated points in depth data d_v', and denote the result as d_v''.

Step 4. Use d_v'' as input, perform Algorithm 5.1 to compute the minimum MSE estimation of the rotation angle θ_{mse} and the translation vector (X_{mse}, Z_{mse}) of the landmark with respect to the learned data of the landmark.

Step 5. Convert the depth data d_v into 3D space coordinates (X, Y, Z) in the CCS as done in Step 4 of Algorithm 5.1 according to Equation (5.5), and convert the resulting 3D space coordinates (X, Y, Z) in the CCS into the coordinates (V_X, V_Y) in the VCS according to Equation (4.1).

Step 6. Use the minimum MSE estimation θ_{mse} and (X_{mse}, Z_{mse}) to adjust the vehicle to the correct position by Equation (4.2).

5.4.3 Experimental Results for Light Pole Detection and Localization

Some experimental results for light pole detection are shown in this section. In one of our experiments, an object appears in the *detection window* in the depth data, as shown in Fig. 5.12. The color image corresponding to the depth data is shown in Fig. 5.13. And the extracted feature points of the captured color image are shown in Fig. 5.14. Finally, the matching result is shown in Fig. 5.15.

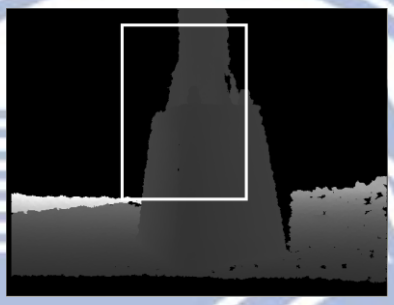


Figure 5.12 A landmark of light pole exists in detection window

Figure 5.13 A color image corresponding to the depth data.

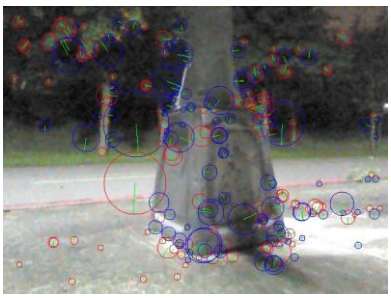


Figure 5.14 The extracted feature points in the captured color image.

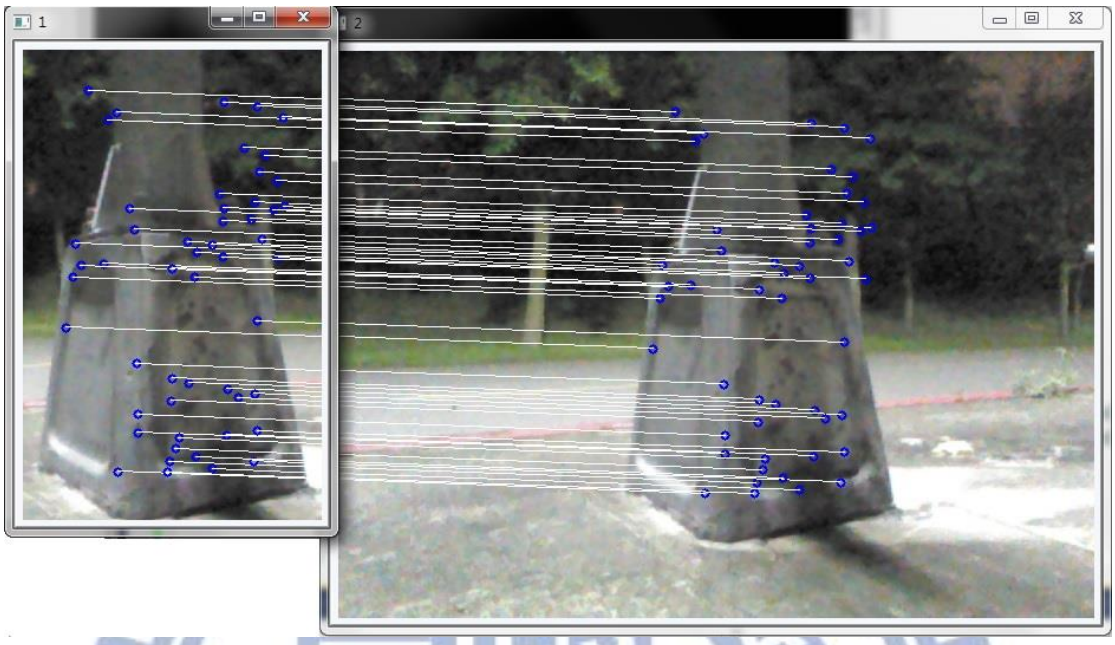


Figure 5.15 The matching result with the ROI image using the learned feature set.

Chapter 6
Landmark Detection Using Depth Information Only

6.1 Introduction

The KINECT device provides several different types of data, and what we use in this study are the color image and depth data. Somehow different from distances measured by other instruments, the depth data not only provides the distance value of the object in front of the KINECT device and within a range, but also form a depth image corresponding to the color image. The depth image can display the shape of the object and the entire background environment. This means that compared to the complexity of the color image, certain features of the object, like the shape, contour and position, can be obtained from the depth image more easily. Therefore, some specific objects of pre-selected landmarks were selected in this study for detection using the depth information (containing the depth image and other data) only.

By the use of the depth information, many types of landmarks can be detected and utilized for vehicle navigation. A technique of line following using the depth image proposed in this study is described first in Section 6.2, and the proposed method of using the SURF extraction algorithm and the depth image to detection objects of landmarks is described in Section 6.3. Then, a ramp detection technique is described in Section 6.4. Finally, we introduce the proposed tree trunk detection and localization technique in Section 6.5.

6.2 Proposed Technique for Curb Line Following

To conduct vehicle navigations on sidewalks, we propose a technique to detect curb lines and compute their distance with respect to the vehicle using the depth information. A technique of curb line detection is described in Section 6.2.1 and the entire algorithm is described in Section 6.2.2. At last, some experimental results for curb detection are shown in Section 6.2.3.

6.2.1 Extraction of Curb Boundaries in Depth Images

As described in Section 6.1, some types of landmark features in the depth information are more easily to obtain and calculate, and the boundary of the curb is one of them. The curb on the sidewalk usually has a height difference from the road, and so this feature in the depth image is obvious as can be seen in Fig 6.1. We use this characteristic to make it easier for us to extract the boundary of the curb.

(a) (b)

Figure 6.1 Two different perspective views of the curb on the sidewalk in the depth image acquired by a KINECT device. (a) A top-to-bottom view. (b) A farther view.

At first, we use the Canny edge detector to obtain the boundary of the curb in the

depth image. In the resulting edge-point image, the Hough transform was adopted to detect the straight line. After the curb line is found out, we retrieve a part of the line segment in the *detection window* to compute the distance of the curb line with respect to the vehicle. Then, the distance is utilized to adjust the direction of the vehicle to follow the curb line on the sidewalk. The proposed algorithm for curb line following is described in detail in the following section.

6.2.2 Algorithm of Curb Line Following

In order to guide the autonomous vehicle to navigate along a path on the sidewalk in the environment with a limited number of landmarks, we use the feature of curb line to conduct line-following navigation. An advantage of using the feature of curb line is that it always exists on the sidewalk along the learned path and provides the direction information. In addition, the vehicle is placed in parallel with the curb line before navigation. Therefore, if we can adjust the direction of the vehicle to be parallel with the curb line constantly when the vehicle moves forward in the navigation, it means that the vehicle will follow the curb line until it breaks or disappears. We can use the *learned* distance value of the curb with respect to the vehicle and the distance *computed* during navigation to "calibrate" the orientation of the vehicle because we assume the vehicle is always facing to the front and parallel to the curb. An illustration of this idea is given in Figure 6.2. And a corresponding algorithm is given as follows.

Algorithm 6.1 ***adjusting the direction of the vehicle by curb line following.***

Input: a depth image D and learned depth data d_v' of the curb feature, a detection window Win_{cb}, two angle threshold values Ang_l and Ang_h, and a distance threshold value Dis_{th}.

Output: none

Steps.

Step 1. Apply the Canny edge detector to the captured depth image D to obtain an edge-point image D_{edge} which includes the feature points of the boundary lines of the curb.

Step 2. Apply the Hough transform to image D_{edge} to detect sufficiently-long edge point sequences as a straight line which must be oriented within the angle range from Ang_l to Ang_h.

Step 3. Retrieve all the edge points on the straight line detected in the *detection window* from the cells of the Hough transform and denoted them as a set P_{sl}.

Step 4. Extract a set P_{dis} of depth values from the depth data d_v' by the following steps.

Step 4.1 Take out a point p in the set P_{sl}, find the corresponding point p' in the depth data d_v' whose coordinates are the same as those of p.

Step 4.2 Take out the depth value $d_{p'}$ of p' and put it into a set P_{dis}.

Step 4.3 Repeat Steps 4.1 and 4.2 until all the points in P_{sl} have been processed.

Step 5. Find the minimum value Dis_{min} in the set P_{dis} of distance values.

Step 6. According to Dis_{min} and the distance threshold Dis_{th}, adjust the direction of the vehicle to be parallel to the curb line according to the rules shown in Table 6.1, which is based on the principle of rotating the vehicle more if the forward-moving direction of the vehicle is more different from that of the curb line.

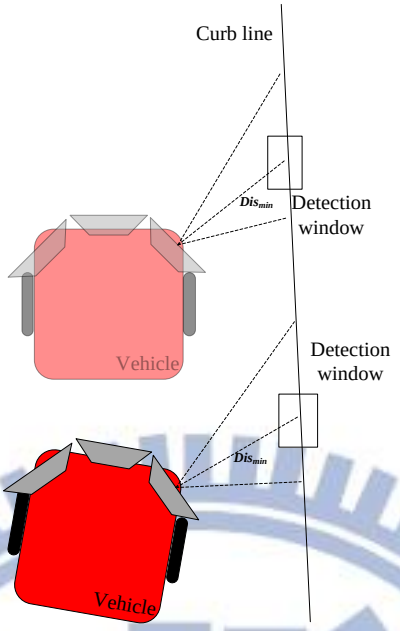


Figure 6.2 Use of the detected distances of the edge points of the curb line to adjust the direction of the vehicle

Table 6.1 The rules of rotation adjustment of the vehicle according to the distance difference value.

Result of $Dis_{min} - Dis_{th}$	Degree of right rotation	Result of $Dis_{min} - Dis_{th}$	Degree of left rotation
<=5	0°	>=-5	0°
>5 and <= 10	1°	<-5 and >=-10	1°
>10 and <= 15	2°	<-10 and >=-15	2°
>15 and <= 20	3°	<-15 and >=-20	3°
>20	5°	<-20	5°

6.2.3 Experimental Results of Curb Detection

Some experimental results of curb detection using the proposed curb line following process are given in this section. An input depth image with a curb line is shown in Fig. 6.3. The extracted curb boundary points from Fig. 6.3 are shown in Fig. 6.4. The result of detecting the curb line using the Hough transform is shown in Figure 6.5.

Figure 6.3 A curb line segment in a depth image.

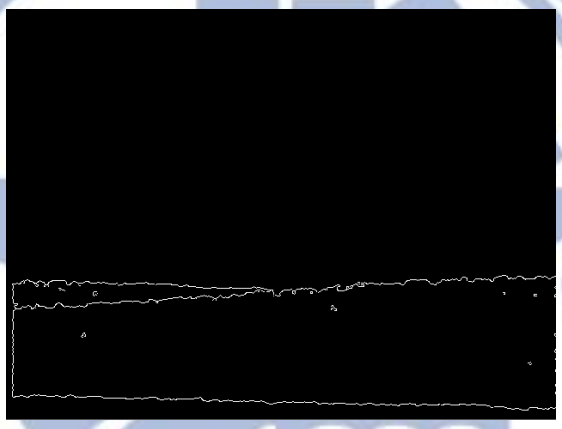


Figure 6.4 The curb boundary extracted by Canny detector

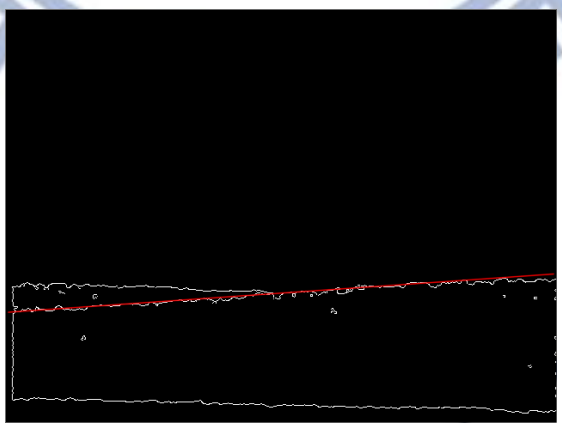
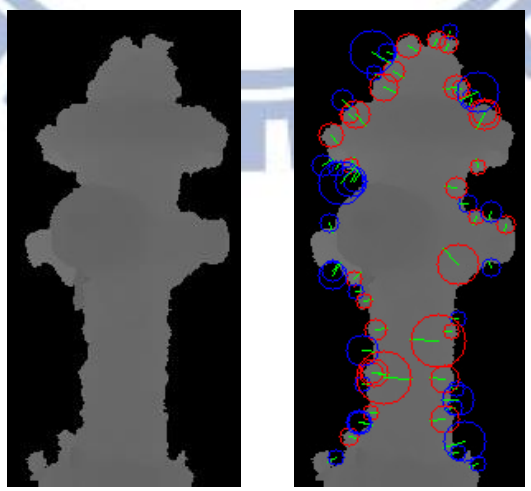


Figure 6.5 A curb line detection result by Hough transform

6.3 Proposed Method for Hydrant Detection and Localization

6.3.1 Hydrant Detection and Localization Using Depth Images

The hydrant is also a commonly-seen object on the sidewalk, and it has obvious color and shape so that we can easily recognize it. Therefore, in the task of autonomous vehicle navigation, we utilize the hydrant as a landmark and detect it to localize the vehicle position. The method for hydrant detection is the same as that for light pole detection, but a difference is that we use the depth image to conduct the SURF extraction and matching algorithm. An example of the SURF's extracted from the depth image of a hydrant landmark is shown in Fig. 6.6. We can see from the figure that the feature points are extracted mostly from the edges of the landmark shape, because the depth image does not have enough texture and differences of contrast other than the edges of the object.



(a) (b)

Figure 6.6 A landmark of hydrant. (a) The landmark in the depth image. (b) Extracted feature points by the SURF extraction algorithm.

In this case, in the object matching phase, the result of matching is susceptible due to possible interference from the background or ground because the captured depth image of the hydrant landmark includes the dilapidated ground information. This broken ground looks just like noise and creates impacts on the result of matching. Therefore, we slightly changed the technique of matching in this case. We use the learned environment parameters of the ground height to remove the ground information in the depth image before matching, and then use the pre-process technique of region growing to find the region of the object. In addition, in the matching technique we also compute the Euclidean distances of the matched points with those of the recorded of the object in the ROI image to confirm that the detected object is close enough. Some experimental results are shown in Section 6.4.2. The detailed algorithm of hydrant detection is described as follows.

Algorithm 6.3 ***detection of the hydrant landmark in the depth image using the SURF extraction algorithm.***

Input: a captured depth image d_v'; a learned feature set F of the hydrant landmark, a ROI image d_{roi}, a distance threshold value th_d, and the environment parameters of the ground height, all retrieved from the database.

Output: a set d_v' (x, y) of feature points in the depth image.

Steps.

Step 1. Remove the ground information in the depth image d_v' and obtain a new depth image d_v''.

Step 2. Use the technique of region growing to find the object of the hydrant and record the coordinates of the hydrant in the depth image d_v''.

Step 3. Use the SURF extraction algorithm to extract the feature points of the object

in the depth image d_v''.

Step 4. Match the set F' of the extracted feature points with the recorded feature set F using the computed contrast differences as described in Section 5.2.2.

Step 5. For each matched point $f_i'(x, y)$ in the depth image d_v'', do the following steps.

Step 5.1 Compute the Euclidean distance to each point $f_i(x, y)$ in the ROI image d_{roi}.

Step 5.2 If the computed Euclidean distance is smaller than a defined threshold th_d, then increment by 1 the value of a *counter* and record the matched point $f_i'(x, y)$ into a set K.

Step 6. If the *counter* is larger than a pre-selected threshold, then go to Step 7; else, quit.

Step 7. Take this set K of feature points together with their coordinates in the depth image as output.

After the hydrant landmark is detected, we conduct next vehicle localization using the landmark. But the technique of vehicle localization we use is the same as the light-pole landmark; therefore, we adapt the algorithm presented in Section 5.4 to be the following algorithm.

Algorithm 6.4 ***vehicle localization by the detected hydrant landmark.***

Input: none.

Output: none.

Step 1. If there exist a hydrant landmark in the current position according to the path information in the database, capture a depth image d_v' as input, and perform Algorithm 6.3 to yield a set feature points $d_v'(x, y)$ in the depth

image.

Step 2. Use the captured depth image d_v' and the set $d_v'(x, y)$ of feature points as input, perform Algorithm 5.3 to find a range R_{thr} of threshold values from thr_h to thr_l.

Step 3. Use R_{thr} to reduce the number of calculated points in the depth data d_v', and denote the result as d_v''.

Step 4. Use d_v'' as input, perform Algorithm 5.1 to compute the minimum MSE estimation of the rotation angle θ_{mse} and the translation vector (X_{mse}, Z_{mse}) of the landmark with respect to the learned data of the landmark.

Step 5. Convert the depth data d_v into 3D space coordinates (X, Y, Z) in the CCS as done in Step 5 of Algorithm 5.1 according to Equation (5.5), and convert the resulting 3D space coordinates (X, Y, Z) in the CCS into the coordinates (V_X, V_Y) in the VCS according to Equation (4.1).

Step 6. Use the minimum MSE estimation θ_{mse} and (X_{mse}, Z_{mse}) to adjust the vehicle to a correct position according to Equation (4.2).

6.3.2 Experimental Results for Hydrant Detection

Some experimental results of detecting a hydrant landmark using the SURF extraction method are given in this section. A depth image of the landmark with dilapidated ground is shown in Fig. 6.7. After we removed the ground information, the depth image becomes that shown in Fig. 6.8. The result of matching the detected hydrant against the learned data is shown in Fig. 6.9.

Figure 6.7 A depth image of a hydrant landmark with the ground information.

Figure 6.8 The ground information has been removed in the depth image of Figure 6.7.

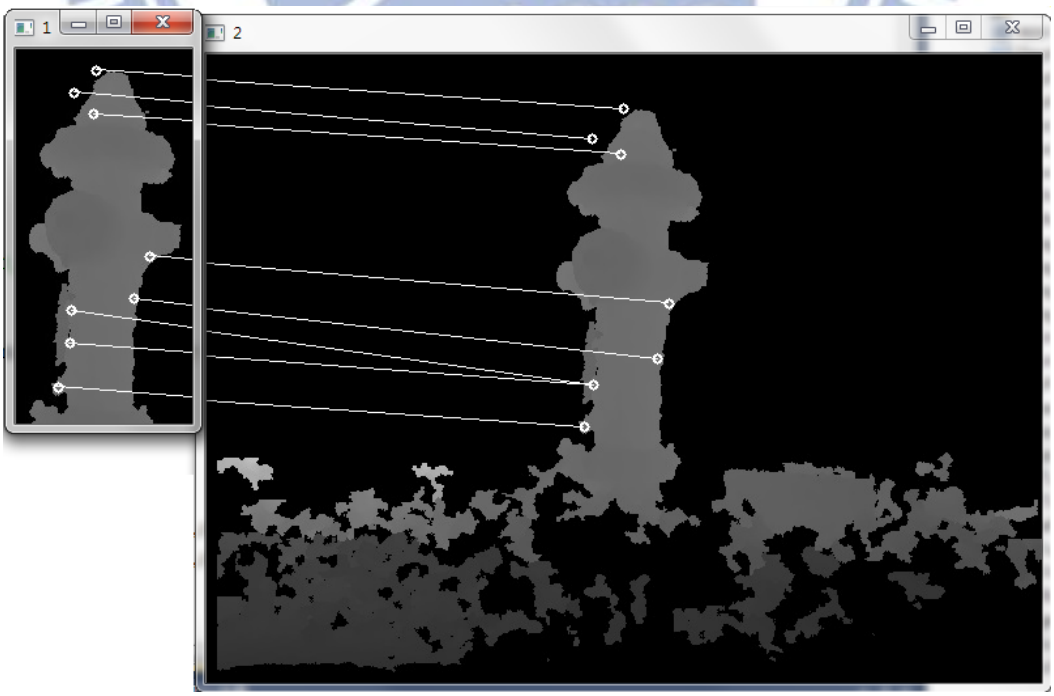


Figure 6.9 The matching result with the ROI image using the learned feature set.

6.4 Proposed Technique for Detection of Ramps in Depth Image

6.4.1 Review of Ramp Detection

In this section, we will introduce the technique of ramp detection proposed by Huang [24] using a KINECT device equipped on a car. The author detects the slope of the downhill ramp for driving safety. If the driver can know the slope of the ramp before driving downhill, then they are able to slow down earlier on the steep slope to reduce danger and improve driving safety. The author uses trigonometric functions and mathematical geometry to measure the slope of the ramp. An illustration of using the KINECT device to detect a ramp is shown as Fig. 6.10, where θ_3 represents the slope of the ramp. It can be easily figured out from trigonometry that

$$\theta_3 = \theta_1 - \theta_2, \tag{6.1}$$

where θ_1 is 30^o in the study of [24] and 0^o in ours (discussed later). According to trigonometry again, the author gets the following equation:

$$\theta_2 = \tan^{-1}(\frac{\Delta X}{\Delta Z}), \tag{6.2}$$

where ΔX and ΔZ can be computed by points $P(X_1, Z_1)$ and $Q(X_2, Z_2)$ which are selected from the centerline of the KINECT device on the ramp. In this way, the slope of the ramp is computed.

6.4.2 Algorithm of Ramp Detection

The proposed technique of ramp detection is based on Huang [24] to judge whether a vehicle navigates through a downhill/uphill ramp. In this study, the vehicle

occasionally will navigate through downhill ramps to touch the road (i.e., not on the sidewalk all the time). Therefore, we use this feature of ramp as a node to provide the information of the next node.

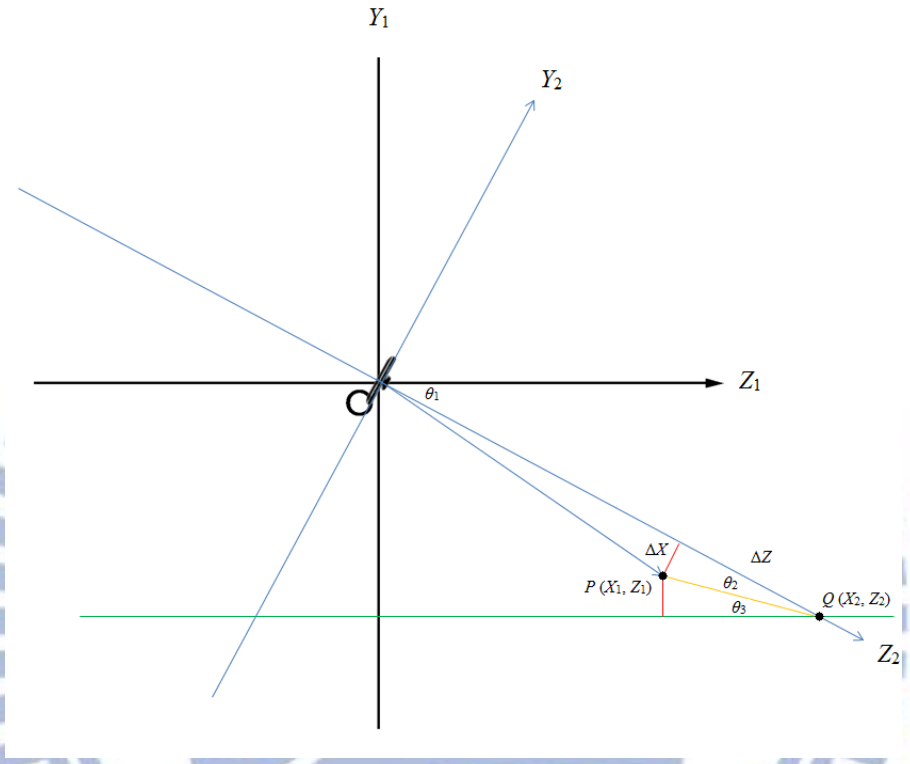


Figure 6.10 An illustration of geometry of slope computation [24].

The proposed ramp detection technique used in this study is illustrated in Fig. 6.11, where the KINECT device facing to the front is equipped on the vehicle and it is assumed that the Z-axes (Z_1) of the KINECT device is parallel to the ground G. Therefore, the angle θ_1 in the figure is 0 degree in this study. The downhill ramp in the depth image is as shown in Fig. 6.12. When the vehicle navigates on the sidewalk, we use Equation (6.2) to detect the slope continuously. The slope of the sidewalk should be computed to be 0 degree but we allow it to be ±1 degree to endure noise or other error sources. Then, when the vehicle detects a downhill ramp on the sidewalk, the computed slope will increase to a certain value. And when the vehicle navigates onto

the ramp, the computed slope will be become negative relative to the peak and increase linearly until the vehicle leave the ramp as shown in Fig. 6.13. We can use these two characteristics of slope changes to know where the vehicle is and whether it is navigating through a ramp. Finally, the above discussion about downhill ramp detection is described as an algorithm in the following.

Algorithm 6.2 ***downhill ramp detection process.***

Input: depth data d_v', and a range of slope values from θ_{down} to θ_{up} which impose limits on possible slope computation results.

Output: none.

Steps.

Step 1. Select the lowest pixel point P on the image boundary from the centerline of the depth data d_v'.

Step 2. Select the center point Q of the depth data d_v'.

Step 3. Obtain the distance values of point $P(X_1, Z_1)$ and $Q(X_1, Z_1)$ in the depth data d_v'.

Step 4. Use Equation (6.2) to compute θ_2.

Step 5. Substitute the value of θ_2 and the value $\theta_1 = 0^o$ into Equation (6.1) to compute θ_3.

Step 6. If $\theta_3 > \theta_{down}$ or $\theta_3 < \theta_{up}$, then move the vehicle forward and repeat Steps 1 through 5; else, go to Step 7.

Step 7. Record this information of θ_3 and check if two values of θ_3 have been recorded: if so, go to Step 8; else, move the vehicle forward and go to Step 1.

Step 8. If the computed $\theta_3 > \theta_{down}$, then determine that the vehicle has navigated through a downhill road; else, if $\theta_3 < \theta_{up}$, then determine that it has

navigated through a uphill road.

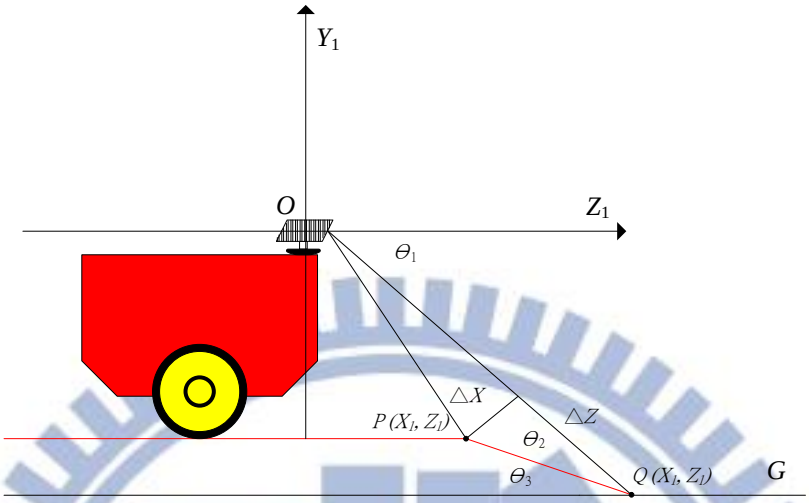


Figure 6.11 An illustration of geometry of slope computation by the KINECT device of face to front with equipped on the vehicle.

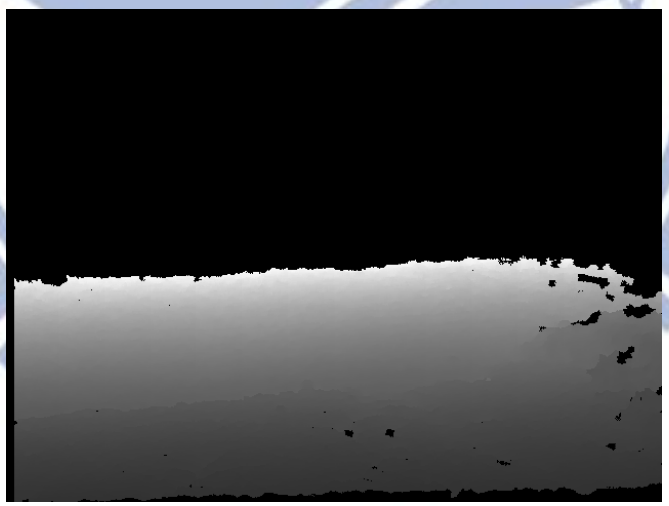


Figure 6.12 A downhill ramp in the depth image captured by the KINECT device facing the front.

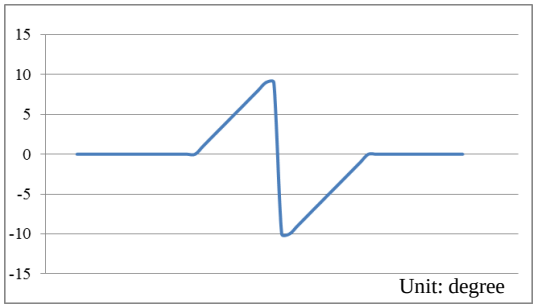


Figure 6.13 Illustration of computed slopes of a vehicle going through a downhill ramp.

6.5 Proposed Localization technique by Tree Trunks

6.5.1 Tree Trunks Detection and Localization

In this section we introduce the proposed technique for tree trunk detection and localization. As shown in Fig. 6.14, a tree trunk is surrounded by a curb on a road. At first, we detect the object in the detection window to check if a tree appears in the depth image. If so, we use the learned environment parameters of ground height to remove the ground in the depth image. Then, we use the Canny edge detector to obtain the boundary of the tree. In the resulting edge-point image, we use the Hough transform technique to find the vertical-line boundaries of the tree trunk. If we can detect the vertical-line boundaries of the tree trunk successfully, then we compute the center of the tree trunk in terms of the 3D space coordinate O(T_x, T_y, T_z) in the depth image by the Equation (5.5). Finally, we compare the computed coordinates of the center of the tree trunk with the learned ones to adjust the vehicle to the correct position. Detailed descriptions of tree trunk detection and localization using the proposed technique are presented as an algorithm as follows.

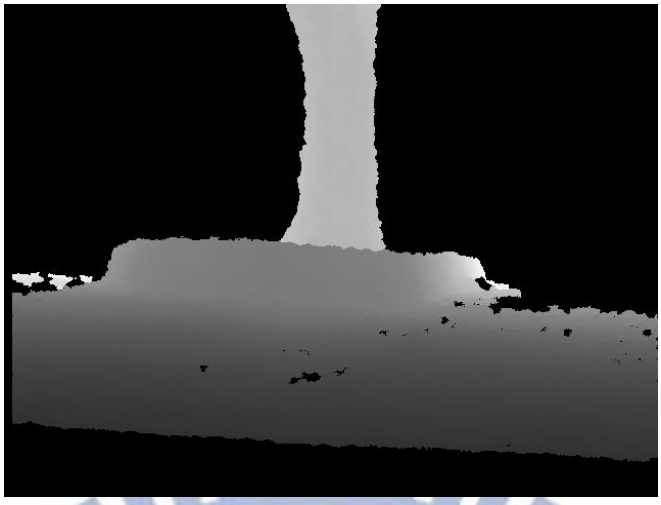


Figure 6.14 The landmark of a tree trunk in the depth image.

Algorithm 6.5: ***tree trunk detection and localization.***

Input: learned depth data d_v' of a tree trunk, learned 3D space coordinates O′(T_x, T_y, T_z) of the center of the tree trunk, a detection window Win_{tt}, a range of angle threshold values from Ang_l to Ang_h, and a distance threshold value Dis_{th}.

Output: none

Steps:

Step 1. Detect any object in the region of the *detection window*.

Step 2. If an object appears in the *detection window* and its distance with respect to the vehicle is smaller than a threshold Dis_{th}, go to Step 3; else, go to Step 1 to continue the object detection work.

Step 3. Remove the ground information in the depth data d_v' by the learned environment parameters of the ground height.

Step 4. Apply the Canny edge detector to the depth data d_v' to extract the feature points of the boundary lines of the tree trunk, and obtain an edge-point image D_{edge}.

Step 5. Use the Hough transform to detect the sufficiently-long straight line and

check if its orientation is within the range from Ang_l and Ang_h in the edge-point image D_{edge}: if so, go to Step 6; else, go to Step 1.

Step 6. Convert the depth data d_v' into the CCS with coordinates (X, Y, Z) in the CCS by Equation (5.5).

Step 7. Compute the center of the tree trunk in the depth image in terms of the 3D space coordinates, O(T_x, T_y, T_z), by the following equation:

$$O\left(T_x, T_y, T_z\right) = \frac{1}{N}\sum_{i=0}^{N} P_i(X, Y, Z) \tag{6.3}$$

where P_i (i = 0, 1, …, N) is a point of the depth data d_v' in the region of the detection window.

Step 8. Compute the translation parameters (X_{mse}, Z_{mse}) by the following equation:

$$X_{mse} = O(T_x) - O'(T_x),\ Z_{mse} = O(T_z) - O'(T_z) \tag{6.4}$$

where O′(T_x, T_y, T_z) are input data of the center of the tree trunk.

Step 9. Convert the coordinates (X, Y, Z) of the learned depth data in the CCS into the coordinates (V_X, V_Y) in the VCS by Equation (4.2).

Step 10. Adjust the vehicle to the correct position (X_{adj}, Y_{adj}) in the GCS by the following equations:

$$\begin{bmatrix} X_{adj} \\ Y_{adj} \end{bmatrix} = \begin{bmatrix} X_{mse} \\ Z_{mse} \end{bmatrix} + \begin{bmatrix} V_X \\ V_Y \end{bmatrix}. \tag{6.5}$$

6.5.2 Experimental Results of Tree Trunk Detection

Some experimental results for tree trunk detection and localization are shown in this section. The depth image resulting from removing the ground information and Canny edge detection is shown in Fig. 6.15. The result of vertical-line tree boundary detection using the Hough transform in the edge-point image is shown in Fig. 6.16.

The result of the computing the center of the tree trunk in the depth image is shown in Figure 6.17.

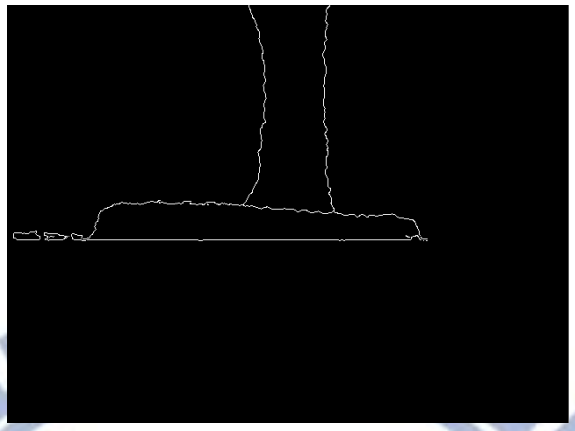


Figure 6.15 The depth image resulting from removing the ground and Canny edge detection.

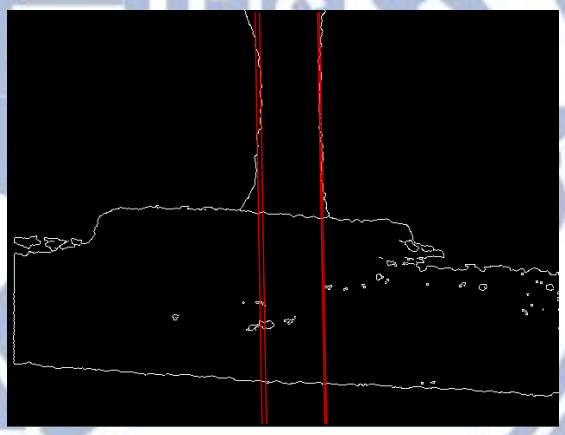


Figure 6.16 The result of tree boundary detection by the Hough transform.

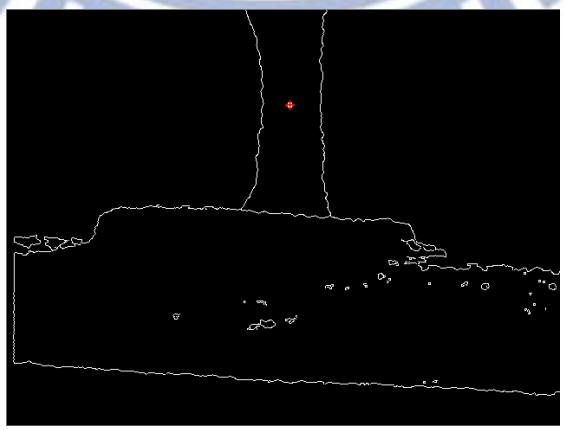


Figure 6.17 The detected center of the tree trunk.

Chapter 7
Experimental Results and Discussions

7.1 Experimental Results

In this section, we will show some experimental results of the proposed vehicle navigation system for video surveillance in the learning and navigation processes. The experimental environment was an outdoor sidewalk in National Chiao Tung University as shown in Figure 7.1.

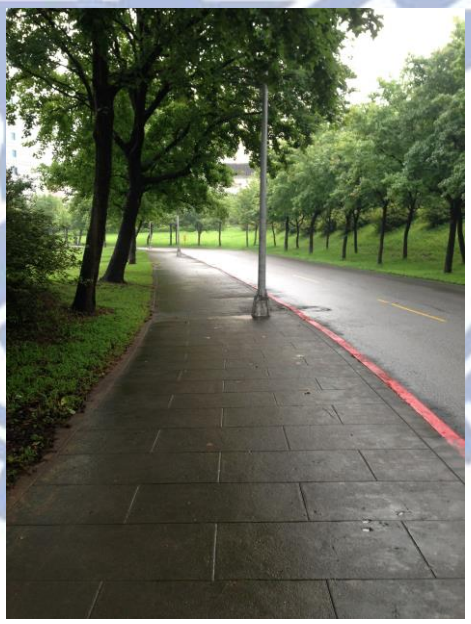
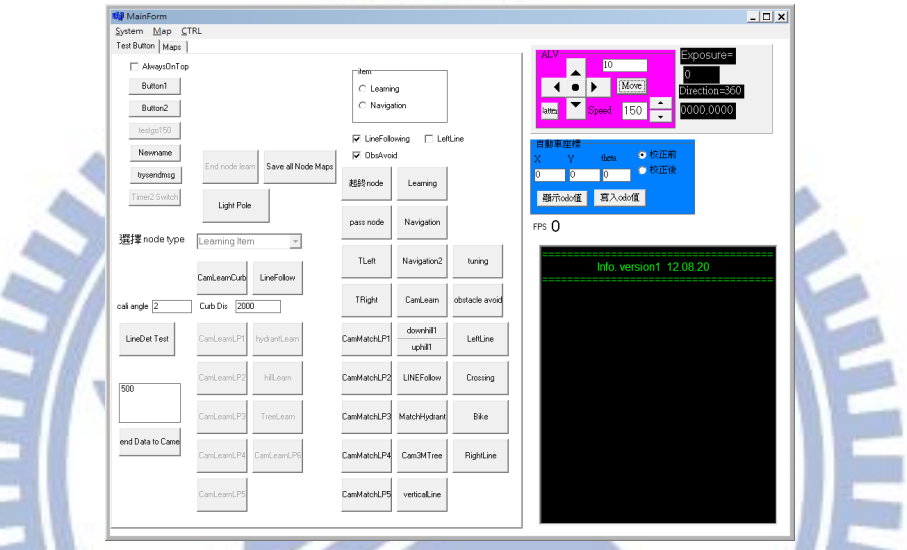


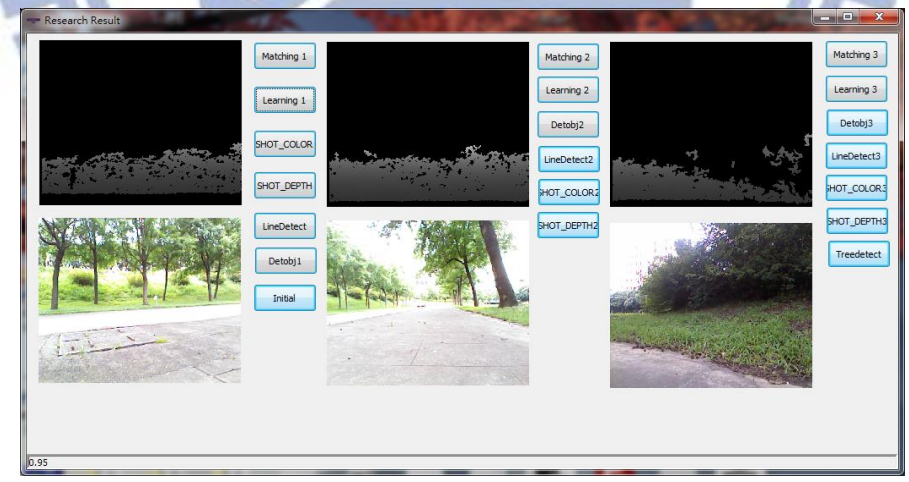
Figure 7.1 The experimental environment.

In the learning process, a trainer guided the vehicle by the use of a learning interface as shown in Figure 7.2 to construct a navigation path. The trainer navigated the vehicle to conduct learning tasks on the vehicle system along the path. After arriving at appropriate locations on the sidewalk, the vehicle was commanded to learn

the local positions and environment parameters of specific landmarks like light pole, hydrant, tree trunk, …, etc. In addition, the slope of the ramp and the distance between the vehicle and the curb were recorded manually as the ground truth by prior measurement. Finally, the trainer obtained a navigation map with a navigation path and multiple learned nodes of landmarks as illustrated in Figure 7.3.



(a)



(b)

Figure 7.2 The Learning interface of the proposed vehicle system. (a) Use of the Borland C++ Builder. (b) Use of the Visual Studio 2010.

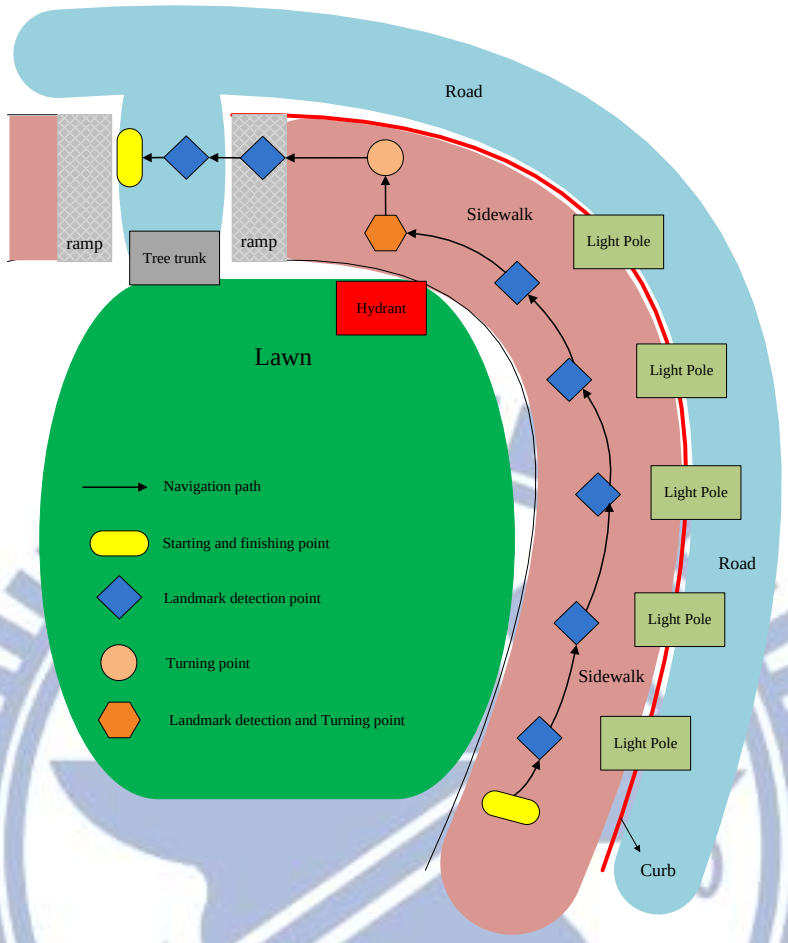


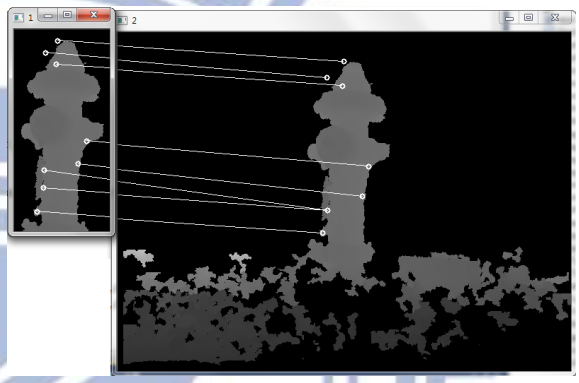
Figure 7.3 Illustration of the learned navigation path.

In the navigation process, the vehicle started from the same position just like in the learning process and navigated along the recorded navigation path nodes mainly with the curb line following technique. Then, the vehicle detected many types of landmarks and localized its position at pre-selected nodes. Some results of landmark detection are shown in Fig. 7.4. By conducting curb detection, the vehicle kept its path parallel to the curb. A result of curb detection is given in Fig. 7.5. Besides, the system is detected to the ramp of downhill in the navigation path and goes through it as shown in Fig. 7.6. Finally, the vehicle reached the appointed terminal node successfully is shown in Fig. 7.7.

(a)

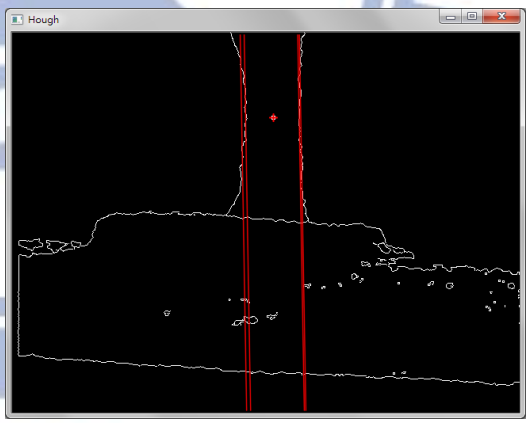
(b)

(c)



(d)

(e)



(f)

Figure 7.4 Some results of landmark detection. (a) The vehicle detects the landmark of light pole in the correct position. (b) The matching result of the light pole. (c) The vehicle detects the landmark of hydrant at the correct position. (d) The matching result of the hydrant. (e) The vehicle detects the landmark of tree trunk at the correct position. (f) The detection result of the tree trunk.

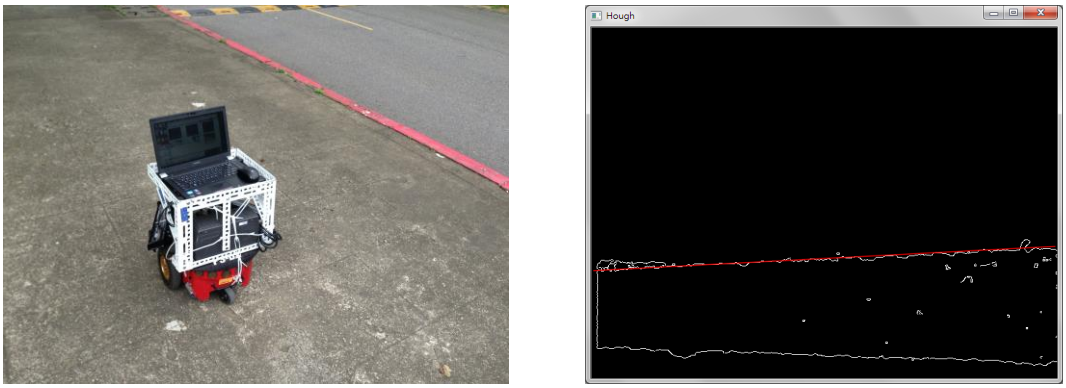
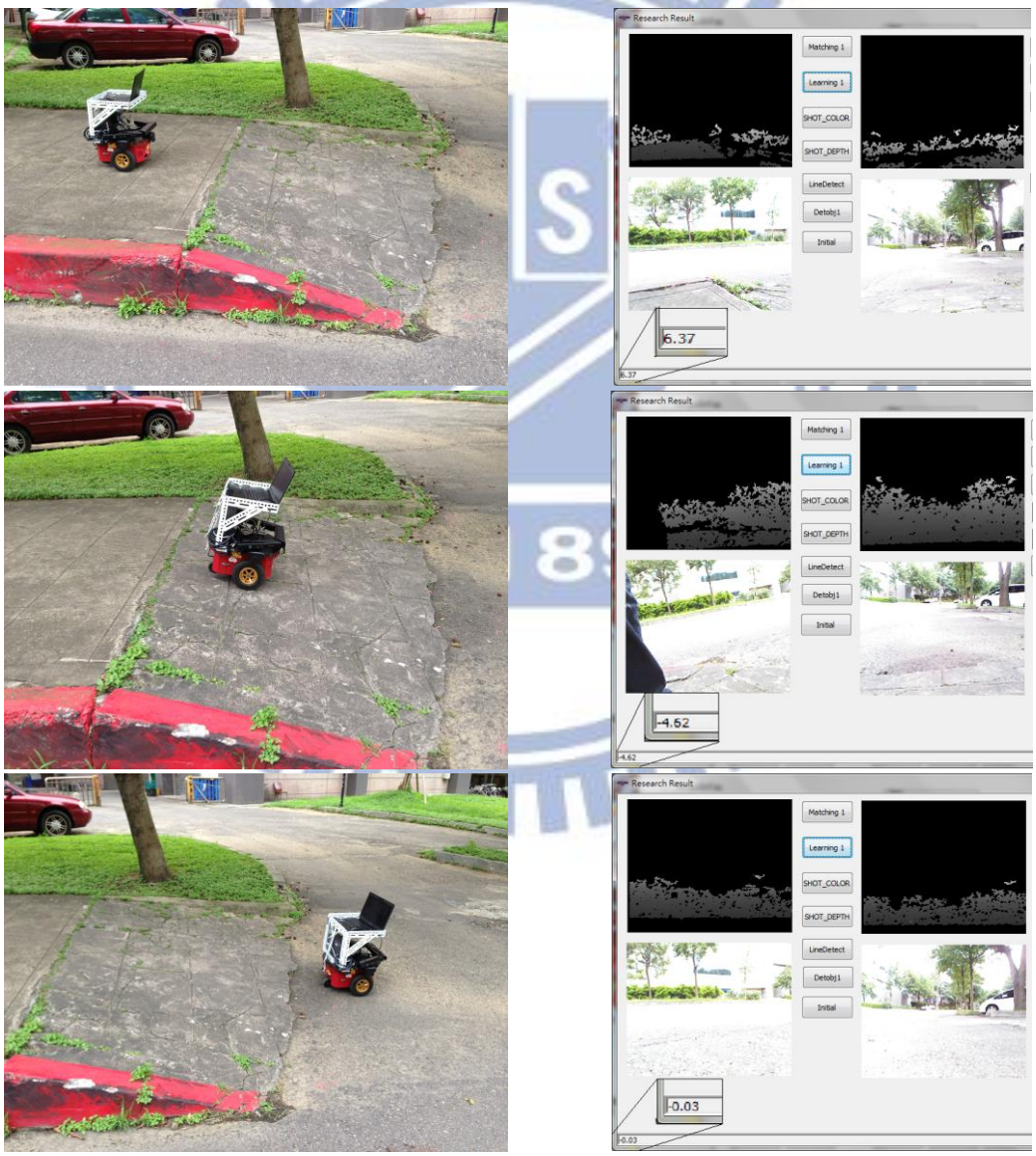


Figure 7.5 The result of curb line detection .(a) A vehicle on the sidewalk.(b) The detection result of the curb.



(a) (b)

Figure 7.6 The vehicle goes through a downhill ramp. (a) The vehicle consecutive positions on the ramp. (b) The computed results of the corresponding slopes of the ramp.

Figure 7.7 The vehicle navigates to an appointed terminal node successfully.

7.2 Discussions

By analyzing the experimental results of the vehicle navigation process, we found some problems. Firstly, for sidewalk curb detection, we only detect the curb in the campus of National Chiao Tung University within the depth image. If we can analyze the color image at the same time, it is believed that the proposed line following technique alone can be used for most environments. Next, in the landmark detection, the SURF extraction algorithm hopefully can be used to detect more types of landmarks in outdoor environments; maybe we can provide more kinds of landmarks for detection by the system. Furthermore, we spent much time to localize the vehicle position using the ICP algorithm; a solution to this issue is to build the data structure of KD-tree for use in the ICP algorithm to speed up the calculation. Finally, more experiments in different environments or continuation of our experimental path may also be conducted to test our system more thoroughly.

Chapter 8
Conclusions and Suggestions for Future Works

8.1 Conclusions

In this study, several techniques and strategies have been proposed and integrated into an autonomous vehicle system for security patrolling in outdoor environments with capabilities of specific object detection, vehicle localization, and adjustment of the navigation path, using three KINECT devices affixed on the vehicle.

At first, by the use of a learning interface designed in this study, a trainer can guide the vehicle to navigate on a sidewalk and construct a navigation path conveniently, including the path nodes, alone-path landmarks, and relevant guidance parameters. However, there are some ground-truth parameters which need prior manual measurements.

Next, a strategy of vehicle navigation for security patrolling with a line following capability has been proposed. The vehicle navigates according to the node data of the path map which is created in the learning phase, and detects along-path landmarks by the SURF extraction algorithm and matches them with the learned data. The matching technique is based on the measures of contrast differences and Euclidean distances using the color image and/or the depth image acquired by the KINECT device.

In the use of color image, the matching technique is only to compare features to

see if they have the same type of contrast. And in the use of the depth image, in addition to the above technique, the Euclidean distance between the matched points is also computed.

Finally, a vehicle location estimation technique by utilizing the depth information and the concept of ICP algorithm has been proposed. The newly captured depth data and the learned ones are used to estimate the vehicle location. The estimation result is obtained through continuous iterations to obtain the minimum MSE estimation value. Also, a ramp detection technique has been adopted and a curb line detection technique has been proposed, both for use to guide the vehicle on a safe path as well as to adjust the vehicle orientation.

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

8.2 Suggestions for Future Works

The proposed strategies and techniques, as mentioned previously, have been implemented on a vehicle system. Based on our experience of the experiments, several suggestions and related interesting issues worth further investigation in the future are stated as follows.

(1) It is interesting to develop an *adaptive* method to calibrate automatically the three KINECT devices equipped on the vehicle.

(2) It is worth to investigate more kinds of features for vehicle location estimation on the sidewalk.

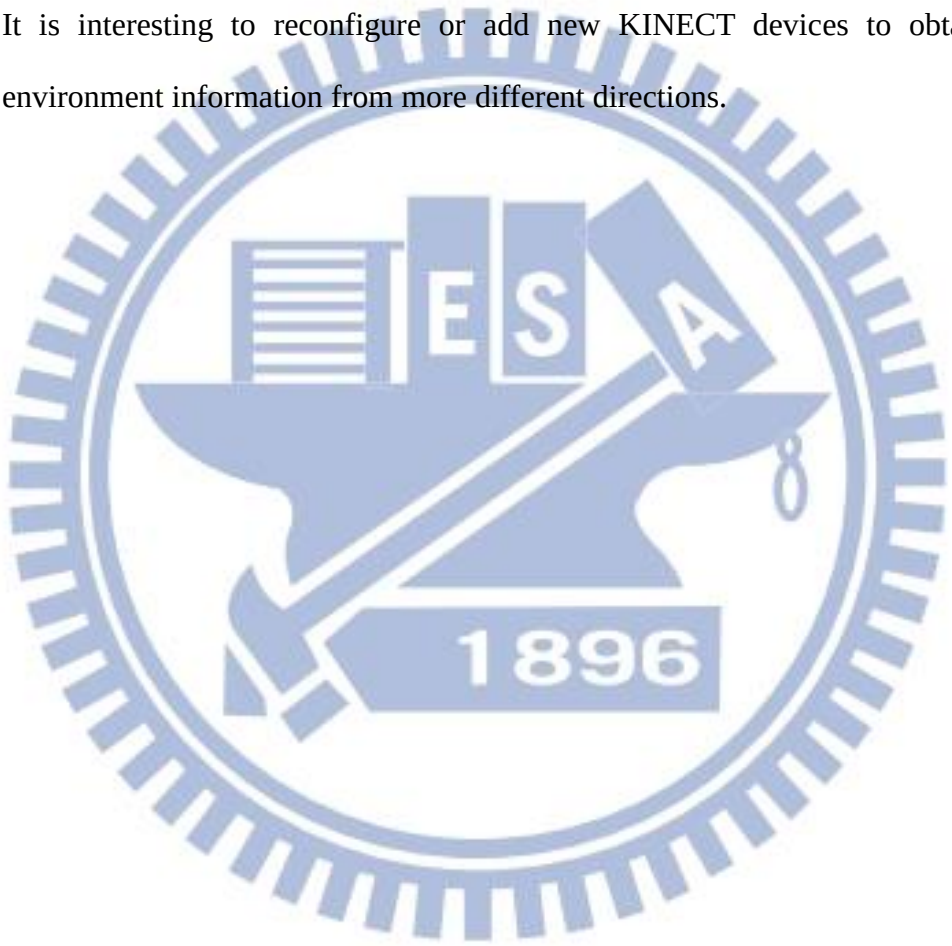
(3) It is challenging to test the proposed system in other experimental environments

because the infrared sensor equipped on the KINECT device will be invalidated by the sun effect.

(4) It is a challenge to develop an automatic learning system to speed up the learning time and to ease the use of the interface.

(5) It is an interesting to develop the capability of starting navigation from arbitrary start points.

(6) It is interesting to reconfigure or add new KINECT devices to obtain the environment information from more different directions.



References

[1] Kinect, "Kinect - Xbox," Available in http://www.xbox.com/en-US/en-uskinect/, Accessed in Aug. 2012.

[2] D. S. O. Correa, et al., "Mobile Robots Navigation in Indoor Environments Using Kinect Sensor," in *proceedings of Second Brazilian Conference on Critical Embedded Systems*, Campinas SP, Brazil, May 2012.

[3] D. Sales, et al., "3D Vision-Based Autonomous Navigation System Using ANN and Kinect Sensor," in *Engineering Applications of Neural Networks (EANN)*, CCIS 311, pp. 305–314, 2012.

[4] J. Cunha, et al., "Using a Depth Camera for Indoor Robot Localization and Navigation ," in *Robotics Science and Systems (RSS) conference*, 2011.

[5] J. Biswas and M. Veloso, "Depth Camera Based Indoor Mobile Robot Localization and Navigation," in *Proceedings of ICRA*, St. Paul, MN, USA, pp. 1697-1702, May. 2012.

[6] A. Robledo, et al., "Outdoor Ride: Data Fusion of a 3D Kinect Camera installed in a Bicycle," in *Proceedings of Australasian Conference on Robotics and Automation*, Melbourne, Australia, Dec. 2011.

[7] C. Rasmussen, "Kinects for Low- and No-Sunlight Outdoor Trail-Following," in *Robotics Science and Systems (RSS) conference*, Sydney, Australia, July 2012.

[8] S. Willis and S. Helal, "RFID information grid for blind navigation and wayfinding," *Proceedings of the 9th IEEE International Symposium on Wearable Computers*, pp. 34–37, Washington DC, USA, Oct. 2005.

[9] L. Ran, S. Helal, and S. Moore, "Drishti: an integrated indoor/outdoor blind navigation system and service," *Proceedings of 2nd IEEE Annual Conference on*

Pervasive Computing and Communications, Orlando, Florida, USA, pp. 23–31, 2004.

[10] 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, Nov. 2009.

[11] E. Abbot and D. Powell, "Land-vehicle navigation using GPS," *Proceedings of the IEEE*, vol. 87, no. 1, pp. 145–162, Jan. 1999.

[12] M. C. Chen, "Vision-based security patrolling in indoor environments using autonomous vehicles," *Proceedings of 2005 Conference on Computer Vision, Graphics and Image Processing*, pp. 811–818, Taipei, Taiwan.

[13] K. L. Chiang and W. H. Tsai. "Vision-based autonomous vehicle guidance in indoor environments using odometer and house corner location information," *Proceedings of 2006 IEEE International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP-2006)*, pp. 415–418, Pasadena, California, USA, Dec. 18-20, 2006.

[14] S. Atiya and Gregory D. Hager, "Real-time vision-based robot localization," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 6, pp. 785–800, Dec. 1993.

[15] D. Lopez, et al., "Hybrid laser and vision based object search and localization," *Proceedings of IEEE International Conference on Robotics and Automation*, pp.2636–2643, Pasadena, CA, USA, May 19-23, 2008.

[16] 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, Dec. 2008.

[17] M. Agrawal and K. Konolige, "Real-time localization in outdoor environments using stereo vision and inexpensive GPS," *Proceedings of 18th International Conference on Pattern Recognition*, Hong Kong, People's Republic of China, pp. 1063–1068, vol. 3, Aug, 2006.

[18] Kinect, "Developer SDK, Toolkit & Documentation, Kinect for Windows," Available in www.microsoft.com/en-us/kinectforwindows/develop/, Accessed in Aug. 2012.

[19] Kinect, "Microsoft Developer Network - Kinect for Windows SDK," Available in http://msdn.microsoft.com/en-us/library/hh855347, Accessed in Aug. 2012.

[20] H. Bay, et al., "Speeded-Up Robust Features (SURF)," in *Computer Vision and Image Understanding*, Vol. 110 (2008) 346–359.

[21] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, Vol. 60, pp. 91-110, 2004.

[22] J. K. Huang, "Autonomous vehicle navigation by two-mirror omni-directional imaging and ultrasonic sensing techniques," *M. S. Thesis*, Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, June 2010.

[23] Besl, Paul J. and N.D. McKay "A Method for Registration of 3-D Shapes," in *IEEE Trans. on Pattern Analysis and Machine Intelligence (Los Alamitos, CA, USA: IEEE Computer Society)*, 1992, 239–256. doi:10.1109/34.121791.

[24] H. C. Huang, "3D Monitoring of Car Surrounds Using Multiple KINECT Devices," *M. S. Thesis*, Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, June 2013.