國立交通大學

多媒體工程研究所

碩士論文

以天花板上多環場攝影機輔助自動車作室內安全監控之研究

A Study on Indoor Security Surveillance by Vision-based Autonomous Vehicle With Omni-cameras on House Ceiling

研 究 生：王建元

指導教授：蔡文祥　教授

中華民國 九十八 年六月

以天花板上多環場攝影機輔助自動車

作室內安全監控之研究

A Study on Indoor Security Surveillance by Vision-based Autonomous Vehicle With Omni-cameras on House Ceiling

研究生：王建元　　Student：Jian-Yuan Wang

指導教授：蔡文祥　　Advisor：Prof. 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 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年六月

以天花板上多環場攝影機輔助自動車
作室內安全監控之研究

研究生：王建元　　　　指導教授：蔡文祥　教授

國立交通大學多媒體工程研究所

摘要

本論文提出了一個以天花板上多環場攝影機輔助自動車做室內安全監控之方法。我們使用一部以無線網路傳遞控制訊號之自動車，並在其上裝置一部攝影機，用以監視室內環境及拍攝入侵者影像。另外使用兩部裝置在天花板上的魚眼攝影機協助此自動車做導航。在此研究中，我們提出了一種即時對環境中空地和障礙物做定位的技術，並以計算出的位置建立完整的環境地圖及規劃自動車的巡邏路線，使自動車可以在複雜的環境中導航並避開障礙物和牆壁。此外，我們亦提出了一種能夠適應高度變化的空間對映法，利用推導出的公式計算設置在不同高度的魚眼攝影機的對映表，以此對映表配合內插法，為環境中的物體進行定位。因為自動車行走時會產生機械誤差，我們也提出了四個策略，以修正自動車的位置及方向。此外，我們亦提出了一種追蹤入侵者的技術，使用追蹤視窗準確地預測及計算人物在扭曲影像中的位置，並在追蹤過程中同時記錄人物的特徵。為了擴大可監視的範圍，我們使用了多台裝置於天花板上的攝影機，為此我們也提出了一種在多台攝影機下“交棒”(handoff)的技術，使自動車或入侵者從一台攝影機的視野範圍移動到另外一台時，能夠不間斷的被追蹤。實驗結果證明我們所提出的方法是可行而且有效的。

A Study on Indoor Security Surveillance by Vision-based Autonomous Vehicles with Omni-cameras on House Ceilings

Student: Jian-Yuan Wang Advisor: Prof. Wen-Hsiang Tsai, Ph. D.

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

ABSTRACT

Vision-based methods for security surveillance using an autonomous vehicle with fixed fish-eye cameras on ceilings in an indoor environment are proposed. An autonomous vehicle controllable by wireless communication and equipped with a camera is used as a test bed and navigates in a room space under the surveillance of multiple fisheye cameras affixed on the ceiling. To learn the information of the unknown room environment in advance, a method is proposed for locating the ground regions, identifying the positions of obstacles, and planning the patrolling paths. The data obtained by the method enable the vehicle to navigate in the complicated room space without collisions with obstacles and walls. Also, a height-adaptive space mapping method is proposed, in which the coordinates of corresponding points in 2-D images and 3-D global spaces are computed by interpolation to form a space mapping table for object localization. Appropriate equations are derived to adapt the table to fish-eye cameras affixed to different ceiling heights. Because the vehicle suffers from mechanic errors, a vehicle location and direction correction method is proposed for correcting the errors according to four strategies. Furthermore, a method for detecting and tracking intruding people is proposed. The approximate position of the person can be predicted first, and the exact position is then calculated via a tracking window in images. Some useful features of the intruding person are computed for person identification. To enlarge the area under surveillance using multiple cameras, the camera handoff problem is also solved by using information of the overlapping regions of the cameras' fields of view. Experiments for measuring the precisions of the proposed methods and tracking intruding persons were conducted with good results proving the feasibility of the proposed methods.

ACKNOWLEDGEMENTS

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

Thanks are due to Mr. Chih-Jen Wu, Mr. Che-Wei Lee, Mr. Guo-Feng Yang, Miss Mei-Fen Chen, Mr. Yi-Chen Lai, Miss Chiao-Chun Huang, Miss Shu-Hung Hung, Miss Chin-Ting Yang, and Mr. Chun-Pei Chang for their valuable discussions, suggestions, and encouragement. 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 family for their lasting love, care, and encouragement. He dedicates this dissertation to his beloved parents.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

1.1 Motivation of Study

As the technology is progressing nowadays, more and more robots emerge in many applications. An autonomous vehicle is an important and common form of robots. It can move and turn by the control of programs and can take images by cameras equipped on it to increase its abilities. It is convenient to use autonomous vehicles to substitute for human beings in many automation applications. For example, a vehicle may be utilized to patrol in an environment for a long time without a rest. In addition, the video which is taken by a camera can be recorded forever for later search for its contents of various interests. For example, if a certain object in a video-monitored house is stolen, the video can be watched to find out possibly the thief, providing a better evidence of the crime than just the memory of a person.

Using vehicles to patrol in indoor environments automatically is convenient and can save manpower. The images which are taken by the cameras on vehicles may be transmitted by wireless networks to a central surveillance center, so that a guard there can monitor the environment without going to the spots of events, and this is also safer for a guard to avoid conflicts with invaders. Additionally, it is useful for a vehicle to follow a suspicious person who breaks into an indoor environment under automatic video surveillance, and clearer images of the person’s behavior can be taken by the cameras on the vehicle. It is desired to investigate possible problems raised in achieving the previously-mentioned goals and to offer solutions to them. Possible problems include:

1. constructing the mapping tables of cameras automatically, so that the positions of vehicles, invading persons, concerned objects, etc. can be computed;
2. detection of concerned objects and humans from images acquired by cameras equipped on vehicles and/or affixed to walls or ceilings;
3. recording of invaders' trajectories and computation of their walking speeds for later inspection for the security purpose.

We try to solve in this study all these problems for indoor environments with complicated object arrangements in space. But if the environment under surveillance is very large, we cannot monitor the entire environment by using just one omni-camera. So it is desired to use simultaneously several cameras affixed on ceilings to cover large environment areas. In order to achieve this goal, possible problems to be solved include:

1. calculating the relative positions of the omni-cameras whose fields of view (FOV's) overlap;
2. calculating the rotation angles of the omni-cameras;
3. handling with the hand-off problem among multiple cameras.

By calculating the relative positions and rotation angles of the cameras, we can calculate the position of an object in images which are taken by the cameras. And when a person walks from one region covered by a camera to another region covered by another camera, the system should know which camera should be used to get the image of the person and where in the image should the person be localized. This is the camera hand-off problem which we desire to deal with in this study.

Because omni-cameras are highly distorted and are affixed to the ceiling, they cannot monitor the whole environment clearly. On the other hand, autonomous vehicles are mobile and are suitable to remedy this shortness of cameras. Hence, we can utilize the vehicles to drive to the place where should be monitored to take clearer images of concerned objects or humans there as stronger evidences for crime investigations.

Another problem is that the FOV's of omni-cameras are finite, and the cameras are expensive. If the indoor environment is very large, we will have to use many omni-cameras on the ceiling, as mentioned previously. But if we can utilize the cameras on vehicles to take images of the place which is out of the FOV's of the cameras, we will not have to use a lot of omni-cameras.

Hence if we want to navigate a vehicle to some spot in the environment, we should calculate the position of the vehicle first, and then we can plan the path for a vehicle from its position to the spot. In most environments, there are a lot of obstacles in them, such as furniture and walls. In order to avoid collisions between vehicles and obstacles, we may gather the information of the environment first. The information may include the positions of still obstacles and open spaces where the vehicles can walk through. Afterward, we may integrate the information to construct an environment map for the purpose of convenience. In short, an environment map is used in obstacle avoidance and path planning in this study. If we want to drive a vehicle to a certain spot out of the FOV's of the cameras, we should calculate the position and direction of the vehicle at any time and plan a path for the vehicle to drive to that spot. Possible problems in these applications include:

1. gathering environmental information and constructing an environment map;
2. calculating the position and direction of each vehicle;

3. path planning and avoidance of still and dynamic obstacles in the path for the vehicle to navigate to its destination.

As a summary, in this study it is desired to investigate solutions to various problems involved in the following topics of indoor autonomous vehicle navigation:

1. security patrolling in indoor environments by autonomous vehicles;
2. effective integration of the omni-cameras on the ceiling and the cameras on the vehicles.
3. following a suspicious person and taking clearer images of her/him by the cameras on vehicles;
4. using the cameras on vehicles to monitor spots which are out of the FOV's of omni-cameras and take clearer images.

1.2 Survey on Related Studies

In the study, we will use multiple omni-cameras on the ceiling to locate the position of a vehicle, so the omni-cameras should be calibrated before being used. Traditionally, the intrinsic and extrinsic parameters of the camera should be calculated in order to obtain a projection matrix for transforming points between 2-D image and 3-D global spaces [1, 2, 3]. Besides, a point-correspondence technique integrated with an image interpolation method have been proposed in recent years for object location estimation [4], but it will cause another problem, that is, the calibration data will change according to the environment where the cameras are used. In this study, we will propose a technique to solve this problem for the case of changing the height of the ceiling on which the cameras are affixed.

Autonomous vehicles in general suffer from mechanical errors, and many methods have been proposed to eliminate this kind of error. The geometric shapes of object boundaries [5, 6] or those labeled by users are utilized frequently [7, 8]. Furthermore, natural landmarks, such as house corners [9, 10] and the SIFT features of images [11], are also used to correct the position of a vehicle. In recent years, techniques of integrating laser range finders with conventional imaging devices have been proposed [12, 13]. Besides, when it is desired to find a specific object in the image, the method of color histogramming is often used [14].

The applications of autonomous vehicles emerge in many aspects, such as house cleaning robots, watchdog systems, automatic guides, etc. In Takeshita [15], a camera was equipped on the ceiling, and a user can control the vehicle to suck garbage on the ground by watching the images taken by the camera. In Yang [16], the vehicles were designed to patrol in an environment, too. He used the vectors of vehicles and obstacles to avoid collisions between them.

1.3 Overview of Proposed System

There are four main goals in this system. First, a vehicle should patrol automatically in an indoor environment whose information has been learned. Second, the vehicle should avoid static and dynamic obstacles, and third it should correct its position automatically. At last, the vehicle should follow an intruding person and take the images of the person.

In order to achieve these goals, the following steps should be done:

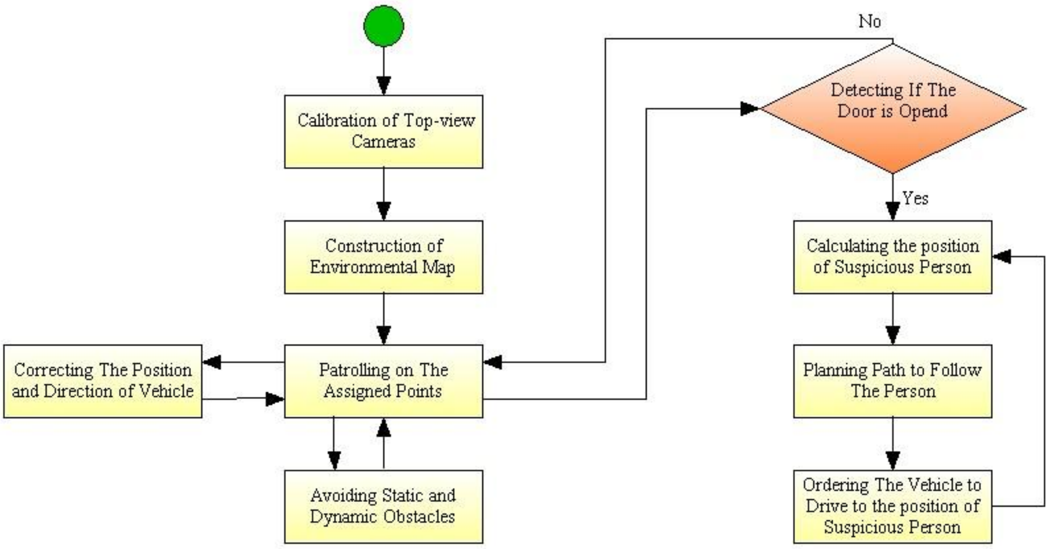
1. construct mapping tables for top-view cameras;

2. acquire environment information by top-view cameras;
3. detect obstacles on the path;
4. correct mechanic errors by top-view omni-cameras;
5. calculate the position of any intruder by top-view omni-cameras continually;
6. initiate the vehicle to follow the intruder and take images of him/her;
7. deal with the hand-off problem of the cameras.

Because we need to convert coordinates between the image coordinate system and the global coordinate system, we have to construct the mapping tables for the cameras we use first. Afterward, the coordinates between the multiple cameras can be transformed correctly. Because the vehicles patrol in the indoor environment, the environment information should be learned in advance. The information learned in this study includes the positions of obstacles and the open space in the environment where the vehicles can walk through. And the information will be used to build an environmental map.

When vehicles patrol in the environment whose information has been learned, the patrolling path can be checked to see if there are obstacles on the path. If so, the vehicles should avoid them automatically. Besides, the vehicles generally suffer from mechanical errors, so it needs to correct their positions and directions continuously, to avoid intolerable deviations from their correct path way.

When a person breaks into the environment, the position of a person will be calculated continually, and then the computer will give orders to guide a vehicle to follow the person. In order to expand the range of surveillance, several omni-cameras are used in the study, so the hand-off problem should be handled. The problem means briefly the need of identifying a person in an image acquired by a camera and passing the information to an image taken by another camera. Figure 1.1 shows a flowchart of the whole proposed system for security patrolling by autonomous vehicles.



Calibration of Top-view Cameras
Construction of Environmental Map
Correcting The Position and Direction of Vehicle
Patrolling on The Assigned Points
Avoiding Static and Dynamic Obstacles
No
Detecting If The Door is Opend
Yes
Calculating the position of Suspicious Person
Planning Path to Follow The Person
Ordering The Vehicle to Drive to the position of Suspicious Person

Figure 1.1 The flowchart of proposed system.

1.4 Contributions

Several contributions are made in this study, as described in the following:

1. A height adaptation method is proposed to construct the mapping tables for omni-cameras in order to make the cameras usable at different ceiling heights.
2. An integrated space mapping method is proposed to localize objects in real space using multiple fisheye cameras.
3. A method is proposed to solve the problem of camera hand-off in highly distorted images taken by fisheye cameras.
4. A method is proposed to gather distorted environment images taken by omni-cameras, and convert them into a flat map.
5. A method is proposed to correct dynamically errors of the position and direction of a vehicle caused by mechanical errors.
6. A method is proposed to avoid static and dynamic obstacles automatically in real

time on vehicle navigation paths.

7. A technique is proposed to calculate the position of a person according to rotational invariance property of omni-camera.

8. A method is proposed to predict the position of a person in a highly distorted image.

1.5 Thesis Organization

The remainder of this thesis is organized as follows. In Chapter 2, the hardware and processes of this system will be introduced. And in Chapter 3, the proposed method for constructing the mapping tables for fisheye cameras will be described.

In Chapter 4, the construction steps of an environmental map and the proposed method for obstacle avoidance are described, and four strategies of correcting the positions and directions of vehicles are also described. In Chapter 5, the method of finding specific partial regions in an image to compute the position of a person and the technique for prediction of the person’s next movement are described. The hand-off problem is solved in this chapter, too.

The experimental results of the study are shown in Chapter 6, and some discussions are also included. At last, conclusions and some suggestions for future works are given in Chapter 7.

Chapter 2

System Configuration

2.1 Introduction

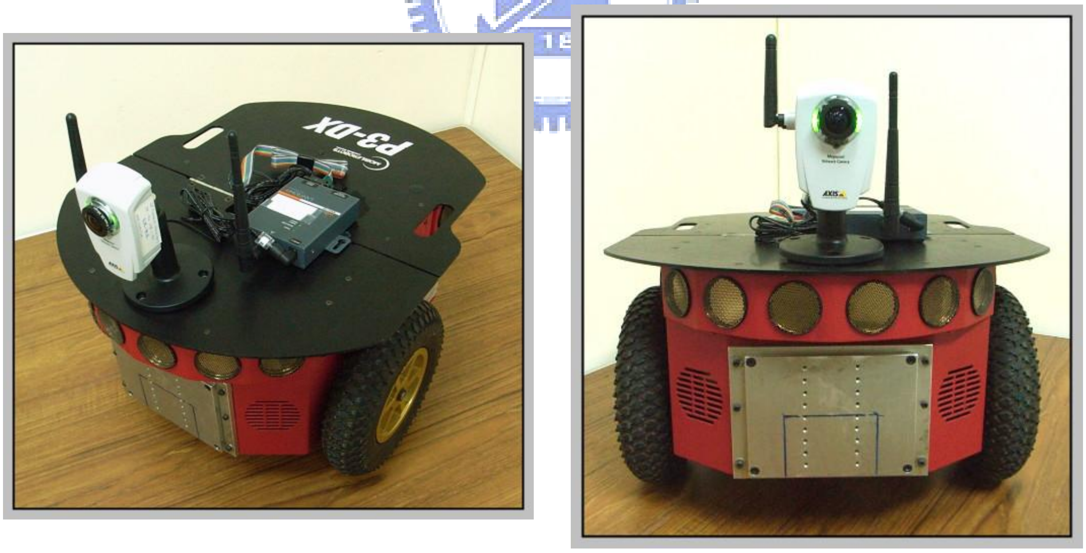
The hardware and software which are used in this study will be introduced in this chapter. The hardware includes the autonomous vehicle we use and the fisheye cameras and wireless network equipments. The software includes the programs for the processes of gathering the information of an environment, constructing an environment map, avoiding obstacles when patrolling in an environment and calculating the position of a person automatically.

2.2 Hardware

The autonomous vehicle we use in this study is a Pioneer 3-DX vehicle made by MobileRobots Inc., and an Axis 207MW camera made by AXIS was equipped on the vehicle as shown in Figure 2.1. The Axis 207MW camera is shown in Figure 2.2.

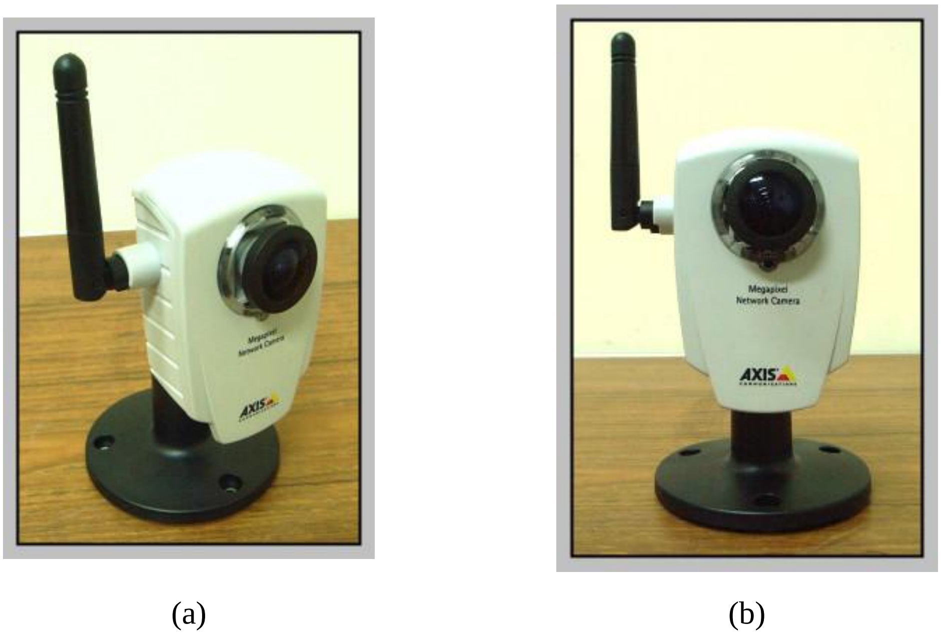
The Pioneer 3-DX vehicle has a 44cm×38cm×22cm aluminum body with two 19cm wheels and a caster. It can reach a speed of 1.6 meters per second on flat floors, and climb grades of 25^{o} and sills of 2.5cm. At slower speeds it can carry payloads up to 23 kg. The payloads include additional batteries and all accessories. By three 12V rechargeable lead-acid batteries, the vehicle can run 18-24 hours if the batteries are fully charged initially. A control system embedded in the vehicle makes the user’s commands able to control the vehicle to move forward or backward, or to turn around. The system can also return some status parameters of the vehicle to the user.

The Axis 207MW camera has the dimension of 85×55×40mm (3.3”×2.2”×1.6”), not including the antenna, and the weight of 190g (0.42 lb), not including the power supply, as shown in Figure 2.4. The maximum resolution of images is up to 1280×1024 pixels. In our experiment, the resolution of 320×240 pixels is used for the camera fixed on the vehicle and that of 640×480 pixels is used for the one affixed on the ceiling. Both of their frame rates are up to 15 fps. By wireless networks (IEEE 802.11b and 802.11g), captured images can be transmitted to users at speeds up to 54 Mbit/s. Each camera used in this study is equipped with a fish-eye lens that expands the field of view of a traditional lens in general.



(a) (b)

Figure 2.1 The vehicle used in this study is equipped with a camera. (a) A perspective view of the vehicle. (b) A front view of the vehicle.



(a) (b)

Figure 2.2 The camera system used in this study. (a) A perspective view of the camera. (b) A front view of the camera.

The Axis 207MW cameras are fisheye cameras. They are also affixed on the ceiling and utilized as omni-cameras, as shown in Figure 2.3. A notebook is used as a central computer to control the processes and calculate needed parameters from the information gathered by the cameras.

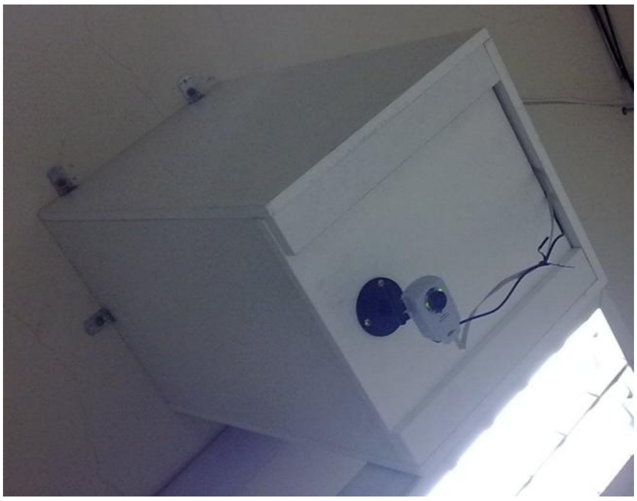
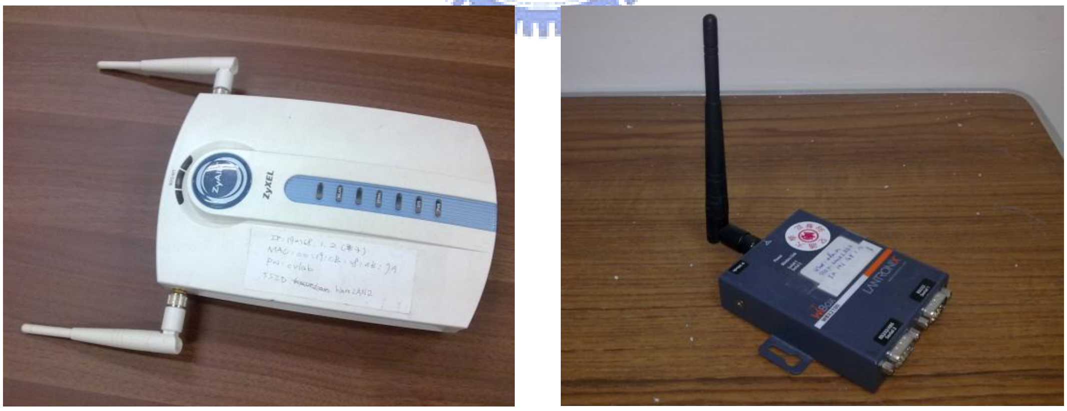


Figure 2.3 An Axis 207MW camera is affixed on the ceiling.

Figure 2.4 A notebook is used as the central computer.

Communication between the hardware components mentioned above is via a wireless network, a WiBox made by Lantronix equipped on the vehicle, in order to deliver and receive the signals of the odometer as shown in figure 2.5.



(a)

(b)

Figure 2.5 The wireless network equipments. (a) A wireless access point. (b) A WiBox made by Lantronix.

2.3 System Process

In the proposed process of constructing the mapping tables for omni-cameras, we calculate the relative positions and rotation angles between the cameras. Afterward, mapping tables are constructed automatically for every camera, and a point-correspondence technique integrated with an image interpolation method is used to calculate the position of any object appearing in the image.

In the proposed process of environment learning, the information is gathered by a region growing technique first. The information includes the position of obstacles and open spaces where a vehicle can drive through. Afterward, the positions will be converted into the global coordinate system, and an environment map will be constructed by composing the coordinates of all obstacles appearing in the vehicle navigation environment.

In the proposed process of security patrolling in an indoor environment, each vehicle is designed to avoid obstacles on the navigation path. If there are some obstacles on a path of the vehicle, the vehicle system will plan several turning points to form a new path for the vehicle to navigate safely. After the vehicle patrol for a while, it will diverge from its path because the vehicle suffers from mechanical errors. In the proposed process of vehicle path correction, we calculate the position of a vehicle in the image coordinate system by the images taken by the omni-cameras affixed on the ceiling, and then convert the coordinates into global coordinates and modify accordingly the value of the odometer in the vehicle.

In the proposed process of person following, the position of an intruding person's feet is calculated also by the images taken by omni-cameras on the ceiling, and then the coordinates are converted into global coordinates, too. Afterward, the vehicle system will calculate the relative position and the rotation angle between the vehicle and the person, and adjust accordingly the orientation and speed of the vehicle to achieve the goal of following the person.

The major processes of the system are summarized and listed below:

1. Construct mapping tables for every top-view cameras.
2. Acquire environmental information by top-view cameras and construct the environment map.
3. Correct mechanic errors continuously in each cycle.
4. Plan a path to avoid obstacles in the environment.
5. Detect, predict, and compute the position of any intruding person by top-view omni-cameras continuously.
6. Handle the camera hand-off problem to keep tracking any intruding person using a single camera at a time.

Chapter 3

Adaptive Space Mapping Method for Object Location Estimation Subject to Camera Height Changes

3.1 Ideas of Proposed Adaptive Space Mapping Method

In this study, we use multiple fish-eye cameras affixed on the ceiling to keep an indoor environment under surveillance. The cameras are utilized to locate and monitor the autonomous vehicle, and trace the track of any suspicious person when he/she comes into the environment. When using these omni-cameras, we want to know the conversion between the ICS (image coordinate system) and the GCS (global coordinate system). So we propose a space mapping method and construct a mapping table for converting the coordinates of the two coordinate systems.

Because the indoor environment under surveillance is unknown at first, we propose further in this study another space mapping method by which the cameras can be affixed to different ceiling heights for use, which we call *height-adaptive space mapping method*. Besides, multiple fish-eye cameras are used in the mean time to monitor an environment in this study, so calculating the relative positions and angles between every two cameras which have overlapping fields of view (FOV's) is needed and is done in this study. Finally, a point-correspondence technique integrated with an image interpolation method is used to convert the coordinates between the ICS and the GCS.

3.2 Construction of Mapping Table

In this section, we propose a method of constructing a basic mapping table for use at a certain ceiling height. The mapping table we use contains 15×15 pairs of points, each pair including an image point and a corresponding space point. And the data of each pair of points in the table includes the coordinates (x_1, y_1) of the image point in the ICS and the coordinates (x_2, y_2) of the corresponding space point in the GCS. We use a calibration board which contains 15 vertical lines and 15 horizontal lines to help us constructing the mapping table.

First, we take an image of the calibration board, find the curves in the image by curve fitting, and calculate the intersection points of the lines in the image. Then, we measure manually the width of the real-space interval between every two intersection points, and compute the coordinates of the intersection points in the GCS in terms of the value of this interval width. Afterward, we affix the cameras to the ceiling, and modify the global coordinates in the mapping table by the height of the ceiling. The details of these tasks will be described in Sections 3.2.1 and 3.2.2. The major steps from constructing a space mapping table to calculating the position of a certain object are described below.

Step 1. Set up a camera (not necessarily attached to the ceiling) at a known height from the floor and take an image of the calibration board under the camera.

Step 2. Find the curves in the image of the calibration board by curve fitting, and calculate the coordinates of the intersection points of the curves in both the ICS and the GCS, forming a *basic space mapping table*.

Step 3. Attach the camera to the ceiling.

Step 4. Calculate the coordinates of the intersection points of the calibration board in the GCS utilizing the height of the ceiling assumed to be known and the content of the basic space mapping table, forming an *adaptive space mapping table*.

Step 5. Calculate the relative positions and rotation angles between the cameras.

Step 6. Calculate the position of any object under the camera by a point-correspondence technique integrated with an image interpolation method using the adaptive space mapping table.

In both the basic and adaptive space mapping tables, we only record the coordinates of the corresponding intersection points between the ICS and the GCS. When we want to calculate the position of a certain object which is not right on the above-mentioned intersection points, a point-correspondence technique integrated with an image interpolation method will be applied (Step 6 above). The detail will be described in Section 3.4. We only have to calculate the coordinates in the GCS and construct the adaptive mapping table *once* after the cameras are attached to the ceiling (Step 4 above). The adaptive mapping table can be stored in a computer and can be used any time.

3.2.1 Proposed Method for Constructing A Basic Mapping Table

The basic mapping table of a camera should be constructed first before the camera can be used for environment monitoring. Assume that the camera is attached at a fixed height *h* (not necessarily to the ceiling). To construct the basic mapping table, at first we put a calibration board on the ground right under the camera. The calibration board contains at least 15 horizontal lines and 15 vertical lines, and the intervals between every two lines are all the same, as shown in Figure 3.1. Then, we use the camera to take an image of the calibration board, extract the quadratic curves in the image by minimum mean-square-error (MMSE) curve fitting, and find the intersections of these curves, as shown in Figure 3.2.

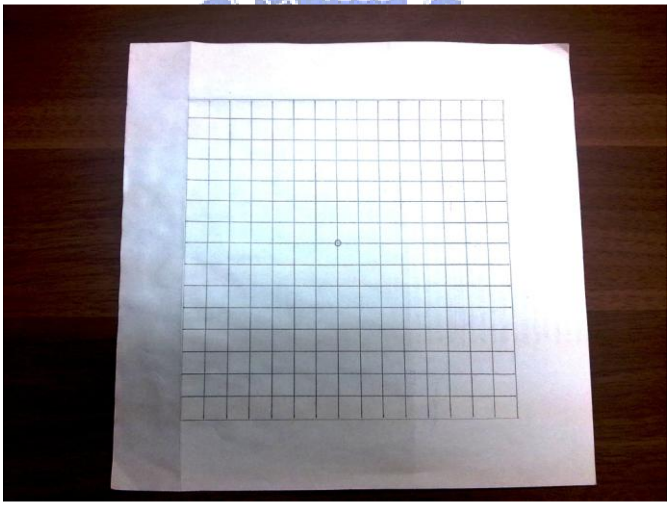


Figure 3.1 The calibration board used for basic table construction with 15 horizontal lines and 15 vertical lines.

These intersection points are described by image coordinates, which are recorded in the basic mapping table. And the height *h* of the camera is recorded, too. We assume that the upmost and leftmost point found in the image is taken to correspond to the origin in the GCS, as shown in Figure 3.3.

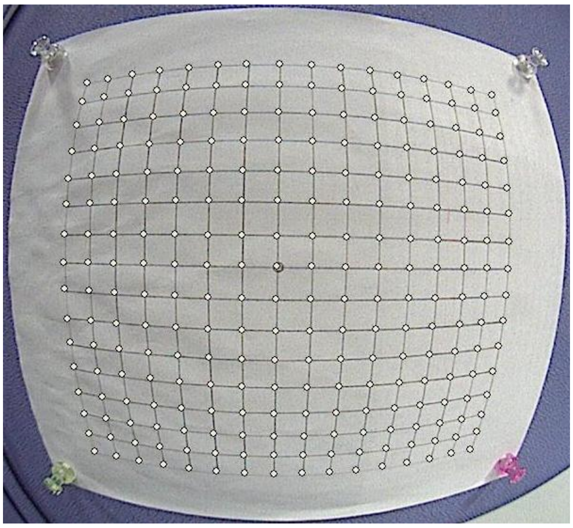


Figure 3.2 Finding the intersection points in the image of the calibration board.

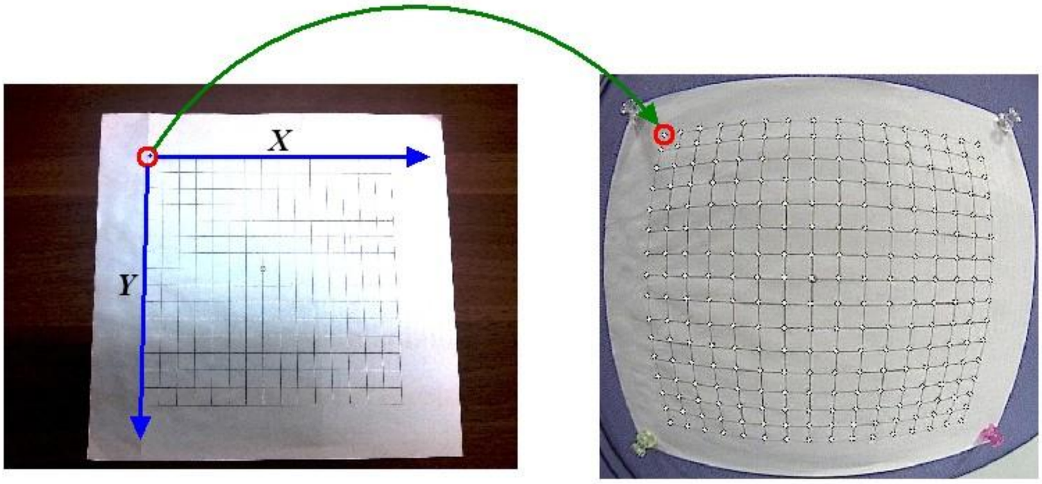


Figure 3.3 The mapping of the origins of the ICS and GCS.

Afterward, we measure manually the width of the real-space interval W_{cali} between every two intersection points on the calibration board, and the global coordinates of the intersections can be calculated by the following way.

First, assume that the upmost and leftmost point in the image is just the projection of the origin of the GCS. So, the global coordinates of this point are (0, 0). The *x*-axis is in the horizontal direction, and the *y*-axis is in the vertical direction.

Hence the coordinates of the intersection points in the GCS can be calculated simply as a multiple of the value W_{cali} as shown in Figure 3.4. Or more specifically, the global coordinates of the (i, j)-th intersection point in Figure 3.4 may be computed as $(i\times W_{cali}, j\times W_{cali})$, where the origin is regarded to be the (0, 0)-th intersection point.

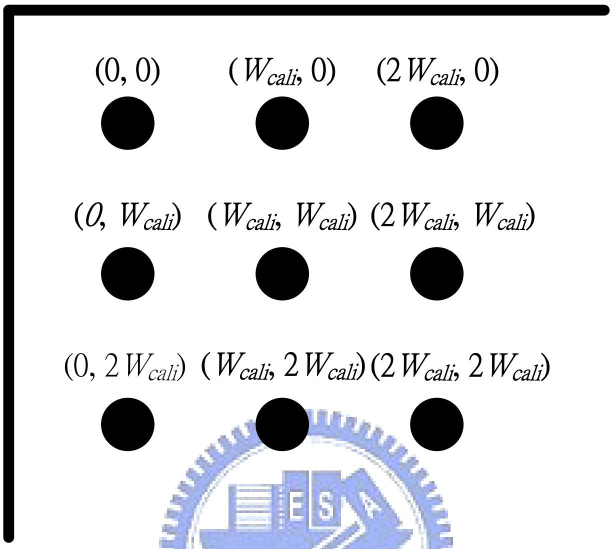


Figure 3.4 Calculating the coordinates of intersection points in the GCS by W_{cali}.

If the cameras are used at a height which is the same as that used during the stage of constructing the basic mapping table, we can then use this table to calculate the position of any object under the camera by a point-correspondence technique integrated with an image interpolation method, which will be described in next section.

But if the camera is used at a different height, the global coordinates in the basic mapping table should be modified. Otherwise, the object position we calculate later will be wrong. The method will be described in Section 3.2.3.

3.2.2 Using a Point-correspondence Technique Integrated With An Image Interpolation Method to Locate Objects

After the basic mapping table is constructed, we can know the corresponding coordinates of the above-mentioned intersection points in the ICS and the GCS. But when we want to calculate the position of a certain object which is not right on any intersection point, a point-correspondence technique integrated with an image interpolation method is applied in this study. As shown in Figure 3.5, we want to calculate the global coordinates of the point *I* which is not right on the intersection point. The detail is described as an algorithm in the following.

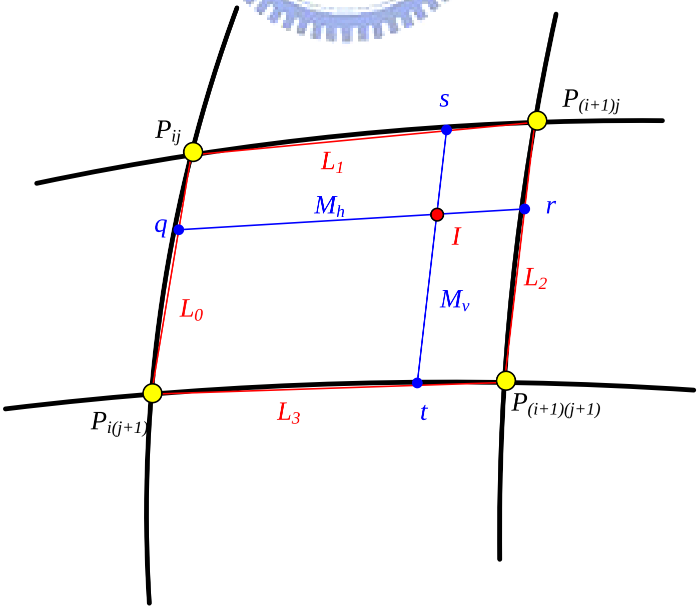


Figure 3.5 Calculating the coordinates of a non-intersection point by interpolation.

Algorithm 3.1: *Calculating the coordinates of an object in the GCS.*

Input: A mapping table and the image coordinates of a non-intersection point *I*.

Output: The global coordinates of the space point *G* corresponding to *I*.

Steps:

Step 1. Derive the equations of the lines L_0, L_1, L_2 and L_3 in the ICS by the image coordinate data of the points P_{ij}, $P_{(i+1)j}$, $P_{(i+1)(j+1)}$ and $P_{i(j+1)}$, as shown in Figure 3.5.

Step 2. Derive the equations of lines M_h and M_v in the ICS.

Step 3. Calculate the image coordinates of the intersection points *s*, *r*, *t*, and *q*.

Step 4. Calculate the global coordinates *G* of point *I* in the GCS.

The above algorithm is just an outline, whose details are now explained. In Step 1, the image coordinates of P_{ij}, $P_{(i+1)j}$, $P_{(i+1)(j+1)}$ and $P_{i(j+1)}$ are known, so the equations of lines L_0, L_1, L_2 and L_3 in the ICS can be derived. In Step 2, the slope of M_h is the average of those of lines L_1 and L_3, and the slope of M_v is the average of those of lines L_0 and L_2. The equation of M_h and M_v in the ICS can be derived by these slopes and *I*. Then, the intersection point *s* of L_1 and M_v can be computed accordingly, so are *r*, *t*, and *q* in similar ways. Finally, the global coordinates (G_x, G_y) of *I* can be calculated by the following formulas according to the principle of side proportionality under the assumption that the space area enclosed by the four corners P_{ij}, $P_{(i+1)j}$, $P_{(i+1)(j+1)}$ and $P_{i(j+1)}$ is not too large so that the linearity inside the area holds:

$$G_x = X + W_{cali} \times \frac{d(q,I)}{d(q,r)}; \tag{3.1}$$

$$G_y = Y + W_{cali} \times \frac{d(s,I)}{d(s,t)}, \tag{3.2}$$

where (*X*, *Y*) are the global coordinates of P_{ij} computed in a way described previously, and $d(q, I)$ means that the distance between points q and I. Note that $(X, Y) = (i\times W_{cali}, j\times W_{cali})$ as mentioned previously in Section 3.2.1 if P_{ij} is the (i, j)-th intersection point in the calibration board.

3.2.3 Proposed Method for Constructing An Adaptive Mapping Table

If the camera is used at a height different from that used in constructing the basic mapping table, then when calculating the global coordinates of the intersections found on the calibration board, the value of the interval should be changed to another value W_{real} as shown in Figure 3.6; otherwise, the computation results will be incorrect. Thereafter, the method of calculating the global coordinates of space points is the same as discussed previously.

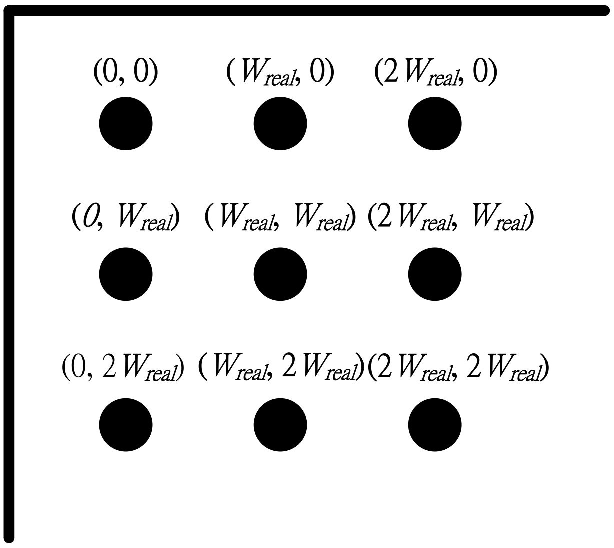


Figure 3.6 Calculating the coordinates of intersection points in the GCS by W_{real}.

To compute W_{real}, we make the following observation first. When the camera is attached to a lower height, the FOV's will decrease, and the number of points found in Figure 3.2 is the same, so the value of W_{real} will also decrease. On the other hand, when cameras are attached to a higher height, the FOV's will increase, and the value of W_{real} will increase, too. Accordingly, we derive a formula to calculate W_{real} in the following. Note that if we use multiple cameras in this study, the global coordinates should be revised additionally, because there may be some rotation between the cameras. The detail of this problem will be described in Section 3.3.

As depicted in Figure 3.7 there is a camera lens and an object O_1 with height h_1 on the left. Hence, the formation of an image of O_1 will be on the right side. The distance between point B and C is d_1, the distance between point C and E is f, and the distance between E and F is a. The height of the projection of the object on the image is i. Because the two triangles ABC and CFG are similar in shape, we can obtain the following equation from the principle of side proportionality:

$$\frac{d_1}{i} = \frac{h_1}{f + a}. \qquad (3.3)$$

Similarly, from the similar triangles DCE and EFG, we can obtain the following equation:

$$\frac{d_1}{i} = \frac{f}{a}. \qquad (3.4)$$

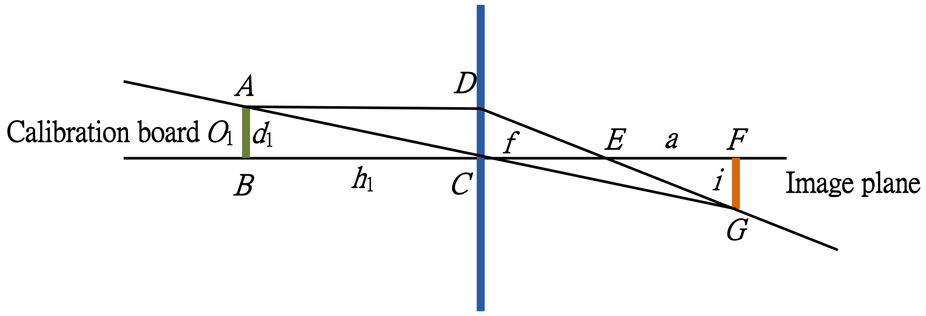


Figure 3.7 The imaging process of an object O_1.

In Figure 3.8 another object O_2 with height h_2 is on the left, and the formation of image is on the right side with height i, too. But the distance between O_2 and lens is farther than O_1. The lens is the same as in Figure 3.7, and we can also obtain two equations:

$$\frac{d_2}{i} = \frac{h_2}{f+b}; \tag{3.5}$$

$$\frac{d_2}{i} = \frac{f}{b}. \tag{3.6}$$

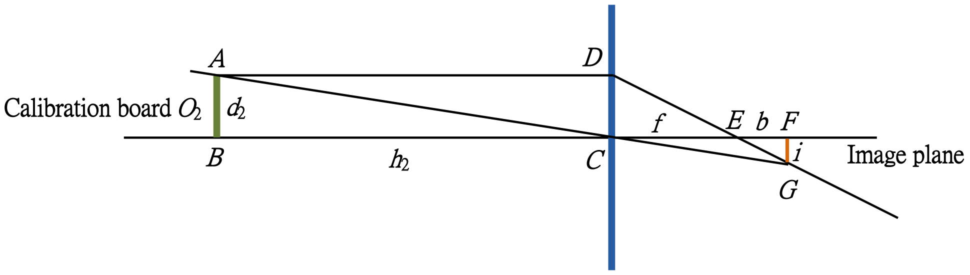


Figure 3.8 The imaging process of object O_2.

By Equations 3.3 and 3.4, we know

$$\frac{h_1}{f+a} = \frac{f}{a}. \tag{3.7}$$

By Equations 3.5 and 3.6, we know

$$\frac{h_2}{f+b} = \frac{f}{b}. \tag{3.8}$$

Then, we divide $\frac{d_1}{i}$ by $\frac{d_2}{i}$ according to Equations 3.4 and 3.6 as follows:

$$\frac{d_1}{i} \div \frac{d_2}{i} = \frac{f}{a} \div \frac{f}{b} \tag{3.9}$$

$$\Rightarrow \frac{d_1}{d_2} = \frac{b}{a} \tag{3.10}$$

$$\Rightarrow b = a \times \frac{d_1}{d_2}. \tag{3.11}$$

We also derive $\frac{h_1}{h_2}$ by Equations 3.7 and 3.8 as follows:

$$\frac{h_1}{h_2} = \frac{\dfrac{f^2 + af}{a}}{\dfrac{f^2 + bf}{b}} \tag{3.12}$$

$$= \frac{f^2 + af}{a} \times \frac{b}{f^2 + bf} . \tag{3.13}$$

Substituting Equation 3.11 into the above for the value of *b*, we get

$$\frac{h_1}{h_2} = \frac{f^2 + af}{a} \times \frac{a \times \dfrac{d_1}{d_2}}{f^2 + af \times \dfrac{d_1}{d_2}} \tag{3.14}$$

$$= \frac{(f + a) \times \dfrac{d_1}{d_2}}{f + a \times \dfrac{d_1}{d_2}} \tag{3.15}$$

$$= \frac{(f + a) \times \dfrac{d_1}{d_2}}{f + b} . \tag{3.16}$$

Because the focus length of the camera does not change, and if the distance between the lens and the object image is fixed, then *f* + *a* will be equal to *f* + *b*. Hence, (3.16) leads to the following formulas:

$$\frac{h_1}{h_2} = \frac{d_1}{d_2} \tag{3.17}$$

$$\Rightarrow d_2 = \frac{h_2 \times d_1}{h_1} . \tag{3.18}$$

We may rewrite the formulas above more clearly by other symbols as follows:

$$W_{real} = \frac{H_{real} \times W_{cali}}{H_{cali}} . \tag{3.19}$$

where W_{cali} is the interval between every two intersection points on the calibration board as mentioned before; H_{cali} is the height between the camera and the calibration board, alse mentioned previously; H_{real} is the real height at which the camera is affixed now; and W_{real} is the desired new interval for use now to replace the value W_{cali} in Equations (3.1) and (3.2).

3.3 Using Multi-cameras to Expand the Range of Surveillance

Because the range of surveillance of a single camera is finite, we use multiple cameras to expand the range of surveillance in the study. First, the adaptive mapping tables should be constructed for every camera by the method mentioned above. But if there are displacements and rotations between the cameras, then the global coordinates of the mapping table should be modified further. The major steps of constructing a modified mapping table are listed below, and the detail will be described in the rest of this section.

1. Calculate the relative position of two cameras.
2. Calculate the relative rotation angle of two cameras.
3. Calculate the global coordinates of intersection points in the calibration board in the expanded range.

3.3.1 Calculating Relative Position of Cameras

We design a special target board with a shape of a very small rhombus region on the ground as shown in Figure 3.9, and put it at the upmost and leftmost intersection area of the two cameras as shown in Figure 3.10. Because we want to calculate the global coordinates of all the intersections in the image via the coordinates of the center of this upmost and leftmost intersection area, we need to calculate the global coordinates of this center point first.

Then, an image of the target board is taken with the first camera. The target board area in the image are found out by the information of color, and the coordinates (*x*, *y*) of the center point of the rhombus shape in the GCS are calculated by the mapping table of the first camera. Accordingly, we can know that the global coordinates of the upmost and leftmost intersection of the second camera are (*x*, *y*). More details are described as an algorithm as follows.

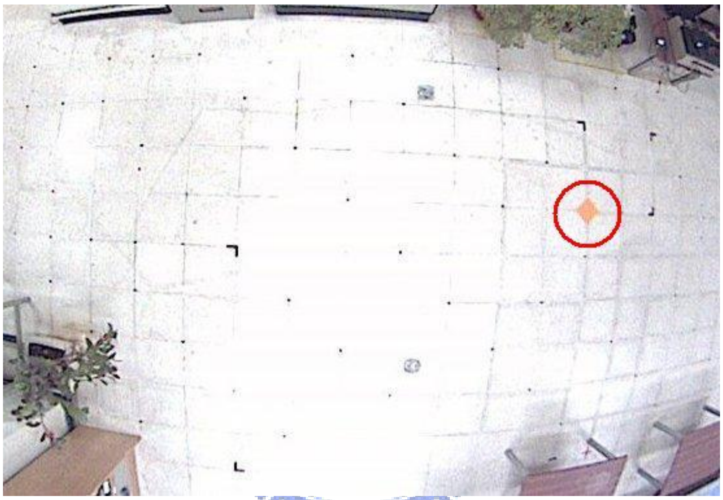


Figure 3.9 An Image of a specific target board on the ground seen from the first camera.

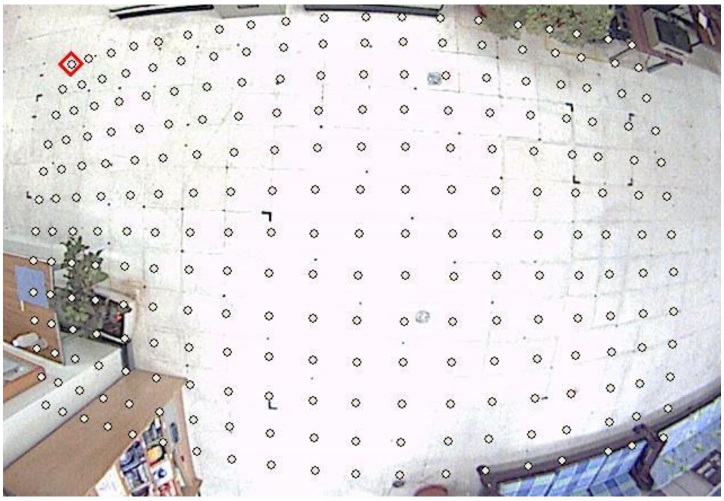


Figure 3.10 The position of the target board on the ground seen from the second camera.

Algorithm 3.2: *Calculating the relative position of two cameras*.

Input: An images *I* of a target board with a rhombus shape taken by the first omni-camera, and the adaptive mapping table of the first camera.

Output: The global coordinates (*x*, *y*) of the center of the rhombus shape in the upmost and leftmost intersection area of the two cameras.

Step:

Step 1. Load the image coordinates of the intersection points of the second camera, as illustrated by Figure 3.10.

Step 2. Put a specific target board on the uppermost and leftmost intersection point of the 2nd camera.

Step 3. Use the first camera to take an image of the target board.

Step 4. Calculate the central point *C* of the rhombus region on the target board in *I* taken by the first camera.

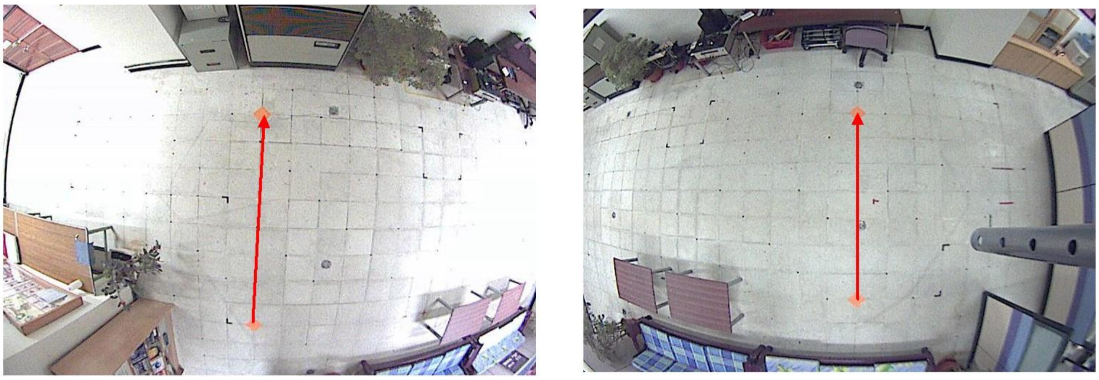
Step 5. Convert the image coordinates of *C* into global coordinates (*x*, *y*) by the mapping table of the first camera using Algorithm 3.1 as the desired output.

3.3.2 Calculating Relative Rotation Angle of Two Cameras

The relative rotation angle of the two cameras should also be calculated. In this phase we prepare two special target boards and put them on the ground. The target boards should be put on the centerline of the FOV's of the cameras as exactly as possible. Then, the center points of the target boards in the images can be calculated, and the vectors through the center points can be obtained, as shown in Figure 3.11.

Assume that the vector in image 1 is $\vec{p}$, and the vector in image 2 is $\vec{q}$. Then the relative angle θ of rotation can be calculated by the following formula:

$$\theta = \cos^{-1}\left(\frac{\vec{p} \times \vec{q}}{\|\vec{p}\| \times \|\vec{q}\|}\right). \tag{3.20}$$



(a) (b)

Figure 3.11 Calculating the vectors of the centers.

θ actually can be figured out to be the angle between the *center lines* (the *y*-axis of the ICS) of the two images as shown in Figure 3.12, and we use this value to modify the global coordinates of adaptive mapping table of the second camera.

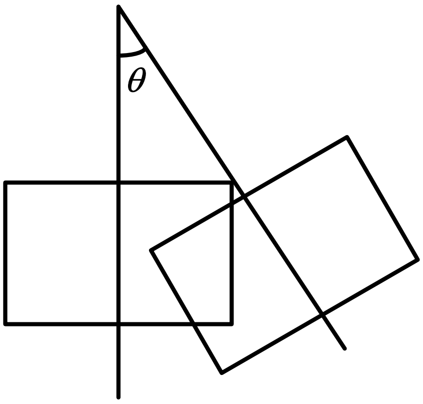


Figure 3.12 θ is the angle between the centerline of two images.

3.3.3 Calculating Coordinates in The GCS

After the relative position and rotation angle are calculated, the adaptive mapping table of the second camera can be calculated by the following method. The black points in Figure 3.13 are represented by the global coordinates in the mapping table of camera 1, and we want to calculate the global coordinates of the blue points in the region of camera 2 to construct the adaptive mapping table of camera 2.

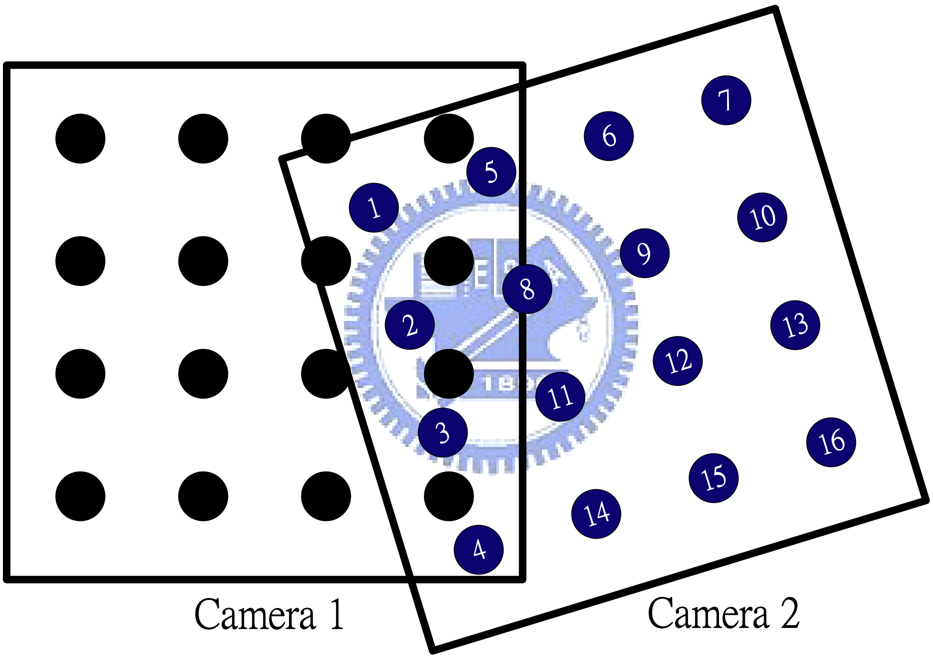


Figure 3.13 Calculating the adaptive mapping table of camera 2.

The coordinates (x, y), (x', y') and (x'', y'') in Figure 3.14 specify three blue points in Figure 3.13. The coordinates (x, y) specify any of the blue points, P_1, and the coordinates (x', y') specify another blue point P_h in the horizontal direction of P_1, and the coordinates (x'', y'') specify a third blue point P_v in the vertical direction of P_1. Symbol p specifies the length between P_1 and P_h, and θ is the rotation angle between the two cameras. Both p and θ are measured in advance. We can obtain four formulas to calculate (x', y') and (x'', y'') via (x, y) as follows:

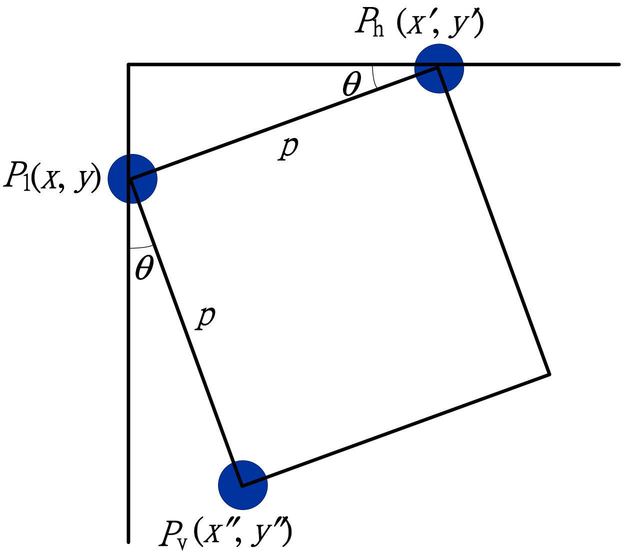


Figure 3.14 Calculating the global coordinates of points in the GCS.

Horizontal:

$$x' = x + p \times \cos\theta \quad (3.21)$$

$$y' = y + p \times \sin\theta \quad (3.22)$$

Vertical:

$$x'' = x + p \times \sin\theta \quad (3.23)$$

$$y'' = y + p \times \cos\theta \quad (3.24)$$

These four formulas are used to calculate all the global coordinates of points in the FOV of the second camera. First, the coordinates (x, y) of blue point 1 is calculated by the method described in Section 3.3.1. Then we may calculate the coordinates of blue points 2, 3 and 4 in Figure 3.13 via Equations 3.23 and 3.24 because they are on the vertical direction of P_1.

For example, when calculating point 2, p is $1 \times W_{real}$, and when calculating point 3,

p is 2×W_{real}, where W_{real} is calculated in advance as described previously. Afterward, the horizontal points in every row are calculated by use of the points of the first column. For example, when calculating the coordinates of point 8, because this point is on the horizontal direction of point 2, Equations 3.21 and 3.22 are used and the values of x and y in them will be the coordinates of point 2, and p will be still 1×W_{real}. The calculating sequence is represented in Figure 3.14 by white numbers in blue circular shapes. In this way, all the coordinates of points in camera 2 can be calculated, and so the adaptive mapping table can be completed.

After all the mapping tables of every camera are constructed, we consider all the mapping tables as a combined one, and so complete the expansion of the range of surveillance.

Chapter 4

Construction of Environmental Maps and Patrolling in Learned Environments

4.1 Introduction

One goal of this study is to make an autonomous vehicle to patrol in an indoor environment automatically. Patrolling points where the vehicles should navigate through for security monitoring are selected by a user freely. In order to achieve this goal, an environment map should be constructed, and the vehicle should have the ability to avoid static and dynamic obstacles, or a crash may happen. Besides, autonomous vehicles usually suffer from mechanical errors, and such errors will cause the vehicle to deviate from the right path, so an automatic path correction process is needed. The process should correct the position and direction of the vehicle at appropriate timings and spots.

4.2 Construction of Environment Maps

For an autonomous vehicle to patrol in a complicated indoor environment, the environment map should be constructed first. An environment map includes a two dimensional Boolean array in this study, and each element in the array represents a square centimeter in the global. If the value of an element in the array is *true*, that means there are some obstacles in this region, and the autonomous vehicle cannot go through this region. On the other hand, if the value of an element in the array is *false*, that means there is no obstacle in this region, and the autonomous vehicle can go through this region.

Furthermore, in this study we use multiple fish-eye cameras affixed on the ceiling to get the images of the environment, and use these images to construct the environment map. The major steps are described below.

1. Find the region of ground in taken images by a region growing technique.
2. Use the combined space mapping table to transform the image coordinates of the ground region into global coordinates to construct a rough environment map.
3. Eliminate broken areas in the rough environment map to get the desired environment map.

More details are described subsequently.

4.2.1 Finding Region of Ground by Region Growing Technique

A region growing method is used in this study to find the region of the ground, as mentioned previously. First, a seed is selected by a user from the ground part in the image as the start point, and the eight neighboring points of this start point are examined to check if they belongs to the region or not. The proposed scheme for this *connected-component check* will be described later. Then each of the points decided to belong to the region is used as the seed again, and the connected-component check is repeated, until no more region points can be found. More details of the method are described as an algorithm in the following.

Algorithm 4.1: *Finding the region of the ground in an image*.

Input: An image taken by a camera on the ceiling.

Output: The image coordinates of the region of the ground in the image.

Steps:

1 Select a seed P from the ground part in the image manually as the start point, and regard it as a ground point.

2 Check the eight neighboring points T_i of P to see if they belong to the region or not.

  2.1 Find all the neighboring points N_i of T_i which belong to the region of the ground for each T_i.

  2.2 Calculate the value of the similarity degree between T_i and each N_i.

  2.3 Decide whether T_i belongs to the ground region according to the similarity values by the following steps.

    2.3.1 Compare the values of similarity calculated in Step 2.2 with a threshold TH_1 separately (the detail will be described later).

    2.3.2 Calculate the number p of similarity values which are larger than TH_1.

    2.3.3 Calculate the number q of similarity values which are smaller than or equal to TH_1.

    2.3.4 Compare p with q, and if the value of p is larger than q, then mark the point T_i as not belonging to the region and go to Step 2 to process the next T_i; else, continue.

    2.3.5 Calculate the similarity degree d between T_i and the average *RGB* values of all pixels in the region of the ground (the detail will be described later).

2.3.6 Compare the similarity degree d with another threshold TH_2, and if d is smaller than TH_2, then mark T_i as belonging to the ground region; else, mark T_i as not.

2.4 Gather the points B_i which belong to T_i and belong to the region of the ground.

3 If there are some points of B_i which are not examined yet, then regard each B_i as a seed P and go to Step 2 again to check if they belong to the ground region or not.

In Steps 2.1 and 2.2, when a point T_i is examined, all the neighboring points N_i of T_i which have already been decided to belong to the ground region are found out first, and a similarity degree between T_i and each of its eight neighboring points, as shown in Figure 4.1, is computed. The similarity degree between two points A and B is computed in the following way:

$$\text{similarity between } A \text{ and } B = |\, r_A - r_B \,| + |\, g_A - g_B \,| + |\, b_A - b_B \,| \qquad (4.1)$$

where r_C, g_C, and b_C are the color values of point C with $C = A$ or B here.

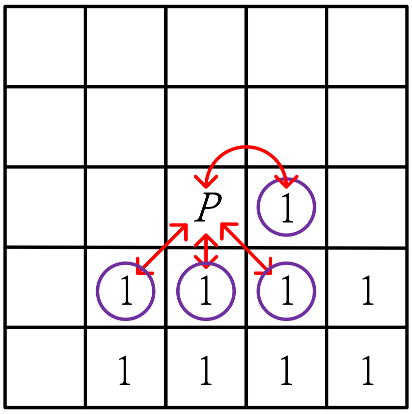


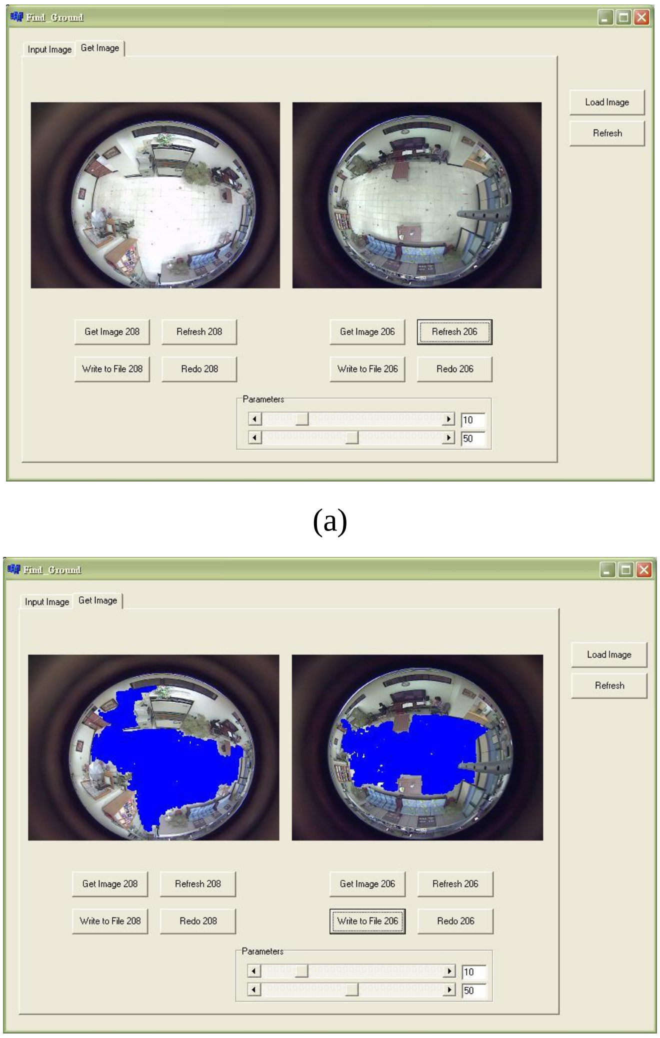
Figure 4.1 Illustration of calculation of the similarity degree between two image points.

In Step 2.3.1, after the similarity degree is calculated, the degree is compared with a threshold TH_1, whose value may be adjusted by a user. If the value is large, then the scope of the ground region which is found will be enlarged; else, reduced.

In Steps 2.3.2 and 2.3.3, the two introduced values p and q are set to zero at first. The value of p represents the number of points whose similarity degree is larger than TH_1, and the value of q represents the number of points whose similarity degree is not so. Hence, if a degree is larger than TH_1, then we add one to p, and if the degree is not so, then we add one to q. Afterward, in Step 2.3.4, if the value of p is larger than q, the point T_i is marked as not belonging to the region, and then go to Step 2 again to check the next T_i. If the value of p is not so, then an additional iterative process is conducted to examine T_i.

Sometimes the boundary between the region of the ground and obstacles is not very clear in images. So in Step 2.3.5, an average values *AVR* is calculated first, which contains 3 values, namely, the average values $R_{avr},$ G_{avr} and B_{avr} of red, green, and blue values, respectively, of all the pixels in the ground region. We use *AVR* to decide whether the pixel T_i belongs to the ground region or not. The similarity degree d between the point T_i and *AVR*, as mentioned in the step, is then calculated according to a similar version of Equation 4.1. In Step 2.3.6, d is compared with another threshold TH_2. If d is smaller than TH_2, then the point T_i is marked as belonging to the region; else, as not.

In Step 2.4, the points B_i which belong to T_i and belong to the region of the ground are found, and in Step 3, these points are regarded as seeds and Step 2 is repeated again to check if these points belong to the region or not. No matter whether the point T_i belongs to the region or not, the point will be marked as *scanned*. An example is shown in Figure 4.2. The two images in Figure 4.2(a) are taken by two fisheye cameras separately, and in Figure 4.2(b) the blue regions are the ground regions found by the above algorithm.



(a)

(b)

Figure 4.2 An example of finding the region of the ground.

4.2.2 Using Image Coordinates of Ground Region to Construct a Rough Environment Map

After the image coordinates of the ground region are found in every image, we can utilize the coordinates to construct a rough environment map by the following method.

Because the mapping tables of the fisheye cameras are constructed in advance, coordinates can be converted between the ICS (image coordinate system) and the GCS (global coordinate system) freely by Algorithm 3.1. And the size of an environment map is defined in advance, so we can convert the global coordinates G_i of each point in the map into image coordinates I_i first.

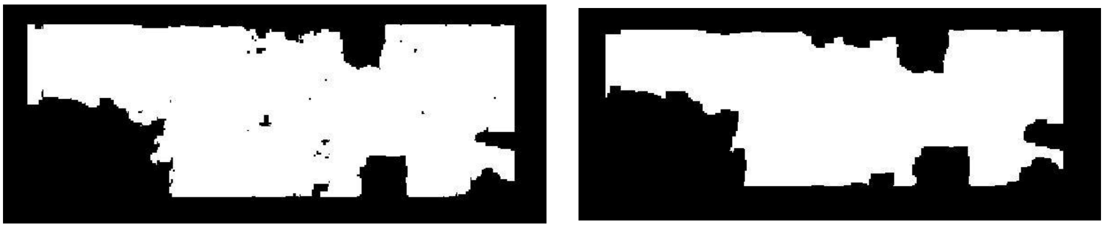
Also, because all the image coordinates of the ground region R_g have been found by Algorithm 4.1 in Section 4.2.1, we can check them to see if the image coordinates I_i specify a point p_i belonging to R_g or not. If p_i belongs to R_g, then we indicate the space point P_i of the global coordinates G_i in the map to be an open way otherwise, to be an obstacle. Hence we can obtain the global coordinates of all the obstacles in the environment, and so can indicate the positions of the obstacles in the map to construct a rough environment map.

4.2.3 Eliminating Broken Areas in a Rough Environment Map

After constructing a rough environment map, there will be a lot of *broken* areas in the map, so a refining process is applied to eliminate these broken areas. The refining process includes two major operations – erosion and dilation. An erosion operation can eliminate the noise in the map, and a dilation operation can mend unconnected areas to make the map smooth.

The erosion operation examines every element in the rough map, and if the value of an element *E* is true, that means there are some obstacles in this region *R*. A mask with *n*×*n* size will then be put on *R* and every element in this mask will be examined to see if the values of them are *true* or *false*. The values of elements in the mask then are gathered to decide the new value of *E*. If the number of the value of *true* is larger than half of the number of elements in the mask, then the new value of *E* will be set *true*; otherwise, the value of *E* will be set *false*. In another situation, if the value of *E* is *false* originally, the new value of *E* will be *false*, too. Because of the property of this erosion operation, the noise in the map will be eliminated.

The dilation operation can expand the region of every obstacle, so if there is a little gap between two regions of obstacles, after doing the dilation operation, the gap will be mended. The dilation operation also scans all of the elements in the map, and if the value of element *R* is *true*, another mask with size *m*×*m* is put on *R*. Afterward, the value of every element which is in the mask are set to *true*, hence every obstacle in the map will be expanded, and the gap will be mended. The degree of expansion depends on the size of mask, that is, if *m* is large, then the degree of expansion will be high. An example of results is shown in Figure 4.3.



(a) (b)

Figure 4.3 An example of refining a map. (a) A rough map. (b) A complete map after applying the erosion and dilation operations.

4.3 Avoiding Static Obstacles

After constructing the complete environment map, we can know exactly the positions of the obstacles in an indoor environment. When an autonomous vehicle navigates in the environment, it should avoid all of the obstacles automatically. When there are some obstacles on the original path from a starting point to a terminal point, the system should plan a new path to avoid the obstacles, and the new path will satisfies the following constraints:

1. The shortest path should be chosen.
2. The turning points should be within the open space where no obstacle exists.
3. The number of turning points should be reduced to the minimum.

The method we use in this study is to find several turning points to insert between the original starting point and the terminal point. Here, by a turning point, we mean one where the autonomous vehicle will turn its direction. The lines connecting these points will be the new path. As shown in Figure 4.4, the purple points are the original starting and terminal points, and the red line is the original path.

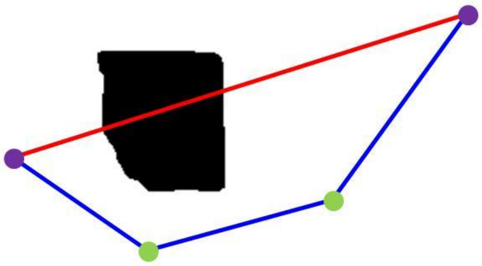


Figure 4.4 Illustration of computing turning points to construct a new path (composed by blue line segment) different from original straight path (red line segment).

But by the environment map, there are some obstacles on the straight path from the starting point to the terminal one, so turning points should be calculated to construct a new path. The green points are the computed turning points, and the blue lines which connect the turning points compose the new path. By the following algorithm, the turning points can be calculated.

Algorithm 4.2: *Calculating turning points to construct a path through no obstacle*.

***Input*:** An environment map *M*, a starting point *S*, and a terminal point *T*.

***Output*:** The coordinates and the number of turning points in the GCS, through which a path from *S* to *T* encounters no obstacle.

***Steps*:**

Step 1. Calculate the equation of the line *L* connecting by *S* and *T* using the coordinates of *S* and *T*.

Step 2. Check whether there are obstacles O_i on *L*, where *i* is the number of obstacles on *L*. If not, the final path is *L* and the algorithm is finish. Otherwise, continue.

Step 3. Calculate the overlapping regions R_i of *L* and the obstacles.

Step 4. Calculate the perpendicular bisectors of each R_i.

Step 5. Extend each bisector until reaching another obstacle or the boundary of map, and compute two intersection points O_i' and O_i'' when reaching another obstacle or boundary.

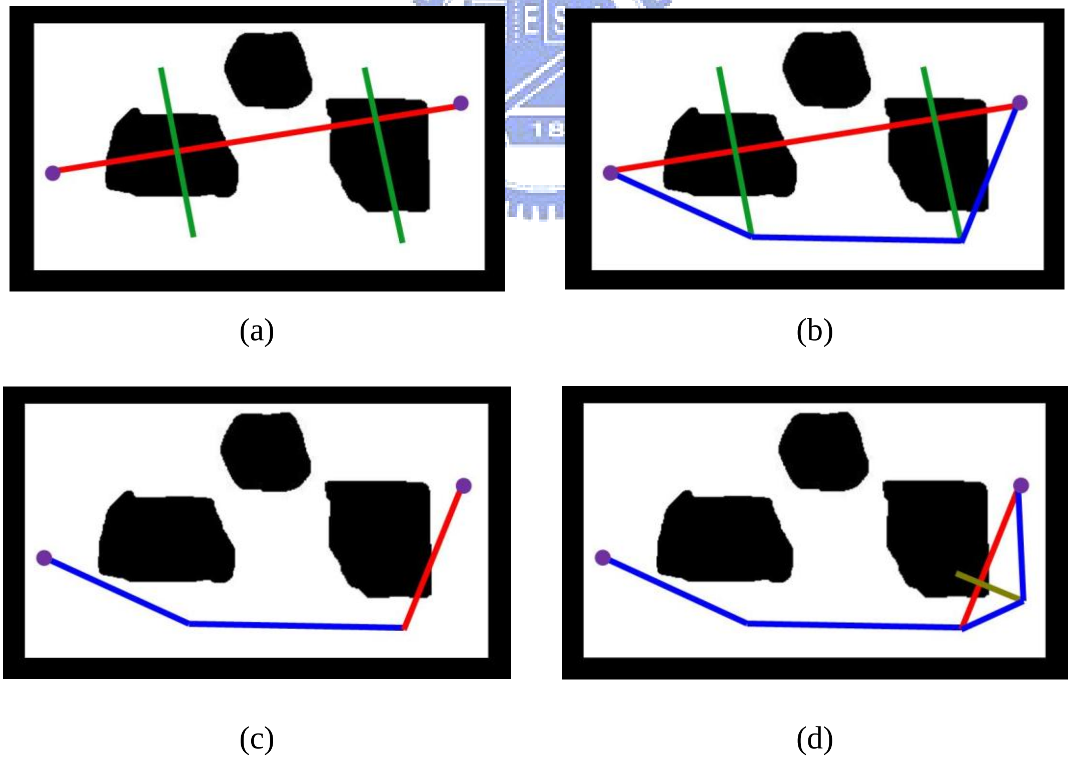
Step 6. Calculate the middle point C_i' between O_i and O_i' for each *i*.

Step 7. Calculate the middle point C_i'' between O_i and O_i'' for each *i*.

Step 8. Compare the length of a path connecting *S*, C_i' (for all *i* in the sequence) and *T* with the length of another path connecting *S*, C_i'' (for all *i* in sequence) and *T*, and choose the path with the shorter length.

Step 9. If there is no obstacle on the new path, then output the turning points on the new path. If there are some obstacles on the sub-paths, then recursively apply this algorithm to each sub-path with obstacles.

After the turning points are calculated, an alternative path is found. The process is illustrated in Figure 4.5. In Figure 4.5(a), the purple points are the starting and terminal points, the red line is the original path, and the green lines are the perpendicular bisectors. In Figure 4.5(b), the first alternative path is found by connecting the turning points, but there are still some obstacles on a sub-path of it, as shown in Figure 4.5(c). So the algorithm is applied again on the sub-path, and the final alternative path is calculated to be as that shown in Figure 4.5(d).



(a)

(b)

(c)

(d)

Figure 4.5 An example of planning an alternative path. (a) The original path with some obstacles on it. (b) The calculated new path but still some obstacles on its sub-path. (c) Repeat the algorithm to the sub-path. (d) The calculated final path.

4.4 Patrolling in Indoor Environment

After an environment map is constructed and a path is planned, the autonomous vehicle can navigate in the indoor environment automatically. But after a while, the vehicle will start to deviate from its path because of accumulated mechanic errors. So the farther the vehicle goes, the more the deviation will become. Hence we should correct the position of the vehicle. Besides, there may be some dynamic obstacles in the environment. When an autonomous vehicle patrols in an environment, it should avoid these obstacles, or a crash might happen. Furthermore, the FOV's of a fisheye camera is finite. In order to expand the range of surveillance, we use multiple fisheye cameras in this study. So the solution to the hand-off problem will be described in this chapter, too. In order for an autonomous vehicle to patrol in an indoor environment automatically and continuously, we should solve the following problems.

1. Correcting the global coordinates of the vehicle automatically.
2. Avoiding dynamic obstacles automatically.
3. Patrolling under several cameras.

The solutions proposed in this study are described in the rest of this chapter.

4.4.1 Correcting Global Coordinates of Vehicle Automatically

Because an autonomous vehicle suffers from mechanical errors, the correction of the position is needed. Furthermore, the error of the direction of a vehicle will be accumulated. In this study, we propose four correction strategies to correct the position and direction of a vehicle. The four strategies are:

1. position correction;
2. direction correction;
3. enforced direction correction;
4. ending path segment correction.

A. Vehicle location correction strategy 1 --- position correction

When a vehicle navigates to a turning point, it should correct its global coordinates. Because a turning point is a destination of a path segment, in order to avoid the accumulation of mechanical errors in this path segment, the position of the vehicle must be corrected at the turning point. In this study, images taken by the fisheye camera are used to calculate the exact position of a vehicle. First, when a vehicle navigates to a turning point, we can get the odometer values of the vehicle, which include the *x*-coordinate and the *y*-coordinate of the vehicle, and the direction angle of the vehicle. In this strategy, the *x*- and *y*- coordinates will be corrected. After getting the values of the odometer, we can know the approximate position of the vehicle. Then we judge which camera should be used to get an image of the vehicle and calculate the center point of the vehicle shape in the image. When the center point is calculated, because the approximate position of the vehicle is known, only a partial image neighboring to the vehicle position need be processed. The steps are shown in Figure 4.6.

The method we propose to calculate the position of the vehicle is described in the following. Because the odometer values are expressed as global coordinates, the values of the position in the odometer should be converted into image coordinates

C_{img} using the corresponding mapping table. Then the partial image of the square region whose center is located at C_{img} is subtracted from the background image. The size of the square region is changeable and is small at the beginning. The background image is an image which is taken by the same camera in advance, so there is no dynamic obstacle in there. After the subtraction of two partial images, the pixels which have big differences between the foreground and background will be found out. Afterward, the erosion and dilation operations mentioned previously are applied to eliminate the noise and smooth the edges in the image. Then, a connected component in the image is segmented out, and the average of all the coordinates in the component is computed to be the new coordinates C_{img} of the vehicle shape center.

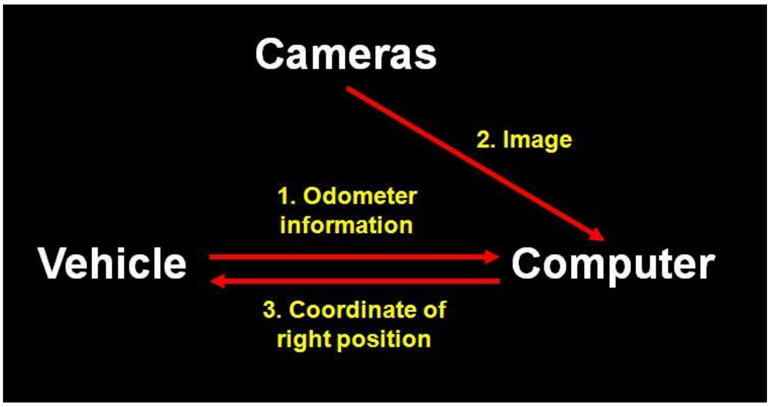


Figure 4.6 Steps of correcting the position of vehicle.

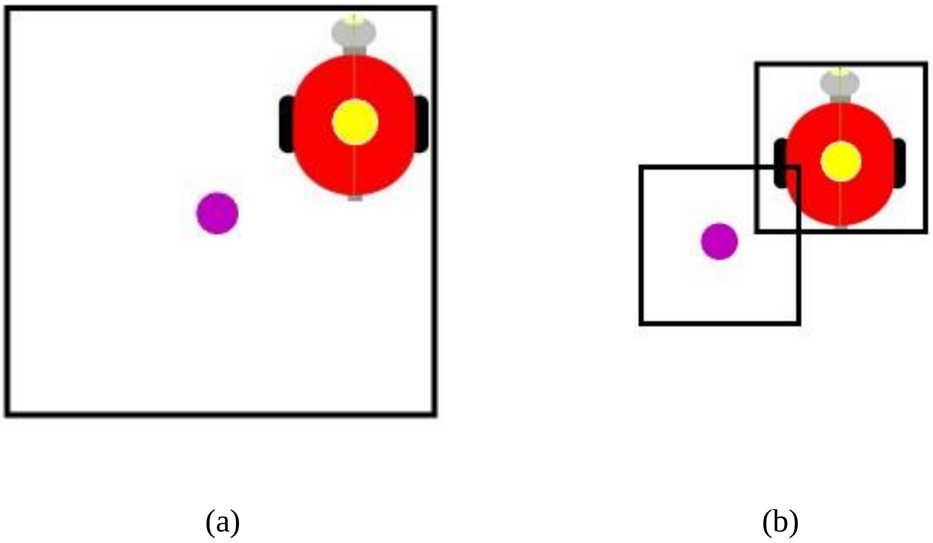
If the region of the whole vehicle is not found out, then the steps mentioned above will be done again with the new C_{img} and a square–regioned tracking window with a bigger size. After the region of the whole vehicle is found, the exact center point of the vehicle is the newest C_{img}. The size of the square region will become bigger and bigger when every iteration of the steps is done, and every C_{img} will become closer and closer to the real center point of the vehicle.

We use a tracking window with a changeable size to find the centroid of the vehicle in this study. The advantages of using a tracking window with a changeable size include:

- avoiding the influence of other dynamic obstacles or people;
- increasing the precision of the vehicle region detection;
- speeding up computation.

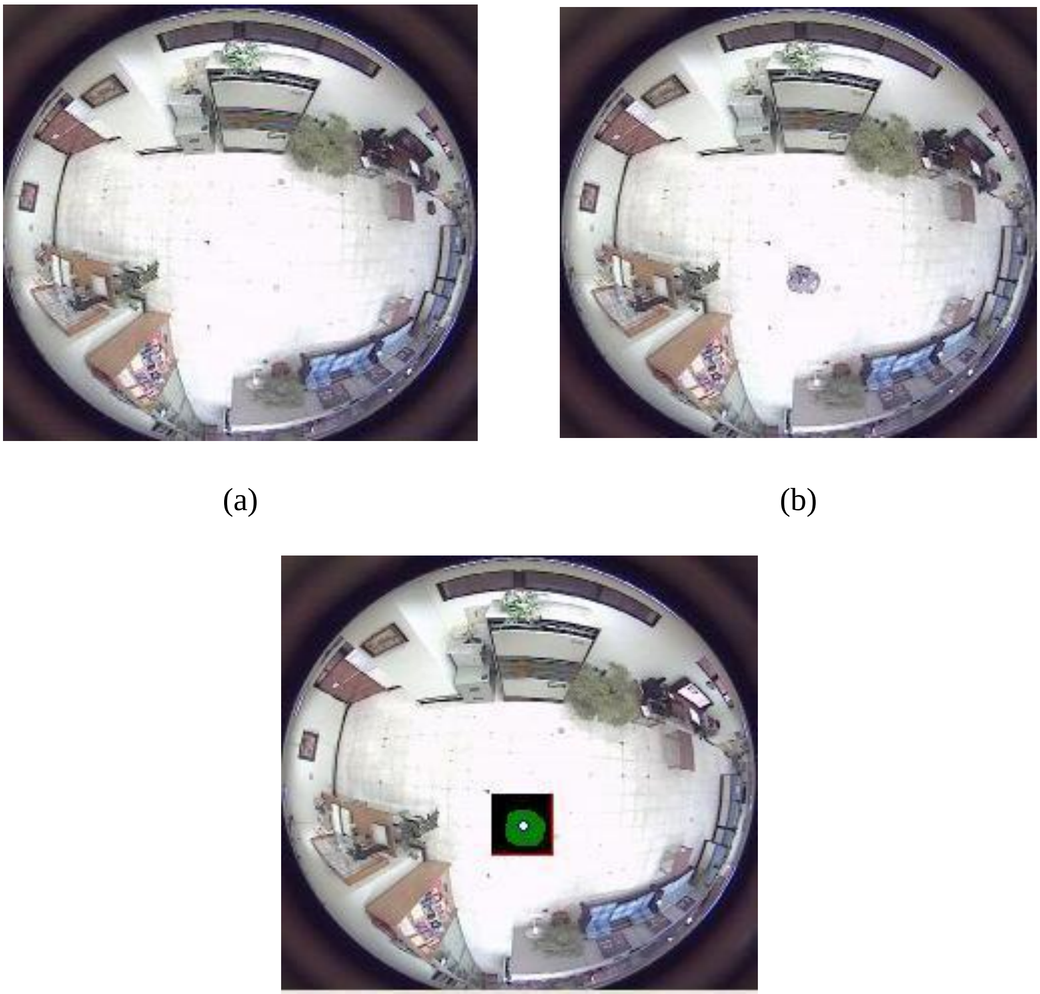
Because we use fisheye cameras in this study, the size of a vehicle in images will change when the vehicle navigates to a different position. If the size of square region is changeable, the area which has to be processed will be much smaller than using a square area with a fixed size. When the processed region is smaller, the probability of interfering by dynamic obstacles will decrease. The processed region with a square mask with a fixed size and a mask with a changeable size are shown in Figure 4.7. The yellow point is the real central point of a vehicle, and the purple point is the position recorded in the odometer with mechanic errors. The total region with a changeable mask is smaller than the region with a mask with a fixed size.

The process of calculating the exact position of a vehicle is described in Algorithm 4.3, and a real example of finding the central point of a vehicle is shown in Figure 4.8.



(a) (b)

Figure 4.7 The calculating region. (a) with a fixed square mask. (b) with a changeable mask.



(a)

(b)

(c)

Figure 4.8 Finding the central point of a vehicle. (a) the background image of an environment. (b) The foreground image with a vehicle. (c) The found central point of a vehicle.

Algorithm 4.3: *Calculating the exact position of a vehicle*.

***Input*:** An image taken by a fisheye camera, and the coordinates of the vehicle position recorded in the odometer.

***Output*:** The exact position of the vehicle.

***Steps*:**

Step 1. Convert the coordinates in the odometer to the ICS and obtain the image coordinates C_{img}.

Step 2. Subtract the partial image in the square region whose center is located at C_{img} from the background image to find the pixels which have big difference values.

Step 3. Apply the erosion and dilation operations to the image of the region.

Step 4. Find the connected component in the image.

Step 5. Calculate the average of all the coordinates in the component and obtain new C_{img}.

Step 6. If the region of the whole vehicle is found out, then go to Step 3; otherwise go to Step 2 with the new C_{img} and a square region with a bigger size.

Step 7. Convert the new C_{img} to the GCS by the method mentioned in Chapter 3 using the mapping table.

B. Vehicle location correction strategy 2 --- direction correction

Mechanical errors not only cause inaccuracy in vehicle positions, but also inaccuracy in vehicle directions. The technique of this strategy is to correct the direction errors. Because the exact positions of a vehicle are calculated at every turning point in Strategy 1, we can use two consecutive positions to calculate the vector of the direction, as shown in Figure 4.9.

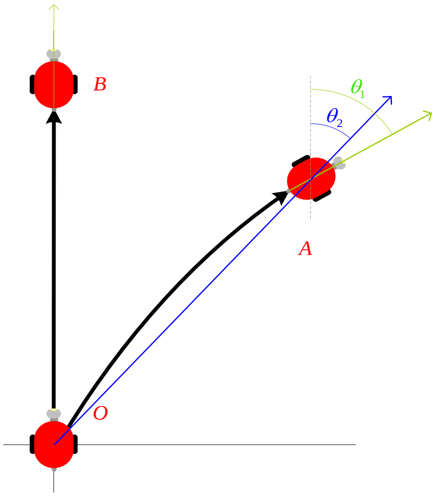


Figure 4.9 Using two consecutive positions to calculate the vector of direction.

But when the path between two consecutive turning points is too long, the mechanical errors will be large. Take $\overrightarrow{OA}$ in Figure 4.9 as an example. The angle calculated by use of the line through points O and A is θ_2, but the exact angle should be θ_1. So if we want to limit the error of the vehicle direction within a certain degrees, say α, a limit of the length of the path between two turning points can be calculated by the following method. The approximating function of direction errors of a vehicle can be obtained by experiments. The function of the error of the vehicle which we use in this study is (Tsai and Tsai [17]):

$$f(x) = 0.00000476X^2 + 0.00592048X + 4.16437951. \tag{4.2}$$

where x is the length of the patrolling path of the vehicle, and $f(x)$ is the error function of the direction in degrees. By the following formula of the solution to a quadratic equation:

$$\begin{aligned} & f(x) = ax^2 + bx + c = 0 \\ & \Rightarrow x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \end{aligned} \tag{4.3}$$

we can obtain the positive solution as

$$\begin{aligned} & 0.00000476x^2 + 0.00592048x + 4.16437951 < \alpha \\ & x < \frac{\sqrt{0.00001904\alpha - 0.00004424} - 0.00592048}{0.00000952} \cdot \end{aligned} \tag{4.4}$$

For an example, if we want to limit the error of the vehicle direction to be within 5 degrees, then the correction of the direction is done when the length of a path is smaller than

$$\frac{\sqrt{0.00001904 \times 5 - 0.00004424} - 0.00592048}{0.00000952} = 127.96 \text{ cm.} \tag{4.5}$$

But when the length of a path is too short, another problem will happen. Because there are some errors when we calculate the center of a vehicle in an image, when the length of a path is too short, such errors will become large relatively. As shown in Figure 4.10, if the points *a* and *b* are the exact center points of the vehicle in two consecutive images, and the calculated centers are *a*′ and *b*′, then the error of the vehicle angle will be φ.

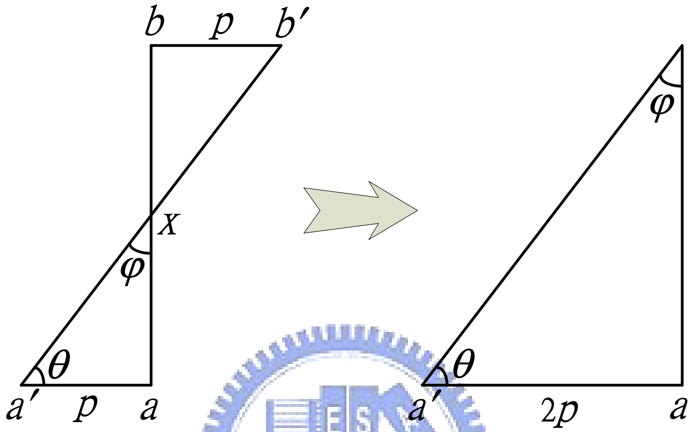


Figure 4.10 False direction caused by the error of finding center.

If we want to limit the error of the vehicle direction to be within α degrees, and the maximum error of finding the center of a vehicle in an image to be *p* cm, then we can obtain another limit of the length of the path by the following equations:

$$\tan\theta = \frac{x}{2p}$$
$$\Rightarrow \theta = \tan^{-1}\left(\frac{x}{2p}\right) \tag{4.6}$$

$$\text{and } 90^\circ - \theta < \alpha\,. \tag{4.7}$$

$$\Rightarrow 90 - \tan^{-1}\left(\frac{x}{2p}\right) < \alpha \tag{4.8}$$

$$\Rightarrow \tan^{-1}\left(\frac{x}{2p}\right) > 90 - \alpha \tag{4.9}$$

$$\Rightarrow \frac{x}{2p} > \tan(90 - \alpha) \tag{4.10}$$

$$\Rightarrow x > \tan(90-\alpha)\times 2p\,. \qquad (4.11)$$

For an example, if we want to limit the error of the vehicle direction to be within 5 degrees, and the error of finding the center of the vehicle in an image to be 3 cm, then the correction of the direction is done when the length of a path is limited to be larger than

$$\tan(90-5)\times 2\times 3 = 68.6\,\text{cm}. \qquad (4.12)$$

By Equations 4.4 and 4.11 derived above, if the length of a path is between the two values V_{max} and V_{min} calculated by these two equations, then the correction of the vehicle direction will be done.

C. Vehicle location correction strategy 3 --- enforced direction correction

Although in Strategy 2 the direction of the vehicle has been corrected, if the length of a path is not between V_{max} and V_{min}, then the correction of the direction cannot be done. So we set a threshold M, and if the accumulation of the length of the vehicle movement exceeds M and the length L of the next path is not between V_{max} and V_{min}, then the vehicle will be enforced to do the correction of its direction by the following method.

First, the next path is divided into two segments. The length of the first segment is set to be ($V_{max}+ V_{min}$)/2, and the length of the second segment is taken to be L – ($V_{max}+ V_{min}$)/2, as shown in Figure 4.11. When the vehicle navigates from point *a* to *b*, the correction of the vehicle direction will be done by the method mentioned in Strategy 2, because the length of this segment is between V_{max} and V_{min}.

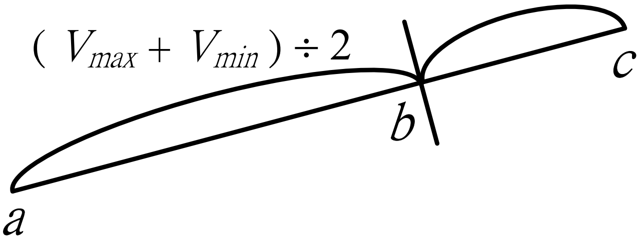


Figure 4.11 Separating the path into two segments.

D. Vehicle location correction strategy 4 --- ending path segment correction

Because of the mechanical errors, if the length of one path segment is very large, a vehicle will suffer from large errors at the end of this path segment. The vehicle may so not be able to monitor the exact place where we want to keep under surveillance. So if the distance between the original destination and the real position of a vehicle is larger than a value which is set in advance, then we will plan another path from the real position of the vehicle to the original destination.

4.4.2 Avoiding Dynamic Obstacles Automatically

When a vehicle patrols in an indoor environment, there may also be some *dynamic* obstacles in the environment, such as humans or furniture. The vehicle should avoid these obstacles, too. First, the image coordinates of the set *R* of the regions of all objects in the environment, including the vehicle and dynamic obstacles, are calculated by subtracting the foreground from the background. Because the entire region of a vehicle in an image can be found out by the method mentioned in Section 4.4.1, and a vehicle is not an obstacle, we remove the region of the vehicle from *R*, and convert the image coordinates of the other regions into global coordinates.

Afterward, because the global coordinates of all dynamic obstacles have been acquired, we can indicate them on the environment map to obtain a new map, and a new path can be planned by the method described in Section 4.3 to avoid all the obstacles.

Algorithm 4.4: *Constructing new map and path for avoiding dynamic obstacles*.

***Input*:** An environment map *M*, an image *I* which is taken by an omni-camera.

***Output*:** A new environment map *M*′ with dynamic obstacles, and a new path to avoid the dynamic obstacles.

***Steps*:**

Step 1. Find the entire region of the vehicle in image *I*.

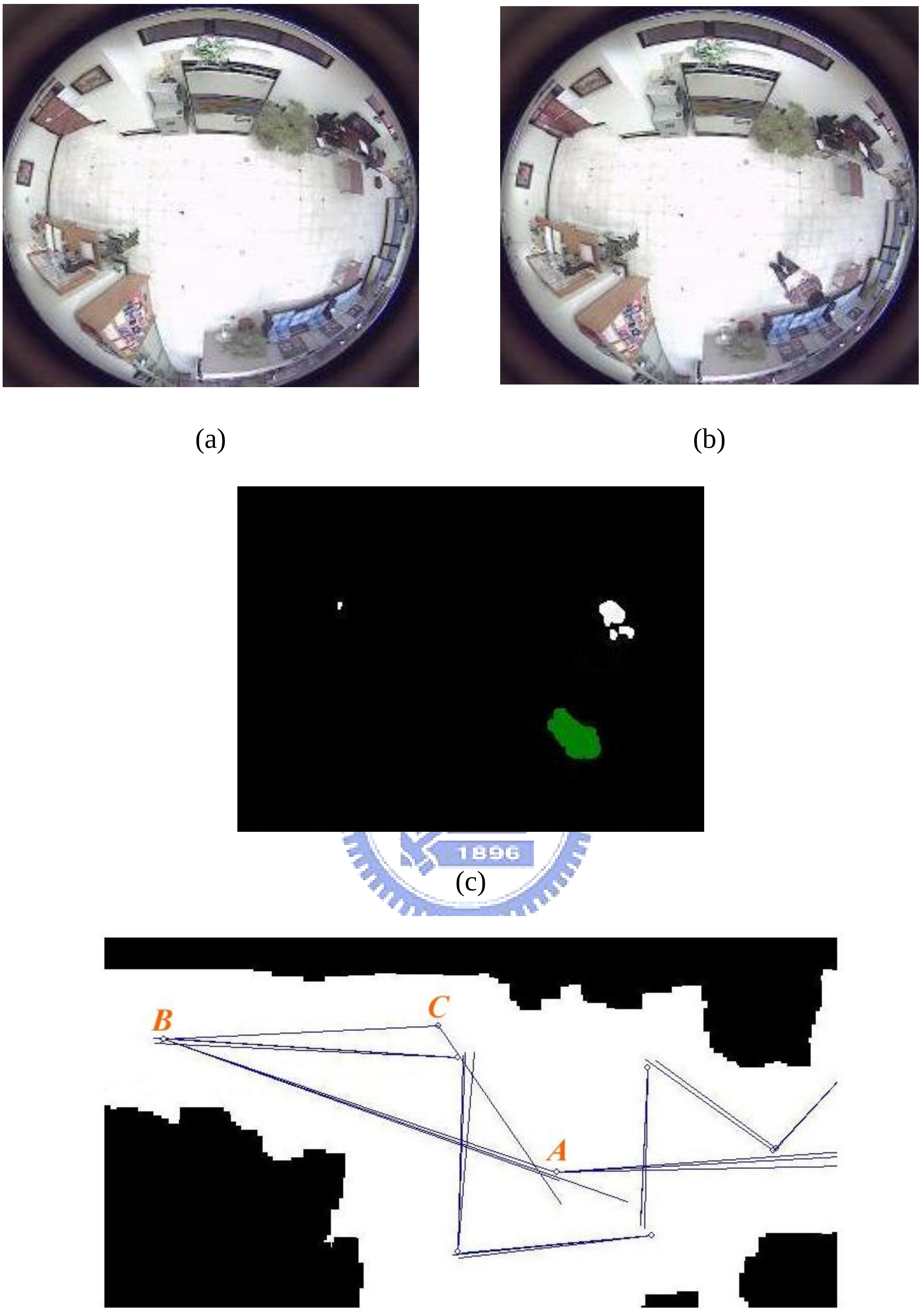
Step 2. Subtract *I* from the background to find all the non-vehicle objects in image *I*.

Step 3. Locate the non-vehicle objects in real space.

Step 4. Indicate all objects on the map *M* to obtain a new map *M*′.

Step 5. Plan a new patrolling path by Algorithm 4.2 to the next destination on *M*′.

An example is shown in Figure 4.12, Figure 4.12(a) is the background of an environment, and in Figure 4.12(b) a person stands in the environment. In Figure 4.12(c), the region of the person is found by subtracting the foreground from the background. In Figure 4.12(d), the original path is from point *A* to *B*, but a new path is planned from *A* through *C* to *B* in order to avoid the dynamic obstacle.



(a)

(b)

(c)

(d)

Figure 4.12 An example of avoiding dynamic obstacles. (a) The background image of an environment. (b) The image with a person stands in the environment. (c) The region of the person is found. (d) A new path is planned to avoid the dynamic obstacle.

4.4.3 Patrolling Under Several Cameras

In order to expand the range of surveillance, we use multiple fisheye cameras in this study, so the vehicles will patrol under several cameras. This creates a problem of *hand-off* between cameras, but the problem here can be simplified by the odometer in a vehicle, because the odometer value indicate the approximate position of a vehicle. The mapping tables of the fisheye cameras are constructed in advance, so the range of surveillance of every camera is known. As shown in Figure 4.13, point *A* is the uppermost and leftmost point in Camera 2, and point *B* is the lowermost and rightmost coordinates in Camera 1. The coordinates of these two points are known by the mapping tables of the two cameras, hence the coordinates of points *C* and *D* can be calculated, and afterward the equation of line *L* can be obtained. Taking the values O_x and O_y of the odometer as variables into the equation of line *L*, we can judge which camera should be used to get the image of the vehicle. The equation of *L* can be computed by:

$$(y_2 - y_1)x + (x_2 - x_1)y = x_2 y_2 - x_1 y_1, \qquad (4.13)$$

and the judgment is conducted as follows:

If $(y_2 - y_1)O_x + (x_2 - x_1)O_y - x_2 y_2 + x_1 y_1 \geq 0$, then the camera 1 should be used; otherwise, the camera 2 should be used.

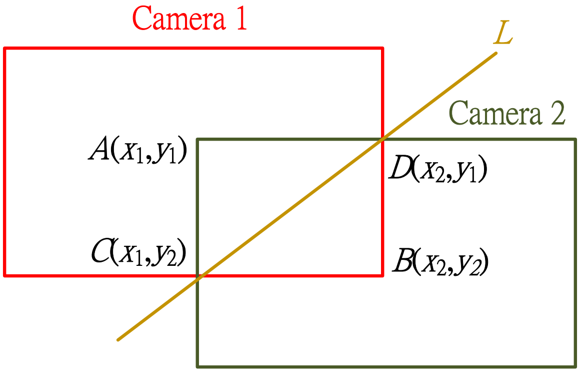


Figure 4.13 Line L separates the region of two cameras.

Chapter 5

Following Suspicious People Automatically and Other Applications

5.1 Introduction

Although there are several fisheye cameras on the ceiling, they cannot move their positions and cannot take images of a person at different angles. Besides, by fisheye cameras sometimes we cannot take clear images of the face of the person. In this study, a technique is designed to follow a suspicious person who breaks into the indoor environment which is under surveillance. Because the autonomous vehicles are highly mobile and can take clear images by the cameras equipped on them, they are proper to assist the surveillance in the indoor environment.

In order to let the vehicles follow a person, the position of the person should be calculated first. In this study, we only calculate a specific partial region, called the tracking window, in an image to decrease the amount of calculation and decrease the probability of interference by other dynamic obstacles. Afterward, the distance and angle between the vehicle and the person are calculated. Then, the central computer can order the vehicle to chase the person. The detail process will be described in Section 5.2.

Because only the partial image is calculated, and the person will move in the environment, hence the next position of a person should be predicted in order to decide the region which should be calculated. But the images are taken by the fisheye cameras, so there will be high degrees of distortion in the images. The position of prediction should be modified when the person appears in different places in the image. The detail process will be described in Section 5.3.

Because the FOV's of one camera is finite, multi-cameras are used in this study to expand the range of surveillance. When a person moves from the region of one camera to that of another, it is necessary to switch the camera to get the images of the vehicle automatically. The detail process will be described in Section 5.4. And other applications, such as recording the track of a person in the computer or calculating the walking speed of the person, are described in Section 5.5.

5.2 Calculating Position of a Person by Specific Partial Region in an Image

One goal in this study is to calculate the position of a person intruding into a space under surveillance. Although we can subtract the foreground from the background to find out all the objects in the environment, the amount of calculation will be very large. Hence we only calculate the specific partial region in an image. Besides reducing the amount of calculation, there are other advantages to do this: decreasing the probability of interference by other dynamic obstacles, and increasing the preciseness of calculating the position of the target.

First, the region of the door in a room space under surveillance is set in advance, and this region will be kept under close monitoring continually. When the door is opened, the foreground image of that region will change, so we can know the door is opened by subtracting the foreground image from the background image in the region of the door. After the door is opened, the process of following any suspicious person will start.

When tracking a person, we use the same idea as finding the central point of a vehicle in Chapter 4. Only the pixels which are in a square region *R* will be calculated, and the size of *R* is changeable. The detail of the process is described in Algorithm 5.1. The coordinates *C* of a certain point is introduced here. It represents the position where we predict the person should be at this time, and the detail process of calculating *C* will be described in Chapter 5.3.

Algorithm 5.1: *Calculating the position of a person*.

Input: An image which is taken by the fisheye camera, and the coordinates *C* of a predicted location of a person.

Output: The coordinates of the exact position of the person.

Steps:

Step 1. Calculate the range of the square region *R* by the use of coordinates *C*, at which the center of a region *R* is located, and the size of the region is small at first.

Step 2. Subtract the partial foreground image in *R* from the background image to find the pixels which have big differences between the two images.

Step 3. Apply the erosion and dilation processes to the image portion in *R*.

Step 4. Find the connected components in *R*.

Step 5. If there are some components whose sizes are larger than a threshold *N* in *R*, then go to Step 6; otherwise, go to Step 2 again with the square region *R* being enlarged to be with a bigger size.

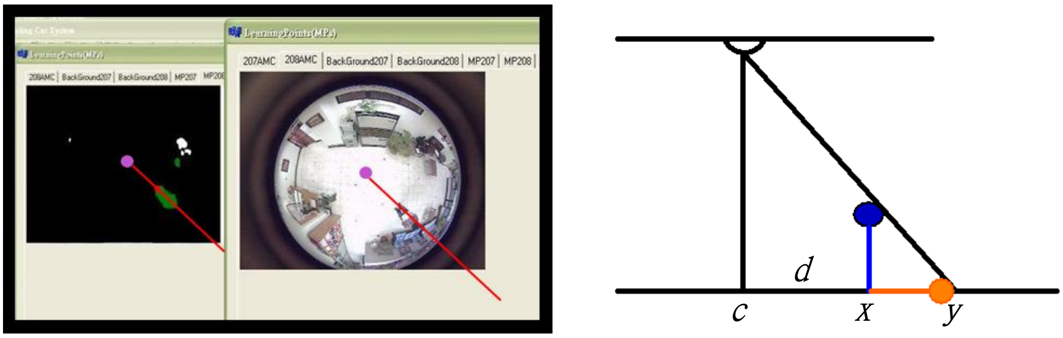
Step 6. Calculate the distances between all the points in the component and the center of the image.

Step 7. Find the point whose distance calculated in Step 6 is the smallest.

Step 8. Convert the point found in Step 7 to the global coordinates as output.

In Step 5, we set a threshold *N* in advance, and if the size of a component is larger than *N*, then we consider the component as an object. When there is no object in the square region *R*, the size of *R* will be increased, and Step 2 is repeated again with the new size. When there are some objects in *R*, we calculate the distances between all the points in the component and the center of the image, and find the point whose distance is the smallest. The reason is described below.

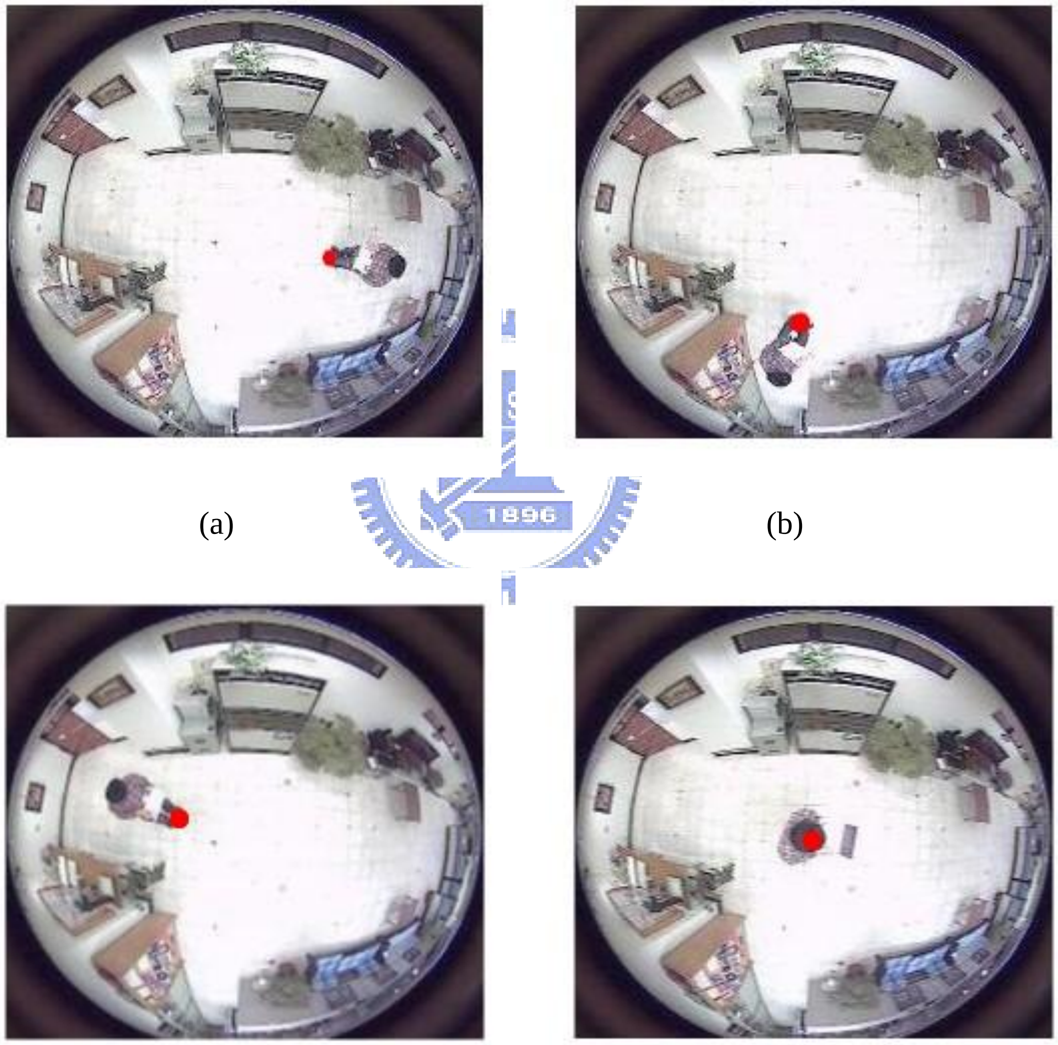
We find the person's feet by the rotational invariance property of omni-cameras. More specifically, it can be proved that the line through the human body (actually through the human's head and feet) always passes the image center according to the above property, as shown in Figure 5.1.



(a) (b)

Figure 5.1 The rotational invariance property. (a) A real example. (b) A vertical sketch chart.

If a person stands at point *x*, a fisheye camera is affixed on the ceiling to take images of the person, the center of the image taken by the camera is *c*, and the distance between *c* and *x* is *d*, then the region of the projection will be from *x* to *y*, as seen laterally. So if we want to find the position where the person stands, we should check all the pixels in the region to find out the point whose distance to *c* is the smallest, namely, the point *x* with the smallest distance *d*. Several real examples are shown in Figure 5.2, where the red points are the calculated positions of the tracked person.



(a) (b)

(c) (d)

Figure 5.2 Finding the positions of a person. (a)(b)(c) The person stands at several different places in an image. (d) The person stands at the center point in an image.

5.3 Predicting Position of A Person

Because a person may move in the environment continuously and we only want to calculate the specific partial region in an image of the person, the position C of a person in the next period of time should be predicted. Then, the region for tracking the person can be decided by C and Algorithm 5.1.

Because the positions of a person in every period of time should be calculated continuously and the length of every period of time is set equal, a simpler solution to the person prediction problem is to extend the line of two older positions of the person to predict the future position of the person. As shown in Figure 5.3, if the person moves from point A to point B in sequence, and the distance between A and B is D_1, we extend the line segment of A and B to a double in length to find the predicting point C for the person, where the distance D_2 is set equal to D_1.

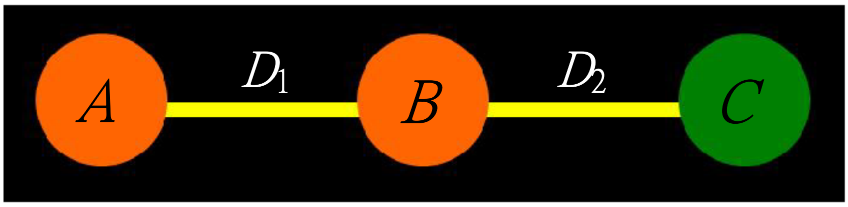


Figure 5.3 Prediction of the position of a person.

But in this study, we need to track a person in the images which are taken by *fisheye* cameras. Because there is a high degree of distortion in each of such omni-images, the speed of movement of a person in the image will also change even if the person walks at a fixed speed. Specifically, if a person moves at the center of an image, his/her moving speed will be faster than that of a person who moves at the edges of the same image. The reason is that the partial image which neighbors the center of an image is enlarged, and the partial image which neighbors the edges of an image is shrunk, as shown in Figure 5.4 where the sizes of two red squares are the same in the real world, but different in the image.

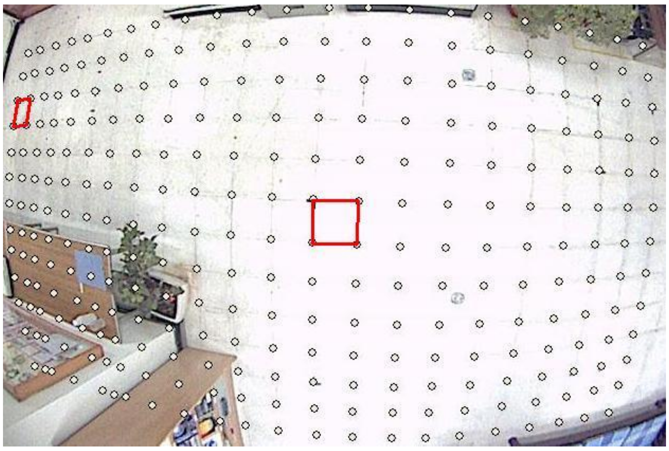


Figure 5.4 An example of distortion in the image taken by a fisheye camera.

Because of the problem mentioned above, three situations are identified in this study which should be dealt with for our purpose of smooth tracking of a person under the surveillance of a fish-eye camera. Situation 1 occurs when a person walks from the edge to the center in an image. In such a situation, the distance of prediction should be enlarged, meaning that D_2 should be longer than D_1 in Figure 5.3. On the other hand, Situation 2 occurs if a person walks from the center to the edge in an image. Here, the distance of prediction should be shrunk, meaning that D_2 should be shorter than D_1. And Situation 3 occurs when the person walks around the center of the image. Here, D_1 should be set equal to D_2.

As a summary, the formula for calculating D_2 can be written as follows:

$$D_2 = D_1 + \rho(|\overline{OA}| - |\overline{OB}|) \tag{5.1}$$

where the symbol O represents the central point of the image, the symbol $|\overline{OA}|$ represents the distance between point A and O, and $|\overline{OB}|$ is interpreted similarly. So

$\left|\overline{OA}\right| - \left|\overline{OB}\right|$ means the distance of approaching to the center O of an image when a person walks from point A to B. The detail of using Equation 5.1 is described below for all of the three situations.

The symbol ρ in Equation 5.1 is a ratio of enlarging and shrinking, whose definition will be different in Situations 1 and 2. The equations for calculating ρ are derived in this study to be:

$$\rho = \frac{p-q}{q} \quad \text{for Situation 1;} \tag{5.2}$$

$$\rho = \frac{p-q}{p} \quad \text{for Situation 2,} \tag{5.3}$$

where p: q is the average ratio of the distance between every two consecutive image points listed in the mapping table as shown in Figure 5.5, where we assume p to be closer to the center of the image than q.

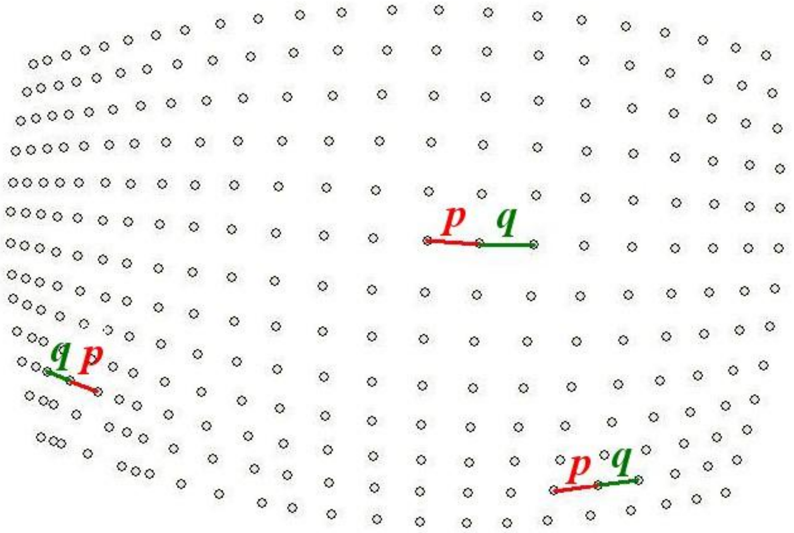


Figure 5.5 Calculating p and q.

In more detail, in Situation 1 because a person walks from the edge to the center in an image, $\left|\overline{OA}\right| - \left|\overline{OB}\right|$ will be larger than 0, and the distance D_2 will be enlarged, as shown in Figure 5.6. For example, if a person moves 30 cm totally, and approaches to the center of the image for a distance 20 cm and p: q is 5: 4, then D_1 is 30cm, and $|\overline{OA}|-|\overline{OB}|=20\,\text{cm}$. The prediction of D_2 will be $30+\frac{5-4}{4}\times 20=35$ cm according to Formula (5.2).

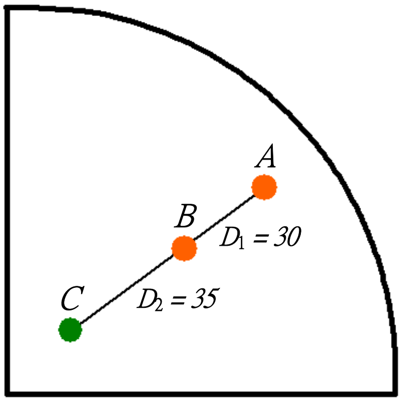


Figure 5.6 Situation 1.

In Situation 2, the person walks from the center to the edge in an image, so $|\overline{OA}|-|\overline{OB}|$ will be smaller than 0, and the distance D_2 will be shrunk, as shown in Figure 5.7. For example, if the values of the variables are the same as the above except that the person is leaving from the center of the image for a distance of 20 cm, then the prediction of D_2 will be $30-\frac{5-4}{5}\times 20=26$ cm according to Formula (5.3).

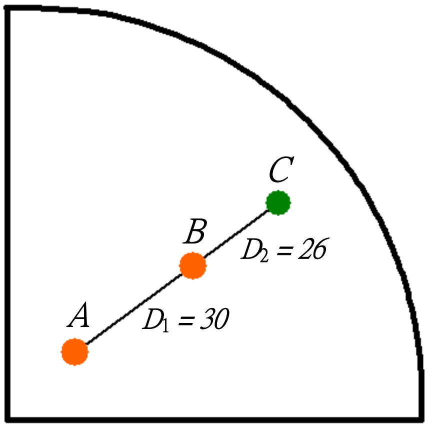


Figure 5.7 Situation 2.

In Situation 3, because the value of $|\overline{OA}|-|\overline{OB}|$ is zero, D_1 will be set equal to D_2 as shown in Figure 5.8.

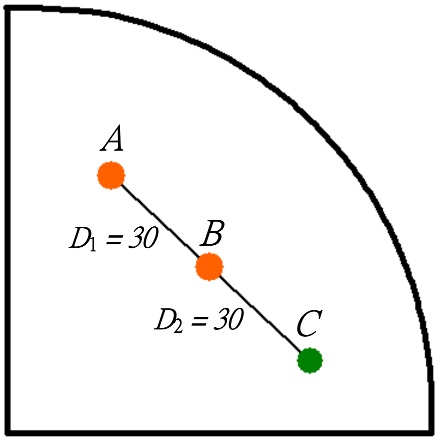


Figure 5.8 Situation 3.

5.4 Using Multi-Cameras to Expand Range of Surveillance

Because the FOV's of a camera is finite, we use multi-cameras in this study to expand the range of surveillance. A person may move its position between these cameras, so we have to decide which camera should be used to take images of the person. A simpler solution to this problem is described in Section 4.4.3, but the target is a person now instead of a vehicle, so we cannot use the values of the odometer anymore.

Besides, because we only calculate a partial image to find the position of a person, if a person walks to the FOV's of another camera, we should also decide which region in the image taken by camera 2 should be calculated to continue finding the position of the person. The proposed method for solving the problem here is described in Algorithm 5.2.

Algorithm 5.2: Calculating the image coordinates of the position of a person in the image taken by a second camera.

Input: The mapping tables of two cameras C_1 and C_2, and the image coordinates I_1 of the position of a person in the FOV's of camera C_1.

Output: The image coordinates I_2 of the position of a person in the image taken by C_2.

Steps:

Step 1. Calculate the coordinates of the four corner points A, B, C and D of the overlapping area of cameras C_1 and C_2, and the equation of line L, as shown in Figure 5.9.

Step 2. Check the position of the person to see if he/she moves from the FOV's of one camera to another and exceeds the line L continuously. If yes, continue; otherwise, use the original camera C_1 to take images.

Step 3. Convert I_1 to the global coordinates G by the mapping table of C_1.

Step 4. Convert G to the image coordinates I_2 by the mapping table of C_2.

Step 5. Use C_2 to continue to take images of the person, and calculate the position of the person by the mapping table of C_2.

In Algorithm 5.2, we calculate the coordinates of points A, B, C and D, and the equation of line L as shown in Figure 5.9 by the method described in Section 4.4.3 first. When a person walks from the FOV's of one camera to another and exceeds the line L, for example from P_1 to P_2, then the coordinates of the person P_{ics} could be calculated as described in Section 5.2 in the ICS (Image Coordinate System), and then converted to the GCS (global coordinate system) P_{gcs}. Afterward, P_{gcs} is converted to the image coordinate system again but in another camera system.

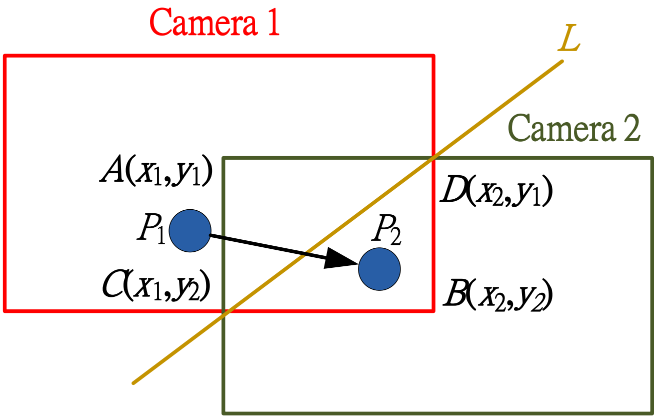


Figure 5.9 An example of hand-off when a person moves from P_1 to P_2.

For example, if a person walks from P_1 to P_2, that is, from a region within the FOV of camera 1 to a region within that of camera 2, and exceeds the line L, the coordinates of P_2 in the ICS will be converted to the GCS in camera 1, and then be converted to the ICS again in camera 2, as described in the algorithm. Afterward, camera 2 is used to take the image of the person.

5.5 Other Applications

In this section, two other applications of tracking a person are described. The first is that we can record the trail of a person, and the second is that we can calculate the walking speed of a person. Both the trail and the walking speed can be stored in a computer. And if something misses in a room, these data can be read out to check to see who has come into this room and where he/she has walked through. Besides, the walking speed is a characteristic of the person, and it can help us to decide the person's sex, age, and finally identity.

5.5.1 Recording Trail of a Person

When a person breaks into the environment under surveillance, he/she will be traced by the omni-cameras continuously as described previously. The positions of the person will be calculated in every 400 ms, and the coordinates of these positions can be reserved in a file, as shown in Table 5.1.

Table 5.1 The record of a person’s trail.

(379,213)	(509,211)	(649,237)
(430,199)	(521,220)	(657,233)
(427,196)	(543,225)	(702,221)
(442,192)	(558,216)	(746,238)
(454,186)	(586,219)	(781,250)
(455,192)	(593,237)	(802,260)
(473,207)	(630,227)	(844,254)
(464,192)	(620,233)	(844,257)
(485,198)	(616,240)	(850,257)
(502,187)	(626,244)	(857,264)

If something is found missing in the environment later, the video taken by the camera on the vehicle and the file recording the person’s trail can be inspected. We can then judge who broke into the room by the video, and draw the trail of the person on the map as shown in Figure 5.10 to check where the person has walked through, and hence judge if he/she has taken the stuff or not.

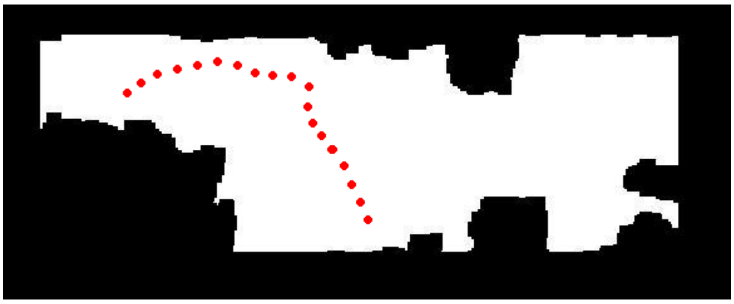


Figure 5.10 The trail of the intruding person drawn on the map.

5.5.2 Calculating Walking Speed of A Person

The purpose of calculating the walking speed of a person is that if the person who breaks into the environment under surveillance has a mask on his/her face, even if we have taken clear images of him/her by the camera equipped on the vehicle, we still cannot tell the identity of him/her. However, we can gather other characteristics of the person, like the walking speed which is one of the characteristics that can help us to identify the person, as mentioned previously.

Because the position of the person is calculated in every 400 ms, we can calculate the distance of movement of the person in the same frequency. Hence, we can take every two consecutive positions P_1 and P_2 of the person, and calculate the walking speed of the person by the following equation:

$$\frac{|P_1 - P_2|}{0.4} \times \frac{60}{100} \text{ meters/minutes} \qquad (5.4)$$

where $|P_1 - P_2|$ means the distance between positions P_1 and P_2 in cm. We can calculate the average walking speed by correcting a set S of all the walking speeds in a certain duration of time, removing data with zero values from *S*, and calculating an average speed value from *S*. The necessity of removing zero-valued data is: if the walking speed *w* is zero, it means that the person is not moving now, and so we cannot consider *w* into the average walking speed.

Chapter 6

Experimental Results and Discussions

In this chapter, we show some experimental results of the proposed security patrolling system. The first is the results of calculating the positions W_{real} of real-world points when a fisheye camera is affixed at the different heights. We compare the values of W_{real} which are calculated by the method proposed in Chapter 3 with those obtained by measuring manually.

The second is the results of calculating the positions of a person in an actual environment in the Computer Vision Laboratory, Department of Computer Science, National Chiao Tung University, and the computing results are compared with the real positions of the person.

The third is the results of the distance of deviating from the original path when an autonomous vehicle patrols in an actual environment, and the position and direction of the vehicle are corrected by the method proposed in Chapter 4. The detail will be described in Section 6.3.

6.1 Experimental Results of Calculating Positions of Real-world Points

In this experiment, a fisheye camera is affixed at several different heights. We calculated the values of the real-world point positions W_{real} for every height by the method mentioned in Chapter 3 and by manually measuring simultaneously.

We construct a basic mapping table for the fisheye camera first. We affix the camera at the height of 20 cm from the calibration board as shown in Figure 6.1. The real-world width between every two consecutive intersection points on the board we used is 1cm obtained by manually measuring. Then we can calculate the image coordinates of the intersections of the lines in the image and construct a basic mapping table by the method proposed in Chapter 3.

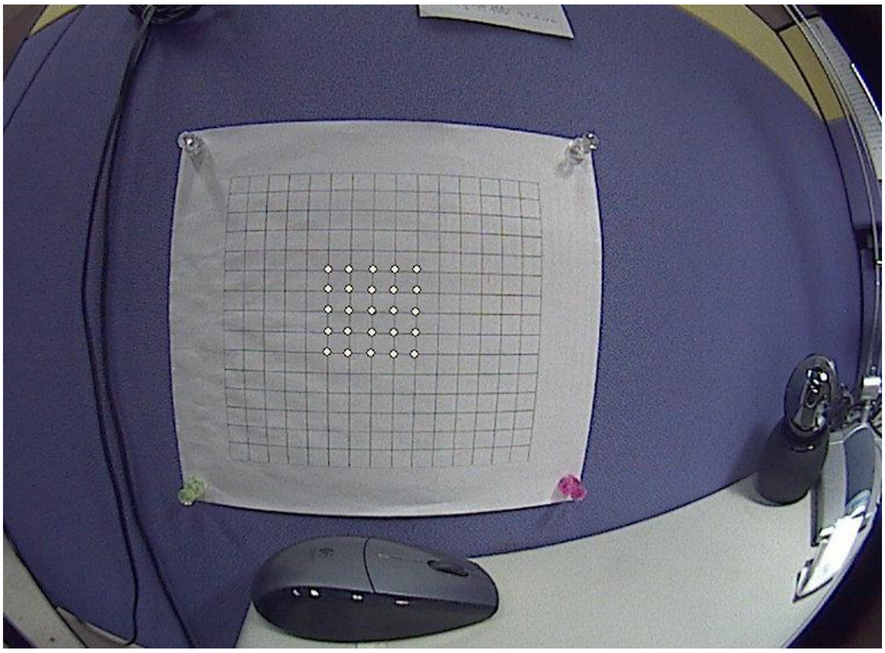


Figure 6.1 The camera is affixed at 20 cm from the calibration board.

After the table is constructed, we affix the camera to the heights of 10 cm, 15 cm, 30 cm, and 40 cm from the calibration board as shown in Figures 6.2, 6.3, 6.4, and 6.5, respectively. We can calculate the values W_{real} of real-world points for every height by the equations derived in Chapter 3 and by manually measuring simultaneously. The results of calculation are shown in Table 6.1, and the average error rate is 2.52%.

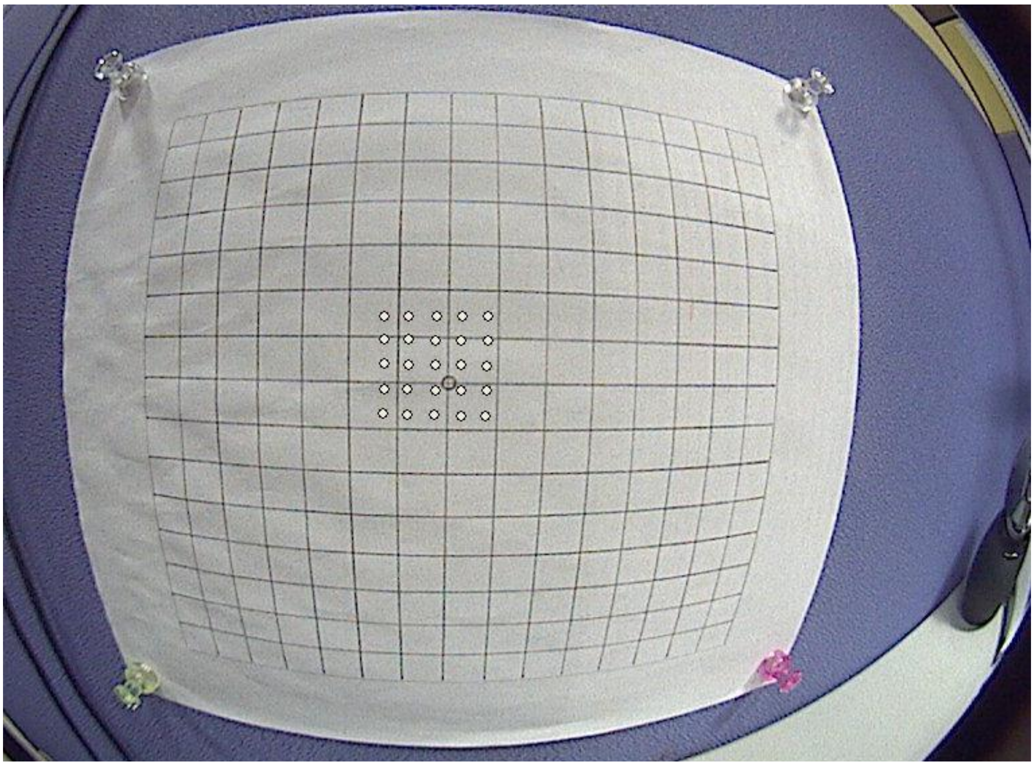


Figure 6.2 The camera is affixed at 10 cm from the calibration board.

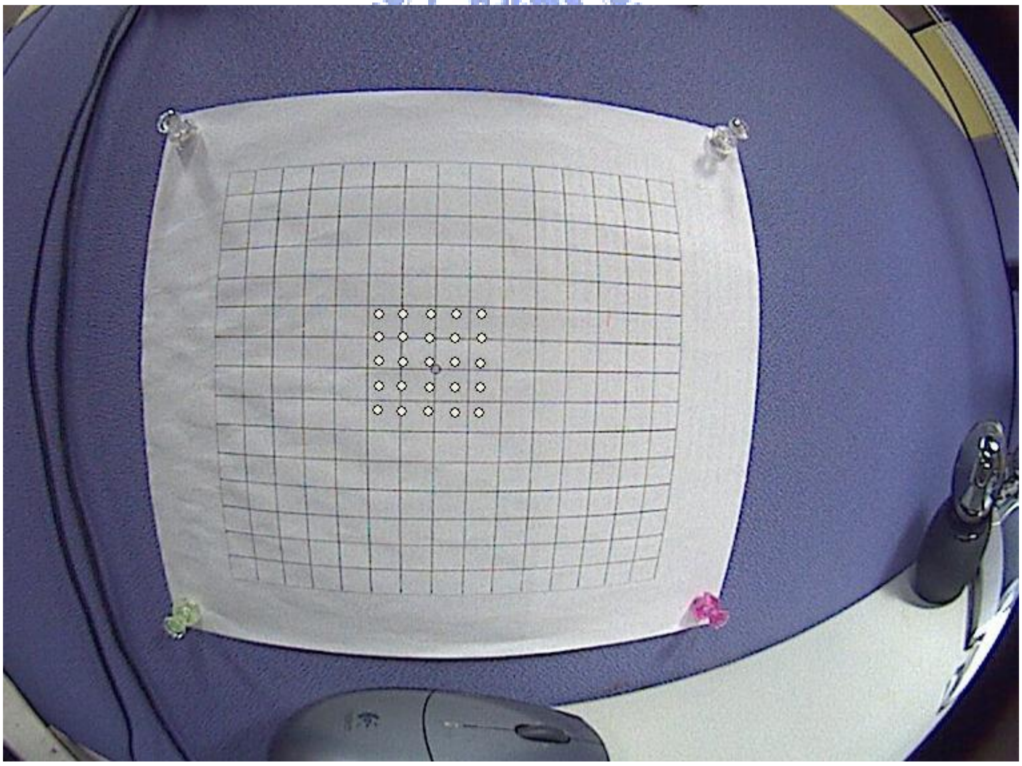


Figure 6.3 The camera is affixed at 15 cm from the calibration board.

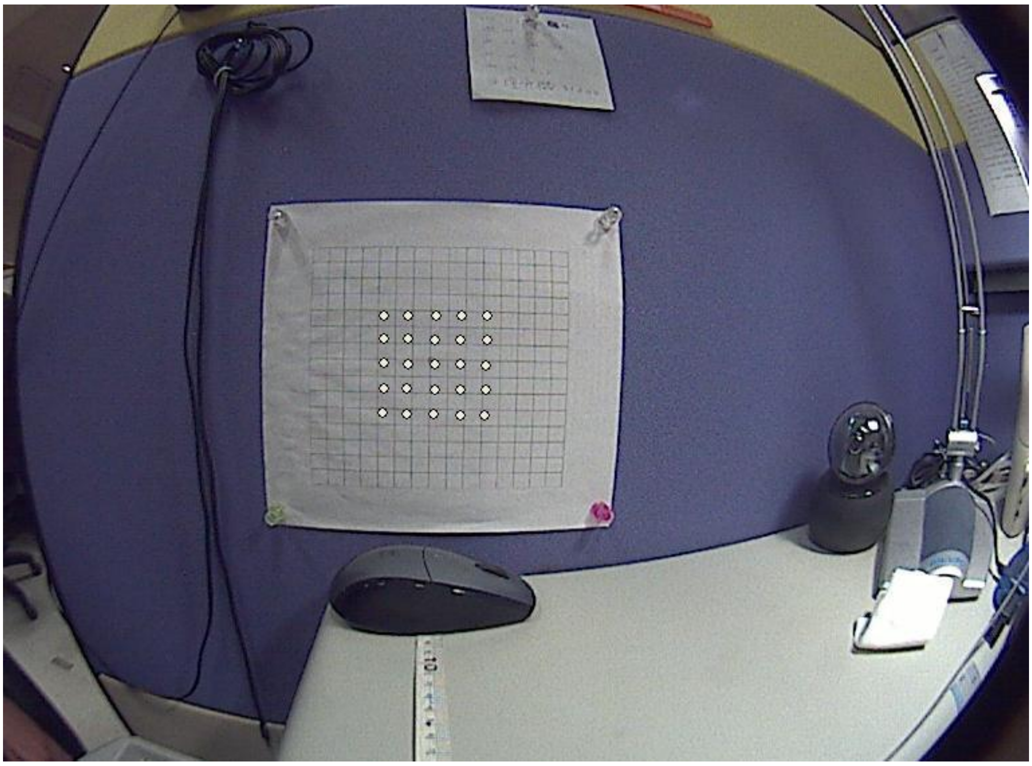


Figure 6.4 The camera is affixed at 30 cm from the calibration board.

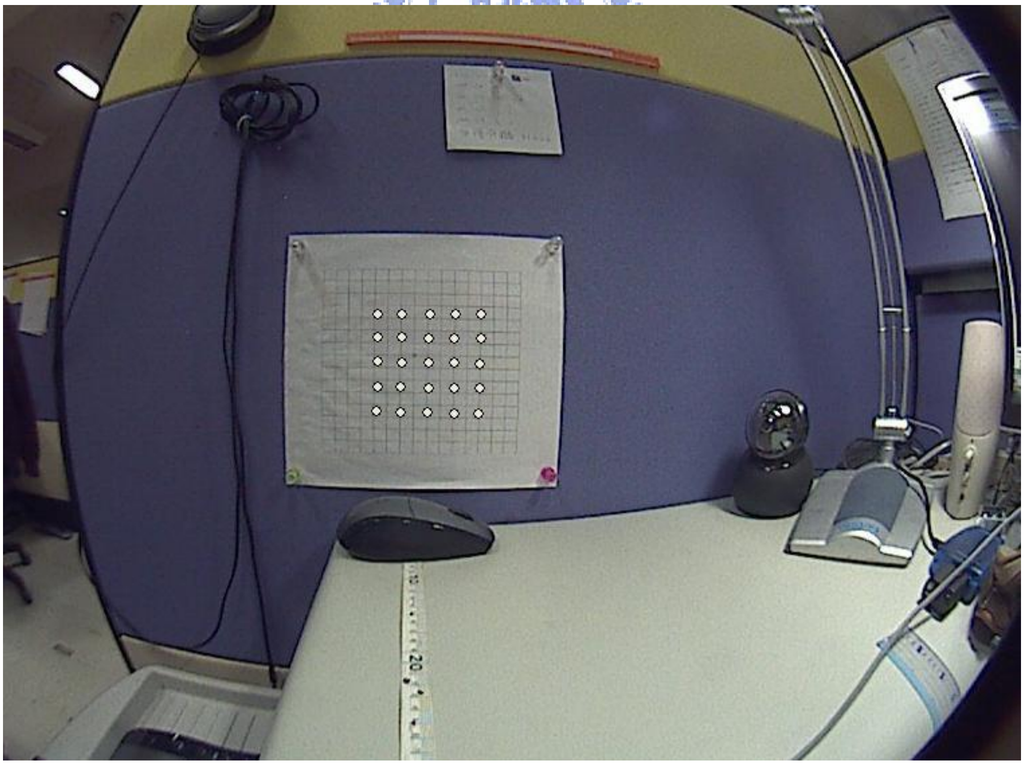


Figure 6.5 The camera is affixed at 40 cm from the calibration board.

Table 6.1 The results of calculating the value of W_{real} by two ways.

No.	Heights	(1)Calculated W_{real}	(2)Measured W_{real}	Error ($\frac{\|(1)-(2)\|}{(1)}$ %)
1	5	0.26	0.25	3.85
2	10	0.49	0.5	2.04
3	15	0.73	0.75	2.74
4	20	1	1	N/A
5	25	1.24	1.25	0.81
6	30	1.45	1.5	3.45
7	35	1.77	1.75	1.13
8	40	1.93	2	3.63

6.2 Experimental Results of Calculating Positions of a Person

The environment for this experiment is an open space area in our laboratory. Because of the property of imaging projection, after the region of a person is found in the image, the point which is in the region and is closest to the center of the image is the position of the person, as shown in Figure 6.6. We calculated several positions of a person by the method described in Chapter 5 and by manually measuring simultaneously. The positions are equally and randomly scattered in the region which is under surveillance in the laboratory as the red points shown in Figure 6.7, and the two images are taken by the two fisheye cameras affixed on the ceiling.

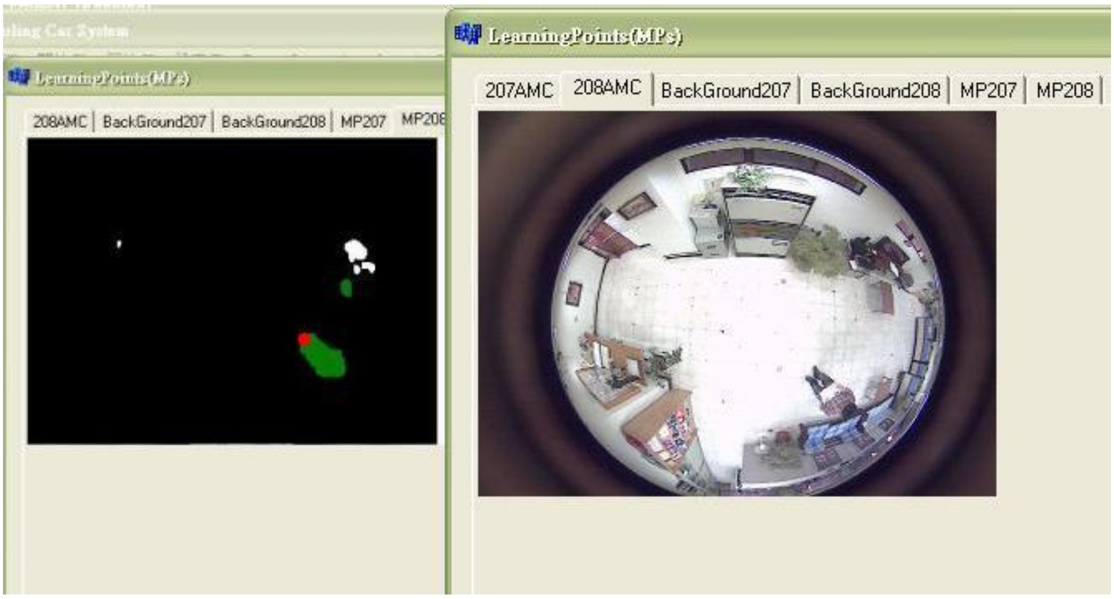
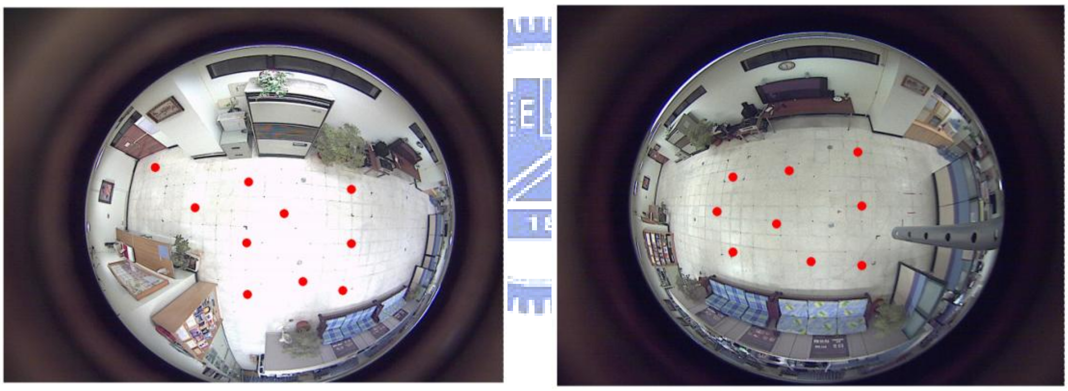


Figure 6.6 Finding the position of a person in the image.



(a) (b)

Figure 6.7 The experimental positions of the person.

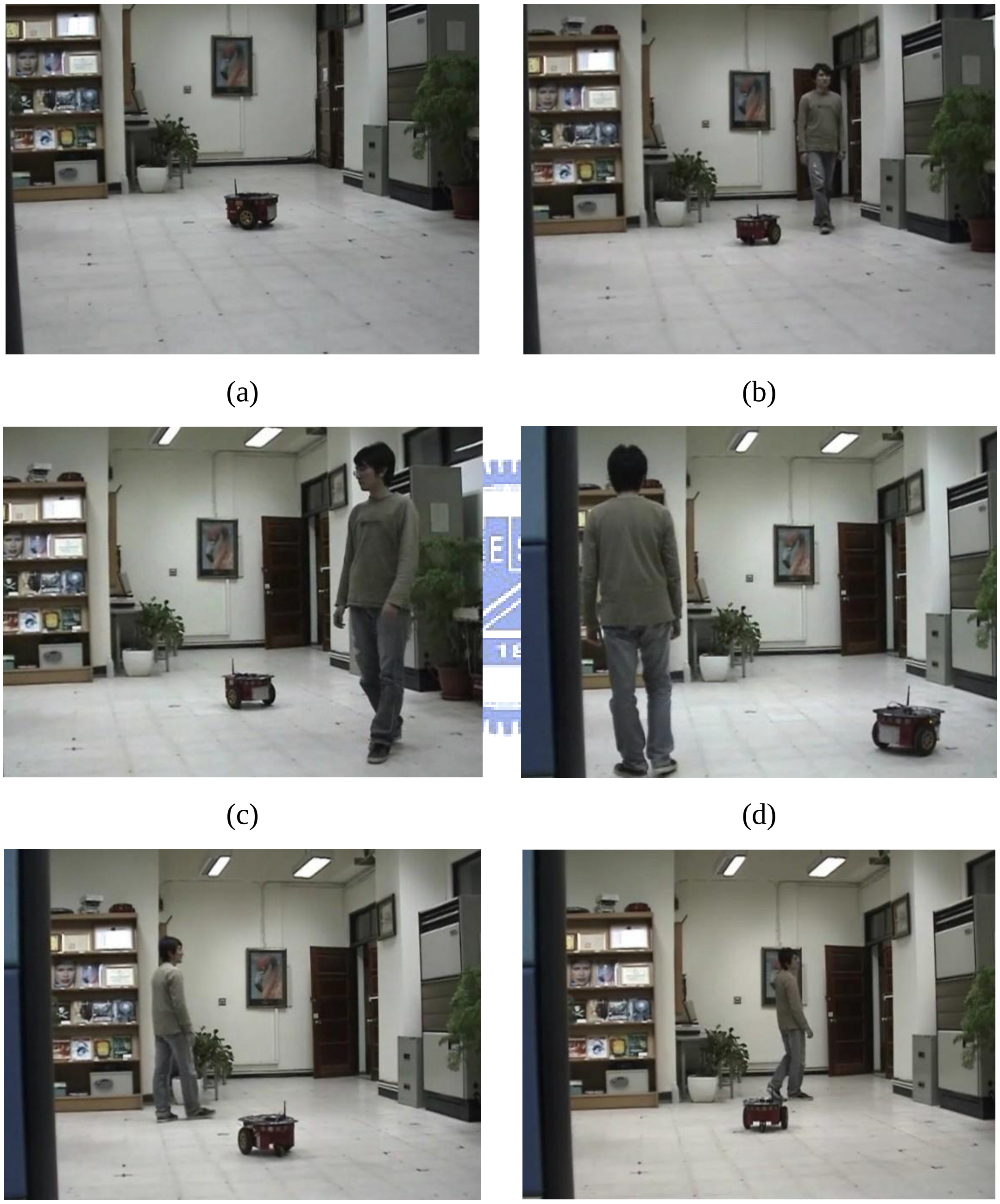
In Table 6.2, we show the global coordinates of the positions of the person which were obtained both by measuring manually and by calculation from the images, and the error rates of the positions so obtained.

The average error rate of finding the position of a person is 1.26% which is small enough for the vehicle to follow the person successfully in real applications.

Table 6.2 Calculating errors of the position of a person.

No.	(1)Actual Position		(2)Calculated Position		Error ($\frac{\|(1)-(2)\|}{(1)}$%)
	x	*y*	*x*	*y*	
1	457.5	213.5	458	198	2.971
2	47.6	7.4	45.9	7.4	0.035
3	163.4	32.2	160.7	34.2	0.020
4	457.5	305	439	289	4.364
5	244	30.5	238.19	30.5	0.024
6	457.5	396.5	442	387	2.973
7	272.3	35.2	269.3	41.1	0.024
8	610	213.5	596	204	2.475
9	382.2	29.8	378.8	40.9	0.030
10	610	305	595	291	2.932
11	399	85.3	392.8	97.3	0.033
12	610	396.5	591	383	3.161
13	915	213.5	906	223	1.383
14	304.5	115.5	299.2	124.2	0.031
15	61	30.5	52.89	28.56	0.122
16	305	30.5	303.48	30.58	0.005
17	915	305	894	290	2.592
18	122	30.5	113.62	28.68	0.068
19	915	396.5	901	383	1.905
20	183	30.5	174.53	30.5	0.046

A real example of following a specific person in our laboratory is shown below. In Figure 6.8(a), the vehicle patrolled in the laboratory. In Figure 6.8(b), a person broke into the laboratory, and the vehicle stopped the patrolling task and started to follow the person. In Figures 6.8(c) through 6.8 (f), the vehicle followed the person continuously until the person left the environment.



(a)

(b)

(c)

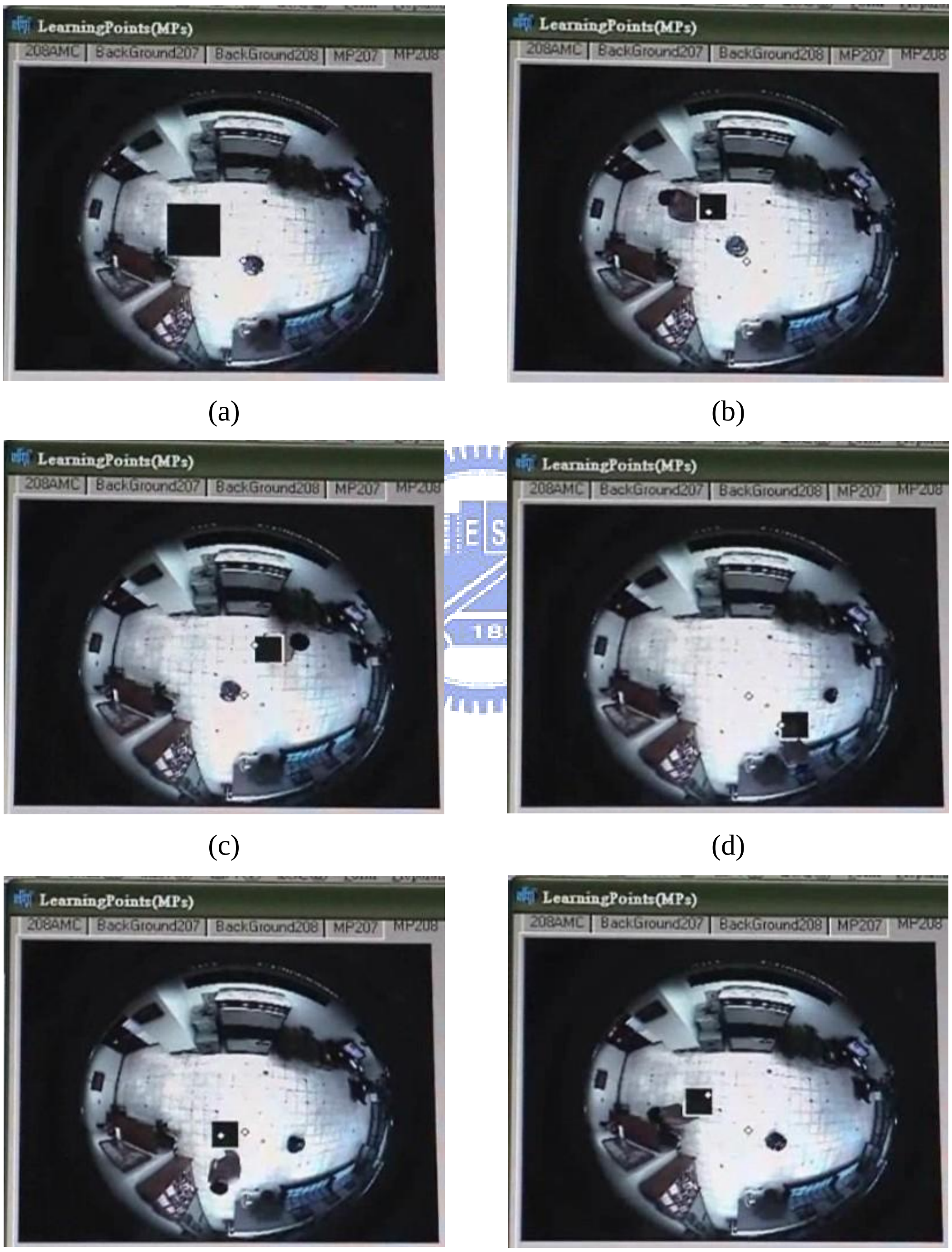
(d)

(e)

(f)

Figure 6.8 A real example of following the specific person. (a) The vehicle patrolled in the laboratory. (b) through (f) The vehicle followed the person continuously.

The images taken by the fisheye cameras on the ceiling is shown in Figure 6.9. The black squares are the regions we processed, and the white circles in the black squares are the positions of the person we calculated.



(a) (b)

(c) (d)

(e) (f)

Figure 6.9 The images taken by the cameras on the ceiling, and the positions of the person are indicated.

6.3 Experimental Results of Distance of Deviations from Navigation Path

Because the autonomous vehicle used in this study suffers from accumulation of mechanical errors, two top-view omni-cameras are utilized to locate and monitor the vehicles. In this experiment, the vehicle patrols in our laboratory continuously and the monitored points are selected by a user in the environment map, as shown in Figure 6.10.

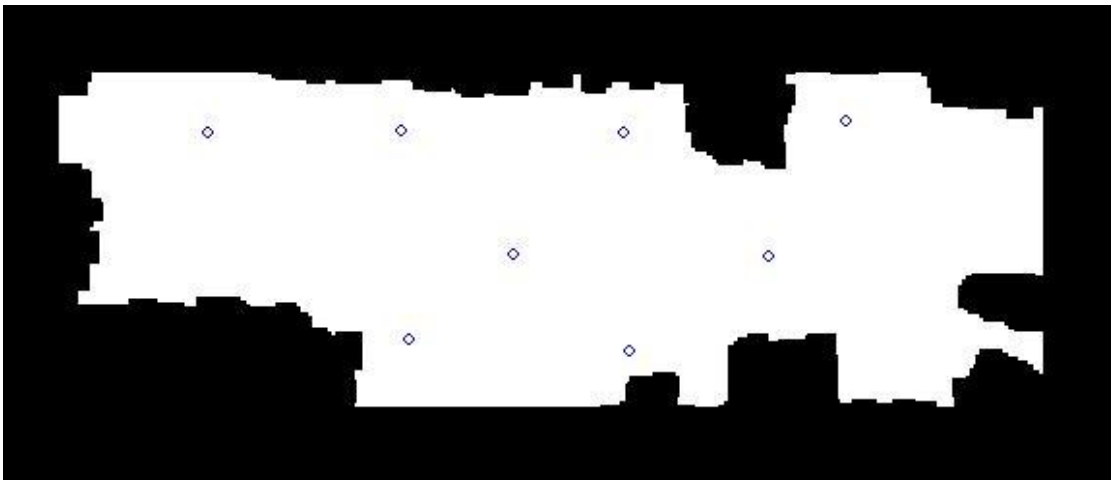


Figure 6.10 The monitored points selected by a user.

We record the length of every segment of the path in cm, and the total length of the navigation of the vehicle in meter in Table 6.3. In our experiments, when the vehicle navigated to the destinations of every segment of the path, the distance between the real position of the vehicle and the original destination of every segment of the path was calculated and recorded in the table, too. The error rates obtained by dividing the distance of the error by the length of the segment are also listed in the table.

We also performed a double-check experiment. In the first time, the vehicle position and direction were not corrected by any method, and the results are recorded in Table 6.3. In the second time, the data were corrected by the method described in Chapter 4, and the results are recorded in Table 6.4.

Table 6.3 Records of the uncorrected mechanic errors in every segment of path

No.	(1) Length of Segment	Length of Total Segments	(2) Distance of Error	Percentage of Error $(\frac{(2)}{(1)}\%)$
1	168	1.68	9	5.36
2	288	4.56	11	3.82
3	93	7.41	23	24.73
4	190	9.31	18	9.47
5	162	10.93	16	9.87
6	62	13.15	32	51.61
7	155	14.7	68	43.87
8	125	15.95	57	45.6
9	367	19.61	24	6.54
10	306	26.73	81	26.47
11	188	31.49	12	6.38
12	192	33.41	38	19.79
13	92	36.24	63	68.47
14	162	37.86	63	38.88
15	160	39.46	101	63.12
16	57	41.58	99	173.68

Table 6.4 Records of the corrected mechanic errors in every segment of path.

No.	(1)Length of Segment	Length of Total Segments	(2)Distance of Error	Percentage of Error ($\frac{(2)}{(1)}$%)
1	131	1.31	9	6.87
2	283	4.14	8	2.82
3	97	7.05	5	5.15
4	195	9.00	12	6.15
5	158	10.58	7	4.43
6	73	12.96	5	6.84
7	160	14.56	8	5.00
8	132	15.88	13	9.84
9	380	19.68	31	8.15
10	317	27.06	16	5.04
11	185	31.99	5	2.70
12	189	33.88	7	3.70
13	91	36.73	7	7.69
14	141	38.34	7	4.34
15	145	39.99	5	3.03
16	53	42.04	5	9.43
17	128	43.32	15	11.71
18	381	47.13	27	7.08
19	316	54.48	10	3.16
20	191	59.46	6	3.14

As seen from the two tables, the average error rate of deviation is 37.35% without any correction, and is 5.81% with the correction of positions and directions using the proposed technique. The average error rate with correction is much smaller than those without any correction. Besides, by the tables above we also can know that the mechanic errors of every segment of the path will not accumulate anymore, because we have eliminated the errors caused in every segment of the path using our method.

6.4 Discussions

The proposed system utilizes the vision-based autonomous vehicle to perform the security patrolling task. For this purpose, some monitored points were utilized to guide the vehicle. By the way, there are more applications of monitoring specific spots in indoor environments, such as providing various services at various application environments. Every monitored point can be regarded, for example, as a *business service point* in which there are some customers. If the environment is a restaurant, the apparatus of showing menus can be equipped on a vehicle, and then the vehicle can move to each service point along assigned optimal paths to ask what dishes or services are needed. If the environment is a company, the vehicles also can be utilized to deliver documents or messages in each service point.

However, there are still some problems in the proposed system. If an object appears next to the vehicle suddenly, the top-view omni-cameras will not have the ability to identify the vehicle. Furthermore, when tracking a person in the environment, if another person walks close to the person who is tracked, we may not be able to calculate the right position of the person. To solve the problem, it might be necessary to add information of color and sample models of the vehicle and the person to this system. The problems are worth for future researches.

Chapter 7

Conclusions and Suggestions for Future Works

7.1 Conclusions

In this study, we utilize a vision-based autonomous vehicle and omni-cameras affixed on the ceiling to perform indoor security surveillance. The indoor environment can be complicated with static and dynamic obstacles. We have proposed several techniques and designed some algorithms for indoor security surveillance, which are summarized in the following.

(1) *A height-adaptive space mapping method* has been proposed, by which we can construct an adaptive mapping table for a camera irrespective of the height of the camera. By the mapping table, we can convert the coordinates of the points in the environment between the image coordinates and the global coordinates. Hence we can calculate the real-world position of a vehicle or a person.

(2) *An environment-information calculation method* has been proposed, by which we can obtain all the ground regions in the environment, which form the environment map of the patrolling environment. By the constructed map, we can know the positions of all the obstacles. When a vehicle navigates, the patrolling path can be checked to see if there is any obstacle on it. If so, we can plan another path to avoid the obstacles also by use of the environment map.

(3) *A fast obstacle avoidance method* has been proposed, by which a vehicle can

avoid static and dynamic obstacles in the environment. If there are some obstacles on the original path, several turning points are calculated by the method and form a new path for navigation. The new path is the shortest path with the least number of turning points.

(4) *A vehicle location and direction correction method* has been proposed. Because the vehicles suffer from mechanic errors, we utilize the top-view omni-cameras to locate them in this study. By the odometer values of the vehicles, we can calculate the centroids of the vehicles in the image. After the centroids are transformed into the global space, the odometer values are corrected by the coordinates of the resulting points. Besides, the directional angles of the vehicles also must be corrected, in which two continuous correct position points are utilized to do the job. By the correction method, the position and the direction of the vehicle can be corrected automatically. Accordingly, the vehicle will not deviate from the planned path too far.

(5) A method for handling the camera handoff problem has been proposed, by which a vehicle can navigate under several cameras to expend the range of surveillance. Besides, when tracking a person in the environment, because the person may move among the FOV's of the cameras, the method is also applied to calculate the position of the person in the images.

(6) A method for calculating the position of a person in omni-images based on the rotational invariance property has been proposed, by which we can calculate the position via the image taken by the fisheye camera on the ceiling. Besides, only the partial image has to be processed to calculate the exact position of the person to reduce the amount of calculation, decrease the probability of interference by other dynamic obstacles, and increase the preciseness of calculating the position of the target.

(7) *A position prediction method* for use in omni-images has been proposed. Because we only process the partial image to calculate the position of a person, the position of the person in the next segment of time should be predicted by the prediction method. Besides, because the fisheye cameras are highly distorted, the predicting position of the person is revised by the proposed method, too.

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

7.2 Suggestions for Future Works

The proposed strategies and methods, as mentioned previously, have been implemented on a vehicle system with multiple omni-cameras on the ceiling. According to this study, in the following we make several suggestions and point out some related interesting issues, which are worth further investigation in the future:

(1) using a pen-tilt-zoom camera equipped on the vehicle to extract features of images to detect whether monitored objects still exist;

(2) increase the ability to detect more dangerous conditions;

(3) increase the ability of warning users immediately through cell phones or electronic mails;

(4) increase the ability of voice control when users want to issue navigation orders to the vehicle;

(5) increase the ability of constructing the adaptive mapping table automatically for the cameras whose FOV’s is not vertical to the ground;

(6) increase the ability of tracking multiple people in the environment simultaneously;

(7) increase the ability of processing the fragmented images taken by the camera due to insufficient network bandwidths;

(8) control the vehicle by the information gathered in the image space to reduce the errors of converting the coordinates of the vehicle or the person between the global coordinates and the image coordinates;

(9) increase the ability of patrolling in a dark room via infrared-ray cameras;

(10) increase the ability of handling more complicated situations of the handoff problem by using more information such as camera positions.

References

[1] J. Kannala and S. Brandt, “A generic camera calibration method for fish-eye lenses,” *IEEE Proceedings of 17th international conference on pattern recognition*, Vol. 01, pp. 10-13, 2004.

[2] S. shah and J. K. Aggarwal, “A simple calibration procedure for fish-eye (high distortion) lens camera,” *IEEE Transactions on Robotics and Automation*, Vol. 4, pp. 3422-3427, May 1994.

[3] S. Shah and J. K. Aggarwal, “Intrinsic parameter calibration procedure for a fish-eye lens camera with distortion model and accuracy estimation,” *Pattern Recognition*, Vol. 29, No. 11, pp. 1775-1788, 1996.

[4] H. C. Chen and W. H. Tsai “A Study on optimal security patrolling by multiple vision-based autonomous vehicles with omni-monitoring,” *Proceedings of 2008 International Computer Symposium*, pp. 196-201, Taipei, Taiwan, Nov. 2008.

[5] I. Fukui, “TV image processing to determine the position of a robot vehicle,” *Pattern Recognition*, Vol. 14, pp. 101-109, 1981.

[6] Betke M and Gurvits L, “Mobile robot localization using landmarks,” *IEEE Transactions on Robotics and Automation*, Vol. 13, No. 2, pp. 251-263, Apr. 1997.

[7] M. J. Magee and J. K. Aggarwal, “Determining the position of a robot using a single calibration object,” *IEEE Conference on Robotics*, pp. 57-62, Atlanta, Georgia, USA, May 1983.

[8] J. Huang, C. Zhao, Y. Ohtake, H. Li, and Q. Zhao, “Robot position identification using specially designed landmarks,” *Proceedings of 2006 IEEE Conference on Instrumentation and Measurement Technology*, Sorrento, Italy, Apr. 2006.

[9] H. L. Chou and W. H. Tsai, “A new approach to robot location by house

corners," *Pattern Recognition*, Vol. 19, pp. 439-451, 1986.

[10] 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*, pp. 415-418, Washington, DC, USA, Dec. 2006.

[11] K. C. Chen and W. H. Tsai, "A study on autonomous vehicle navigation by 2D object image matching and 3D computer vision analysis for indoor security patrolling applications," *Proceedings of 2007 Conference on Computer Vision, Graphics and Image Processing*, Miaoli, Taiwan, June 2007.

[12] D. Cobzas, H. Zhang, and M. Jagersand, "Image-based localization with depth-enhanced image map," *Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2003)*, pp. 1570-1575, Taipei, Taiwan, 2003.

[13] P. Biber, H. Andreasson, T. Duckett, and A. Schilling, "3D modeling of indoor environments by a mobile robot with a laser scanner and panoramic camera," *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, Sendai, Japan, Sept. 2004.

[14] W. Lu and Y. P. Tan, "A color histogram based people tracking system," *IEEE Circuits and Systems*, Vol. 2, pp. 137-140, May 2001.

[15] T.Takeshita, T. Tomizawa, and A. Ohya, "A house cleaning robot system – path indication and position estimation using ceiling camera," *IEEE SICE-ICASE*, pp. 2653-2656, Busan, Korea, Oct. 2006.

[16] J. J. Yang, "Design and implementation of an image processing based watchdog robot," *MS Thesis*, National Cheng Kung University, Tainan, Taiwan, June 2002.

[17] S. Y. Tsai and W. H. Tsai, "A study on mobile robot navigation with simple learning procedures by ultrasonic sensing and computer vision techniques,"

Proceedings of 2008 International Computer Symposium, pp. 207-212, Taipei, Taiwan, Nov. 2008.