VISION-BASED SECURITY PATROLLING IN INDOOR ENVIRONMENTS USING AUTONOMOUS VEHICLES

Ming-Che Chen (陳明哲)[1] and Wen-Hsiang Tsai (蔡文祥)[1, 2]

[1]Department of Computer & Information Science
National Chiao Tung University, Hsinchu, Taiwan 300
[2]Department of Computer Science & Information Engineering
Asia University, Taichung, Taiwan 413
E-mail: {gis92624, whtsai}@cis.nctu.edu.tw

ABSTRACT

A vision-based approach to security patrolling in indoor environments using autonomous vehicles is proposed. A small vehicle with wireless control and image grabbing capabilities is used as a test bed. Three stages of security patrolling are proposed. First, a simple learning strategy is designed for flexible learning of reachable spots and monitored objects in indoor environments. Accordingly, a planned path is obtained, and monitored objects and doors are specified by analyzing user commands. Next, following the learned path, the vehicle can accomplish specified navigation sessions. Two methods, one for mechanic error correction modeling and the other for vehicle position modification by monitored object positions, are proposed for navigation accuracy maintenance. Finally, an object matching algorithm is used for checking the existence of monitored objects and the status of door opening. Experimental results show the flexibility and feasibility of the proposed approach.

1. INTRODUCTION

The goal of this study is to design an intelligent system for indoor security patrolling by means of a vehicle. It is desired to design the system to be capable of indoor navigation and security monitoring, including object existence verification and door situation recognition. To achieve this goal, learning of navigation paths and recording of the features of monitored objects are required before the vehicle can navigate automatically.

Li [10] proposed a learning method for dealing with complicated surroundings as well as techniques for autonomous vehicle navigation. The user controls the vehicle to patrol and analyze the captured image in the learning process, and a navigation map is created dynamically. The vehicle can keep away from obstacles when it navigates automatically. Chen [12] proposed two navigation modes and a fuzzy guidance technique. A navigation map is created by two kinds of learned data and the fuzzy technique is applied to achieve vehicle guidance with an obstacle avoidance capability.

It is usually difficult to segment objects from complicated background in images. Kass, WitKin, and Terzopoulos [1] proposed the *snake algorithm* to segment patterns in the image. However, there are some drawbacks in the algorithm. Some researches [2-6] tried to improve the algorithm by more complicated techniques. After detecting an object from the image, the features of the object is recorded in the learning process. In order to represent the shapes of objects, Sussman [7] mentioned the technique of fitting an ellipse to a set of data points by using the least square fitting technique. And Pilu, Fitzgibbon, and Fisher [8] proposed the ellipse representation of object pixels by using the edge pixels of the object.

Most vision-based methods for vehicle guidance use 3-D vision techniques. Instead, to accelerate the vehicle navigation speed, we adopt the 2-D vision approach. Accordingly, we adopt a technique for mapping vehicle location in an acquired image into the real vehicle location in the environment space. We call the technique location mapping calibration in this study. Furthermore, the vehicle we use for this study, like most vehicles, has precision errors in mechanic control of vehicle directions, and so we have developed a mechanic error correction method in this study to improve vehicle guidance precision. We also have designed a simple and flexible learning process in which no vision is used except when recording of the features monitored objects and doors is conducted. Instead, only the positions and directions of the vehicle at turning spots in a navigation path, which consists of line segments, are recorded and processed into a map of graphs with nodes representing turning points. We have also improved the snake algorithm for monitored object detection and designed a method for door opening statue checking. Finally, we have designed a method for correcting vehicle locations by the use of learned object locations.

The remainder of this thesis is organized as follows. In Section 2, the system configuration is described. In Section 3, the proposed methods for location mapping

calibration and mechanic error correction are described. The proposed learning process and object detection method are described in Section 4. The proposed vehicle navigation process, the strategies for navigation accuracy maintenance, object recognition, and judgment of the door situation are described in Section 5. Some experimental results are given in Section 6. Finally, some conclusions are included in Section 7.

2. SYSTEM CONFIGURATION

The vehicle used in this study as a test bed is shown in Fig. 1. There are two larger wheels and one auxiliary small wheel at the rear of the vehicle. A camera and an odometer are equipped on the vehicle together with an embedded hardware system. The odometer provides the location of the vehicle in each navigation session. The direction of the vehicle is also computed and provided by the vehicle system. The origin of space coordinate system is taken to be the start position of the vehicle.

There is a wireless device in the vehicle and another in the PC. Commands issued be the user from the PC may be transmitted to the wireless signal receiver on the vehicle through an access point in a wireless network system in the environment. Also, the image grabbed by the camera on the vehicle may be transmitted to the PC for monitoring by the user.

Fig. 1 The vehicle used in this study.

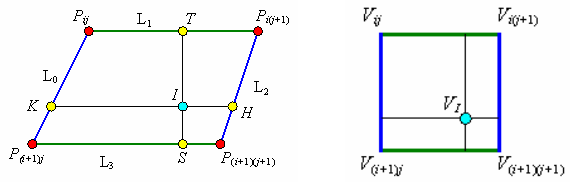
3. LOCATION MAPPING CALIBRATION AND MECHANIC ERROR CORRECTION

We process an image captured by the camera to obtain the relative position between an object and the vehicle. The method is 2-D in nature. It is described in Section 3.1. On the other hand, the odometer and the vehicle system provide the positions and the directions of the vehicle in the environment during each navigation session. Possible mechanic errors might cause the real positions of the vehicle to deviate from the values provided by the odometer. The proposed correction method to reduce such mechanic errors is described in Section 3.2.

3.1. Location mapping calibration

We use a point set $\boldsymbol{P} = \{P_{00}, P_{01}, \ldots, P_{mn}\}$ whose coordinates in the vehicle coordinate system (VCS) attached on the floor are known in advance, and their corresponding pixel set $\boldsymbol{V} = \{V_{00}, V_{01}, \ldots, V_{mn}\}$ appearing in the image, to compute the VCS coordinates of a set of pixels by using an interpolation method. The method is illustrated in Fig. 2.

More specifically, given a pixel I in the image, which falls in a region with four corner points P_{ij}, $P_{(i+1)j}$, $P_{i(j+1)}$, and $P_{(i+1)(j+1)}$ of $\boldsymbol{P}$ whose vehicle coordinates are known, we utilize the image distances between I and the four lines formed by P_{ij}, $P_{(i+1)j}$, $P_{i(j+1)}$, and $P_{(i+1)(j+1)}$ as weights to compute the relative vehicle coordinates of I in the interpolation. The details are ommited.



(a) (b)

Fig. 2 Illustration of interpolation. (a) A region contains point I in a image. (b) Mapping of the region onto a floor region in vehicle coordinate system (real space).

3.2. Mechanic error correction

There are two problems caused by mechanic precision errors. One is that the moving path is a curve in a line following session. The other is that the actual position at each navigation spot is different from the position and direction provided by the vehicle system.

As shown in Fig. 3, assume that the vehicle moves from a start position P_0 and arrives supposedly at another position P_c in a straight-path navigation session. But the position specified by the coordinates provided by the odometer is actually point P_1 instead of P_c because the vehicle has moved through a curve path due to certain mechanic error accumulation. Also the direction, which is supposed to be the original one set at the start position, has actually been rotated for an angle of θ.

Because the path of the vehicle is a curve, we compute a curve equation to represent the path. In more details, at first we measure the vertical deviation distances manually when the vehicle navigates forward for a number of equal distances. According to the deviation values, we compute a second-order equation $y = ax^2 + bx + c$ by a curve fitting technique, as shown in Fig. 4. Among the equation, x is the distance between the starting position P_0 and the position P_1, and y is the deviation distance from the expected path.

In Fig. 3, the vehicle starts moving from point $P_0(x_0, y_0)$ toward the goal point $P_1(x_1, y_1)$. The final position the vehicle arrives at is point $P_c(x_c, y_c)$. We compute the distance x_d as $x_d = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}$. Utilizing x_d and the curve equation $y = ax^2 + bx + c$, the deviation distance y_d can be computed to be $y_d = ax_d^2 + bx_d + c$ for use in the navigation.

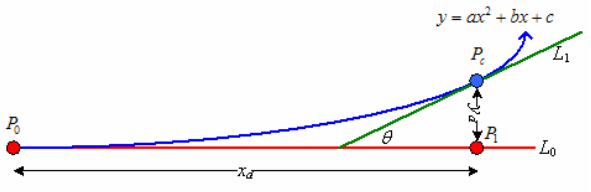


Fig. 3 Illustration of mechanic error correction.

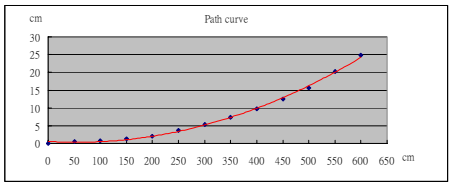


Fig. 4 A figure of the error correction curve.

Furthermore, to get the rotation angle θ shown in Fig. 3, we utilize the tangent of the curve $y' = 2ax + b$ to compute the direction of the curve at the location with deviation x_d as a vector $\vec{b} = [1, 2ax_d + b]$. Also, the direction of the x-axis may be specified as a unit vector $\vec{a} = [1, 0]$. So, we can get the angle θ at the deviation x_d by the formula $\theta = \cos^{-1}(\vec{a} \cdot \vec{b} / |\vec{a}||\vec{b}|)$. Then we can use coordinate transformation to get the coordinates of point P_c as follows:

$$x_c = x_d cos\theta - y_d sin\theta + x_0 ; \; y_c = x_d sin\theta + y_d cos\theta + y_0 .$$

4. LEARNING STRATEGIES FOR INDOOR NAVIGTAION

The purpose of learning in this study is to gather appropriate features of the environment for smooth and stable guidance of the vehicle during navigation. Two kinds of navigation data are used in this study. One is path data and the other is object data. An example of learning process is shown in Fig. 5.

An improved snake algorithm is proposed in the study to detect an object region in the image. After detecting an object region, three features of the object are computed as learned data for use when the vehicle monitors the objects during navigation.

4.1 Process of Learning Navigation Paths

A user gathers patrolling path data, by using the PC through wireless network, to control the vehicle to patrol in the environment. An example of path learning results is shown in Fig. 6.

The coordinates of each vehicle position recorded in the learning process are called "*node*". At the beginning, we record the first node as $(x_0, y_0) = (0, 0)$ into the set $\boldsymbol{N_{path}}$ and mark the node as N_0 with index 0 at the start position. Then, we record the node $N_i(x_i, y_i)$ into the set $\boldsymbol{N_{path}}$ by taking the values of the odometer (x_i, y_i) and mark the node P_i with the next index number, when the vehicle is at one of the following three situations:

(1) the position and the direction of the vehicle are corrected by the system according to the mechanic error model;
(2) the user controls the vehicle to turn;
(3) the user controls the vehicle to learn the data of certain objects.

We also record the finally node N_t into the set $\boldsymbol{N_{path}}$ and mark it as N_t by the next index number when the learning process is finished. We save finally all the nodes of the set $\boldsymbol{N_{path}}$ into the PC and finish the learning process.

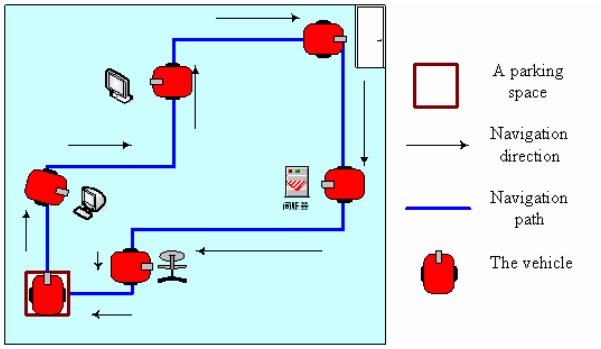


Fig. 5 An example of the learning process.

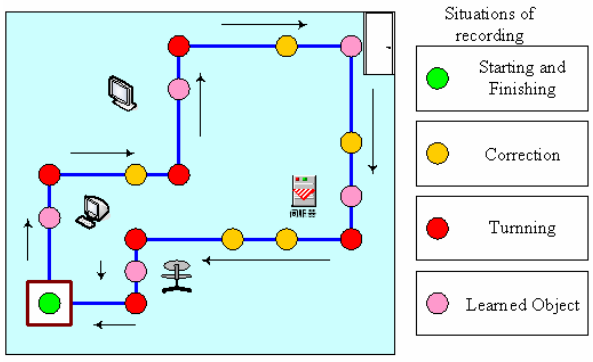


Fig. 6 Example of path learning with critical nodes recorded.

4.2 Process of Learning Monitored Objects

When the user controls the vehicle to move to the front of certain objects, the user must use the PC mouse to choose an object which appears in the image. As soon as the user chooses the object, the features of the object are computed automatically and saved. The entire learning process of objects is described as follows.

Algorithm 1: *object detection by improved snake algorithm.*

Input: an object image *I*.
Output: a feature set *O*.
Steps:
Step 1. Choose an object by "click and drag" as shown in Fig. 7.
Step 2. Compute the coordinates of the control points

according to the edge of the rectangle, as shown in Fig. 8(a).

Step 3. Compute the center point P_0 of the control points and the moving path of each control point.

Step 4. Compute the internal energy $E_{internal}$ and the external energy $E_{external}$ of every control point in the following way.

$$E_{internal} = \alpha \cdot \left| \frac{d\mathbf{v}_i(s)}{ds} \right|^2 + \beta \cdot \left| \frac{d^2\mathbf{v}_i(s)}{ds^2} \right|^2 + \gamma \cdot \left| \vec{V} \right|^2 ;$$

$E_{external} = \lambda \times$ [–(value of *Sobel operator* of the control point in the image)^2].

where $\mathbf{v}(s) = (x(s), y(s))$ specifies the image coordinates.

Step 5. Compute the original snake energy E_{snake} by summing up all the values of $E_{internal}$ and $E_{external}$ in the following way:

$$E_{snake} = \sum_{i=1}^{n} E_{internal}^{i} + \sum_{i=1}^{n} E_{external}^{i} .$$

Step 6. For each control point, do the following steps.

6.1 Compute the next position of the control point on the shrinking path and its internal energy $E_{internal}$ and external energy $E_{external}$ and the new snake energy $newE_{snake}$ by summing up $E_{internal}$ and $E_{external}$.

6.2 Compare the original snake energy E_{snake} with $newE_{snake}$.

6.3 If the original energy E_{snake} is smaller than $newE_{snake}$, stop moving the control point to the next position; else move the control point to the next position and substitute E_{snake} with the new snake energy $newE_{snake}$.

6.4 Compute the distances D_1 and D_2 between the control point and its two adjacent points.

6.5 Sum up D_1 and D_2. If their sum is larger than a threshold D_t, then adjust the coordinates of the control point such that the distances between the control and the two adjacent points are smaller. Also, compute the snake energy.

Step 7. When reaching the minimum energy, stop moving the control points, as shown in Fig. 8(b).

Step 8. Compute the object edges using the coordinates of the control points.

Step 9. Detect the object region using the coordinates of the object edge points.

Step 10. Compute three sets of feature data of the object:

A. means and standard deviations of the R, G, and B values of the pixel colors of the object.

B. the lengths of the long and short axes of the ellipse fitting the pixels of the object region by the following equations.

$$a = \sqrt{\frac{(\sum_{i=1}^{n} x_i^2 y_i^2)^2 - (\sum_{i=0}^{n} x_i^4)(\sum_{i=0}^{n} y_i^4)}{(\sum_{i=0}^{n} x_i^2 y_i^2)(\sum_{i=0}^{n} y_i^2) - (\sum_{i=0}^{n} x_i^2)(\sum_{i=0}^{n} y_i^4)}} ;$$

$$b = \sqrt{\frac{(\sum_{i=0}^{n} x_i^2 y_i^2)^2 - (\sum_{i=0}^{n} x_i^4)(\sum_{i=0}^{n} y_i^4)}{(\sum_{i=0}^{n} x_i^2 y_i^2)(\sum_{i=0}^{n} x_i^2) - (\sum_{i=0}^{n} x_i^4)(\sum_{i=0}^{n} y_i^2)}} .$$

where x_i and y_i are coordinates of the object pixels in the image and n is number of pixels.

C. the global coordinates of the object position bt the following equations.

$$Gx_{obj} = Vx_{obj}\cos\theta - Vy_{obj}\sin\theta + Gx_{car} ;$$

$$Gy_{obj} = Vx_{obj}\sin\theta + Vy_{obj}\cos\theta + Gy_{car}$$

where Vx_{obj} and Vy_{obj} are the relative coordinates between the object and the vehicle; Gx_{car} and Gy_{car} are the coordinates of the vehicle in the global coordinates; and θ is the direction angle of the vehicle in the global coordinates system.

An experimental result is shown in Fig. 7(b).

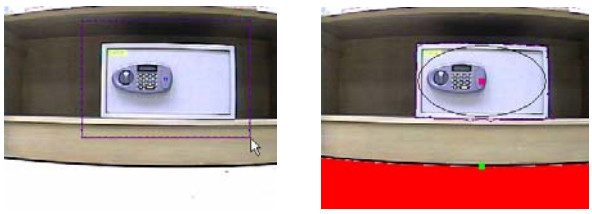
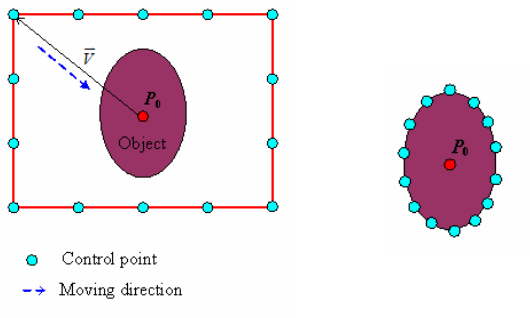


Fig. 7 The process of object detection. (a) Choosing an object in the image. (b) Result of computing feature data.



(a) (b)

Fig. 8 Process of executing the improved snake algorithm. (a) Locating control points and computing moving paths. (b) Experimental result of getting the minimum snake energy.

4.3 Process of Learning Monitored Doors

In the process of learning monitored doors, we point out a door on the image I by using a cursor, as shown in Fig. 9. We then use a region growing technique to find out the door region and save its means

of the R, G, and B values in the image and coordinates in the global coordinate.

Fig. 9 A learning process of choosing a door manually. (a) Choosing a door. (b) A door region is obtained.

4.4 Process of Automatic Path Map Creation from Learned Data

After ending the learning process, the path data, the object data, and the door data have all been saved. We then use the index numbers of all nodes and the objects linked by some nodes to create a path map for use in later navigation sessions, as shown in Fig. 10. By using the index numbers of the nodes, the vehicle can move along the navigation path in order.

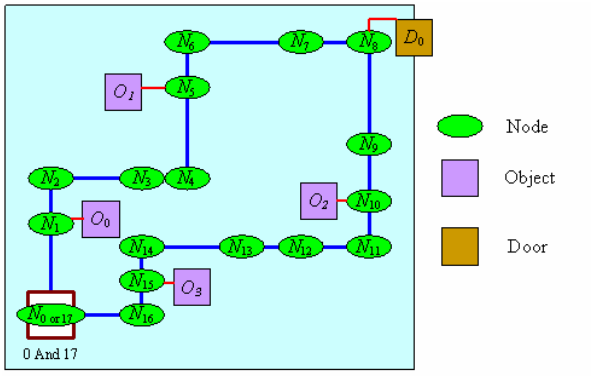


Fig. 10 An example of navigation map.

5. SECURITY PATROLLING IN INDOOR ENVIRONMENTS

In the security patrolling process, the vehicle navigates along a generated path by visiting the path nodes consecutively through the routes specified by the node edges and checks the existence of the learned objects. A security patrolling process is described in Fig. 11. The detailed process of navigation for security patrolling is described in the following.

5.1. Navigation Strategy for Straight-Line Sections

The navigation strategy of line following is adopted to reduce deviations from the route when the vehicle passes a node in its navigation path. The line following process will ensure that the vehicle passes each node during the process of visiting two adjacent nodes.

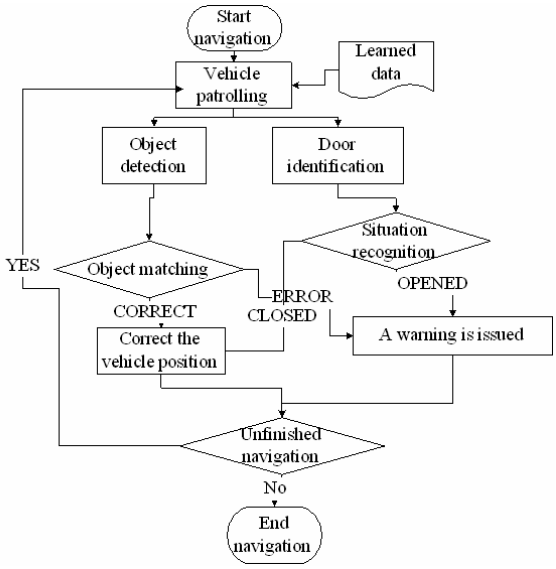


Fig. 11 Flowchart of security patrolling.

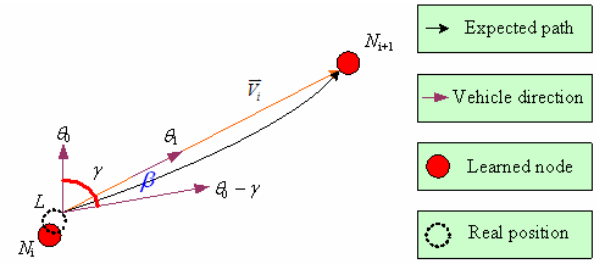


Fig. 12 An illustration of the line following technique.

Algorithm 2: *navigation by line following*.

Input: the coordinates $L(x_{odo}, y_{odo})$ and direction angle θ_0 of the vehicle provided by the odometer, and the next node $N_i(x_{i+1}, y_{i+1})$.

Output: navigation between two adjacent nodes.

Steps:

Step 1. Compute a vector $\overrightarrow{V_i}$ by using the following equation: $\overrightarrow{V_i} = \begin{bmatrix} X_i \\ Y_i \end{bmatrix} = \begin{bmatrix} x_{i+1} - x_{odo} \\ y_{i+1} - y_{odo} \end{bmatrix}$.

Step 2. Compute the direction angle θ_1 of the vehicle in the global coordinate system after the vehicle turns toward the node N_{i+1}.

Step 3. Compute the rotation angle as $\alpha = \theta_1 - \theta_0$ and the navigation distance as $d = |\overrightarrow{V_i}|$.

Step 4. Because of the mechanic error, the vehicle can not move to the correct position. A correction angle β is computed as follows by using the curve equation $y = ax^2 + bx + c$:

$$\beta = \tan^{-1}\left(\frac{a|\overrightarrow{V_i}|^2 + b|\overrightarrow{V_i}| + c}{|\overrightarrow{V_i}|}\right).$$

Step 5. Compute the real rotation angle as $r = \alpha - \beta$.

Step 6. Turn the vehicle leftward for the angle of r if r is larger than zero; otherwise, turn the vehicle

rightward for the angle of r, so that the direction of the vehicle becomes $\theta_0 - \gamma$.

Step 7. Move the vehicle forward. Read the odometer to obtain the current vehicle location L_v and compute the diatance the vehicle has moved as $d_1 = |L_v - L|$.

Step 8. End this navigation session if $d_1 \geq d$.

5.2. Object Security Monitoring Process

A matching rule is proposed to determine whether the object is exactly the same as the learned one. A detailed matching algorithm is described as follows.

Algorithm 3: *object Matching*.
Input: An image I and a learned object data O_i.
Output: A Boolean value, true or false.
Steps:

Step 1. Turn toward a learned object by using the position of the object.

Step 2. Execute the improved snake algorithm to detect objects in the image.

Step 3. If no object is detected, return "false;" else, continue following steps.

Step 4. Compute the features of the object.

Step 5. Compare the features with those of the learned object in the following ways.

A. Compare means and standard deviations of R, G, and B values.

B. Compare shape data by using parameters, a and b, of ellipse representation in the following way.

$$\left| \frac{Learn\_a}{Learn\_b} - \frac{a}{b} \right| \leq th .$$

where th is a threshold.

Step 6. If the conditions are not satisfied, return "false;" else, return "true."

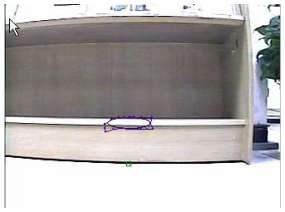


Fig. 13 An experimental result of detecting no object.

5.3. Detection of Door Opening

The algorithm of door opening detection is performed when the vehicle move to the front of a door.

Algorithm 4: *detection of door opening*.
Input: an image I and a set data of a door D_i.
Output: a Boolean value, true or false.
Steps:

Step 1. Detect edges in I by the *Sobel operator*, as shown in Fig. 14.

Step 2. Detect the edge of the door E_d and the edge of the baseline E_b and choose the right or left side of the door baseline according to learned baseline data.

Step 3. Compute the slopes of E_d and E_b by line fitting.

Step 4. Compare a_d and a_b by the following inequity:

$$|a_d - a_b| \leq th.$$

where th is a threshold, and a_d and a_b are the slopes of E_d and E_b.

Step 5. If the inequality is not satisfied, decide the door to be open, and return "true;" else, continue the following steps.

Step 6. Compute the color data by selecting a rectangular region, as shown in Fig. 14.

Step 7. Compare the color data: if not similar, return "false;" else, return "true."

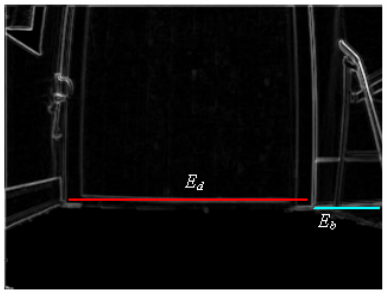


Fig. 14 An experimental result of detection of edges.

(a) (b)

Fig. 15 Situations of the door detection. (a) The door is open. (b) The door is closed

5.4. Improving Guidance Precision in Navigation by Learned Object Location

Although the line following technique and the mechanic error correction method are helpful for the improvement of the navigation accuracy, it is still possible that the vehicle moves with uncorrected deviations from normal paths after long navigation distances. A vision-based technique is proposed to correct such navigation deviations in this study. The main idea is to correct the position of the vehicle by using the learned position of the monitored object.

In the object detection process, the vehicle turns toward an object according to the position of learned object and we can get the object coordinates (Vx_o,Vy_o) relative to the vehicle, as shown in Fig. 16(a). If no mechanic error occurred, the center of the object region is at the image center. We utilize this concept to correct

the direction of the vehicle in the environment by the following equation:

$$\theta_1 = \theta_0 - \tan^{-1}\left(\frac{Vy_o}{Vx_o}\right) - \frac{\pi}{2}$$

where θ_0 is direction angle provided by the odometer, and θ_1 is the actual angle. We then compute the coordinates of the vehicle (Ox_v, Oy_v) in the object coordinate system by transforming the coordinates (Vx_o,Vy_o) by the following equation as illustrated in Fig. 16(b):

$$(Ox_v, Oy_v) = ((-1) \times Vx_o, (-1) \times Vy_o).$$

We also compute the angle ρ between the object coordinate system and the global coordinate system using the direction angle θ_1:

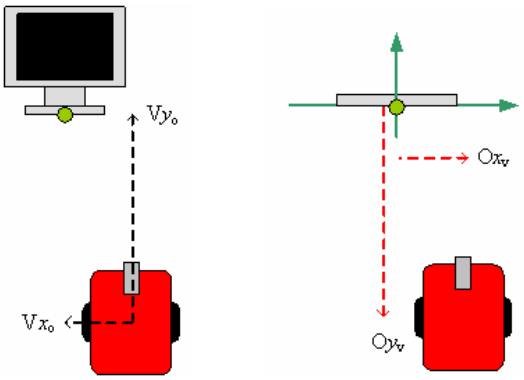
$$\rho = \theta_1 - \pi/2$$

Finally, we compute the real position of the vehicle by the following equations:

$$x_v = Ox_v \times \sin\rho - Oy_v \times \cos\rho + Learn\_x;$$
$$y_v = Ox_v \times \cos\rho - Oy_v \times \sin\rho + Learn\_y,$$

where *Learn_x* and *Learn_y* are the coordinates in the global coordinate system.

After correcting the position values of the vehicle, we compute the subsequent vehicle route by using the line following technique.

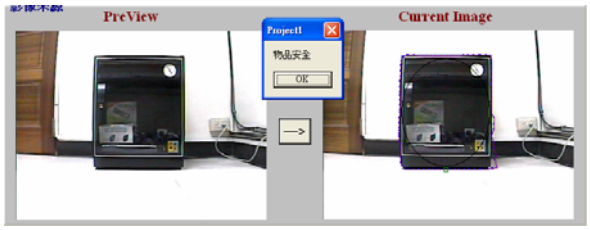


(a) (b)

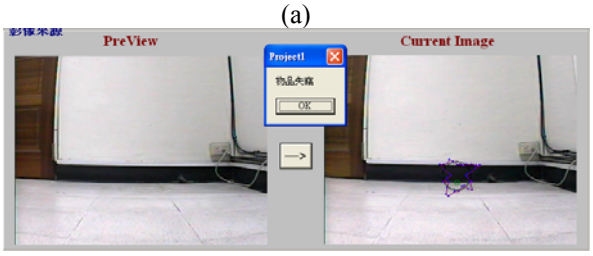
Fig. 16 Coordinate Transform between the VCS and the object coordinate system. (a) A sidelong view of the VCS. (b) A vertical view of the object coordinates.

6. EXPERIMENTAL RESULTS

Some experimental results of monitoring objects and doors in navigation sessions are shown in Figs. 17 and 18. There are two regions in the images; the left side is the view of the vehicle, and the right side is the image processing result. Some warning messages of monitoring results are shown in the image. Fig. 17(a) and (b) demonstrate that our system successfully recognizes the existence of a monitored object. Fig. 18(a) and (b) include another successful example of successfully checking door situations.

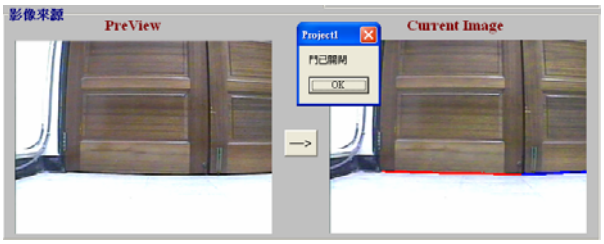


(a)

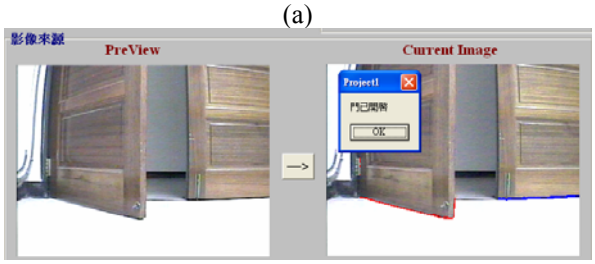


(b)

Fig. 17 An experimental result.



(a)



(b)

Fig. 18 Another experimental result.

7. CONCLUSIONS

Several techniques and strategies have been proposed in this study and integrated into an autonomous vehicle system for security patrolling in indoor environments with mechanic error correction and visual object monitoring capabilities.

At first, a vehicle location mapping method to avoid 3-D vision processing has been proposed. Also a mechanic error correction model based on a second-order curve equation has been proposed to improve navigation accuracy.

Next, some learning strategies have been proposed, including learning of the planned path and monitored objects and doors. The user can easily control the vehicle to navigate in the environment and select monitored objects in the image. And in order to make a precise navigation along a path, a method for use of the learned object location as an auxiliary tool to adjust the position and direction of the vehicle has been proposed.

In addition, several techniques have been proposed for object monitoring in the navigation, namely, object detection, recognition, searching, and door opening detection. The techniques are mainly based on a modified snake algorithm.

In the future, researches may be directed to investigations of monitoring of objects with more complicated shapes and textures on their surfaces. Monitoring of environment events is also interesting to study.

Acknowledgement

This paper is partially supported by Technology Development Program of Academia, ROC, under Grant No. 94-EC-17-A-02-S1-032.

References

[1] Michael Kass, Andrew WitKin and Demetri Terzopoulos, "Snake: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, Jan. 1988, pp. 321-331.

[2] P. C. Yuen, Y. Y. Wong, and C. S. Tong, "Contour detection using enhanced snakes algorithm," *Electronics Letters*, vol. 32, issue 3, Feb. 1, 1996, pp. 202-204.

[3] Donna J. Williams and Mubarak Shah, "A fast algorithm for active contours," *Proceedings of Third International Conference on Computer Vision*, Osaka, Japan, Dec. 4-7, 1990 , pp. 14-26.

[4] F. Leymarie and M. D. Levine, "Tracking deformable objects in the plane using an active contour model." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 6, pp. 617-634, 1993.

[5] Dong Joong Kang, "A fast and stable snake algorithm for medical images," *Pattern Recognition Letters*, vol. 20, issue 5, May 1999, pp. 507-512.

[6] Tianqing Li, Yi Zhang, Danya Yao, and Dongcheng Hu, "FFT Snake: a robust and efficient method for the segmentation of arbitrarily shaped objects in image sequences," *Proceedings of the 17th International Conference on Pattern Recognition*, vol. 2, Aug. 23-26, 2004, pp. 116-119.

[7] http://members.bellatlantic.net/~vze2vrva/ellipse_fitting.html, Ellipse Fitting.

[8] Maurizio Pilu, Andrew W. Fitzgibbon, and Robot B. Fisher, "Ellipse-specific direct least-square fitting," *Proceedings of International Conference on Image Processing*, vol. 3, Lausanne, Switzerland, Sept. 16-19, 1996, pp. 599-602.

[9] Chung-Chi Lai, "A Study on Automatic Indoor Navigation Techniques for Vision-Based Mini-Vehicle with Off-Line Environment Learning Capability," *Master Thesis, Department of Computer and Information Science, National Chiao Tung University*, Taiwan, June, 2003.

[10] Pao-Lung Li, "Path Learning, Planning, and Guidance for ALV Navigation Inside Buildings," *Master Thesis, Department of Computer and Information Science, National Chiao Tung University*, Taiwan, June, 1997.

[11] Hsien-Yi Chiu, "Automatic Vehicle Navigation and Parking in Building Corridors Using Panoramic Sensing and 2D Image Analysis Techniques," *Master Thesis, Department of Computer and Information Science, National Chiao Tung University*, Taiwan, June, 2002.

[12] Y. C. Chen and W. H. Tsai, "Vision-based autonomous vehicle navigation in complicated room environments with collision avoidance capability by simple learning and fuzzy guidance techniques," *Proceedings of 2004 Conference on Computer Vision, Graphics and Image Processing*, Hualien, Taiwan, Republic of China, Aug. 2004.

[13] Rui-Chih Liu, "Security Patrolling in Building Corridors by Multiple-Camera Computer Vision and Automatic Vehicle Navigation Techniques," *Master Thesis, Department of Computer and Information Science, National Chiao Tung University*, Taiwan, June, 2001.