

Automatic Navigation for A Mobile Robot with Monocular Vision

Qiang Zhan, Shouren Huang, Jia Wu

Robot Research Institute
Beijing University of Aeronautics and Astronautics
Beijing, 100083, China
qzhan@buaa.edu.cn

Abstract—Monocular vision based navigation method has the merits of simple computation and cheap hardware and is promising to realize real time navigation. A monocular vision based navigation method for a mobile robot moving in unknown environment is presented in the paper. By a special installation of a monocular camera on the top of a mobile robot, the method can realize the obstacle detection, distance measurement and path planning based on one single image, so as to realize automatic navigation of a mobile robot in unknown environment. Experiments on the method were done in real environment and it shows that a mobile robot can move automatically and safely under the guide of its monocular vision system.

Keywords—obstacle detection, distance measurement, local path planning, monocular vision, automatic navigation

I. INTRODUCTION

Automatic navigation is necessary for a mobile robot moving autonomously in unknown environments, no matter specific or non-specific. Detecting environment obstacles, measuring the distance between the robot and obstacles and planning a viable path are three key functions that a complete automatic navigation method must have. Obstacle detection is a very important part to realize automatic navigation. At present, most autonomous mobile robots rely on range data for obstacle detection, and popular sensors and methods for range-based obstacle detection include ultrasonic sensors, laser rangefinders, radar, stereo vision and optical flow, but they are still too computation-expensive to realize real time detection. Monocular vision-based obstacle detection has the merits of simple computation and cheap hardware and is promising to realize real time detection. Shakey, the first autonomous mobile robot developed by Stanford University, used a monocular vision system to detect obstacles by applying an edge detector to the monochrome input image so as to navigate the robot moving on textureless floor tiles. However, Shakey's environment was so specific that obstacles were uniformly coated with carefully selected colors, the lighting, walls and floor were carefully set up to eliminate shadows^[1]. Mobile robots Polly and Frankie developed by the AI lab of MIT used similar edge detectors to detect obstacles. Key factors such as texture, illumination were artificially controlled, too. There would be mistakes when reflections, shadows or similar colors exist between the floor and obstacles^[2]. Turk and Marra from MIT developed an algorithm that used the color information from one camera instead of edges to detect obstacles on ground with minimal texture. The algorithm detected obstacles by subtracting one color image from a

consecutive one, so either the robot or the obstacles must be in motion^[3]. When the ground has complex texture the method would fail. The above methods detecting obstacles according to environment appearance are affected greatly by environment factors and can only be used in specific conditions and environments. Monocular vision systems can get the depth information of environments by a corresponding point calibration method based on the transformation relations between the image coordinate frame and the robot coordinate frame. In the process of corresponding point calibration, the precision of the transformation matrix would be impaired if the calculated image coordinates of the corresponding calibration points are not precise enough, and it could even result in the fluctuation of the transformation results. The calibration should be undertaken when the position and attitude of the camera are fixed and any parameters' change would result in recalibration. Cheng and Zelinsky presented a monocular camera-based distance measurement method that the transformation equation between the world coordinate frame and the image coordinate frame could be acquired according to the geometrical relations of how the camera was mounted on a robot^[5]. However, the method requires the camera should be tilted to the extent that the entire field of view of the camera intersects the floor, and such situations are not common. Lei Guo et al proposed a monocular vision based real time distance measurement method that focuses on roads with lanes or floor with parallel lines. The pitch angle of the camera, the key parameter for calculating the distance, could be calculated under the parallel constraint^[10]. The method does not work if there are no parallel lines on the ground, such as an indoor environment. Path planning is one of the basic parts of automatic navigation of mobile robots and the main aim is to find an optimal or near optimal path without collision with other obstacles from a given beginning point to a given final point^[9-12]. Global path planning and local path planning are two main types of methods. As only part of the unknown environment can be 'seen' from the camera, local path planning would be useful here.

To present an obstacle detection method mostly independent of environment lighting disturbance (such as illumination, reflection and shadows), to measure the distance between obstacles and the robot using a moving monocular camera, and to do local path planning are the main tasks to be solved in the paper, aiming to propose a new navigation method that can guide a mobile robot to move safely in non-specific and unknown environments.

II. NAVIGATION METHOD

The proposed monocular vision based navigation method consists of three sub algorithms: obstacle detection algorithm, distance measurement algorithm and local path planning algorithm. First, some constraints reasonable for a variety of environments are made:

- ①. The ground type at the robot's initial location is favorable.
- ②. The ground is flat.
- ③. There are no overhanging obstacles.
- ④. Boundaries between ground and obstacles are visible in the image.

Constraints ② and ③ are known as the ground plane constraints, which, together with the other two constraints, imply that the nearest obstacles can be first detected by scanning a single image bottom up. These constraints also pave the way for measuring the distance between the camera and obstacles with monocular vision method.

A. Obstacle Detection Algorithm

Obstacle detection is to process the digital color image taken by a camera and locate the obstacles in the image. Obstacles boundary, namely the intersecting lines between obstacles and the ground, are used to represent the obstacles' location in the image. The boundary divides the image into parts of different meanings: parts above the obstacles boundary mean area with obstacles and parts below the obstacle boundary mean safe area.

The aim of obstacle detection is to determine the intersecting lines between obstacles and the ground and get the image coordinate points set of the lines. It consists of 4 steps as follow:

- ①. Filter the input color image with a 5×5 Gaussian filter to reduce the level of noise.
- ②. Transform RGB color space into HSI color space.
- ③. Histogram the H and S components of the reference window and current window.
- ④. Compare histograms of reference window and current window and judge whether there exit obstacles. If there do exit obstacles, get the coordinates of the obstacles' boundary.

The formula used for comparing the histograms of reference window and current window is :

$$\begin{cases} D_1 = \sum_{v=0}^{255} |h_{current}(v) - h_{bottom}(v)| \\ D_2 = \sum_{v=0}^{255} |s_{current}(v) - s_{bottom}(v)| \end{cases} \quad (1)$$

where, v is the value of H or S component, $h_{current}$, $s_{current}$ represents the histograms of H, S components in current window, and h_{bottom} , s_{bottom} the histograms of H, S components in reference window. When the value of $D1+D2$ is larger than a given value, the current window is assumed to

contain an obstacle. The threshold of $D1+D2$ is dependent on the size of the window.

Fig.1 shows the original color image, the vertical slice, the current window and the reference window. Fig.2 shows the obstacles' boundary of Fig.1, which correctly describes the intersecting lines between obstacles and the ground. In Fig.2, area above the boundary means obstacles (desks and chairs) and area below the boundary means safe ground.

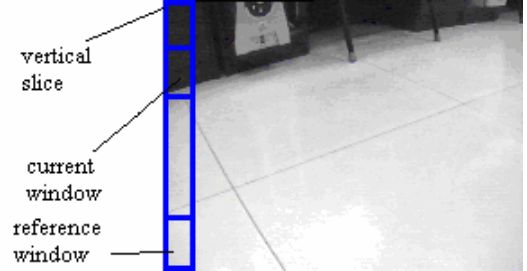


Figure 1 Original Image and Windows

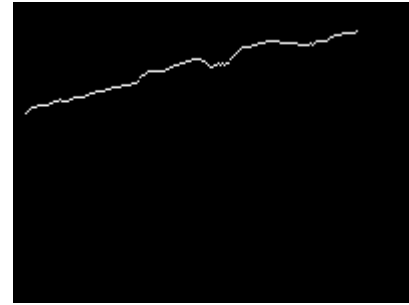


Figure 2 Obstacles Boundary

B. Distance Measurement Algorithm

The transformation equation between the image coordinate frame and the robot coordinate frame, deduced from the camera model and the geometrical relations of the robot and the camera, will be used to realize monocular vision-based distance measurement.

As showed in Fig.3, a camera is mounted on the top of a mobile robot with a certain pitch angle and the optical axis of the camera intersects with the ground. The meanings of those symbols in Fig.3 are explained as follows:

- 2α -- vertical field angle of view
- 2γ -- horizontal field angle of view
- δ -- pitch angle of camera
- b -- length of the front blind area
- d -- distance between the nearest point of view and the intersection point of the optical axis and the ground
- h -- height of the camera to the ground
- $2w$ -- width of the field of view at the intersection point of the optical axis and the ground

The following four geometrical relations can be gotten from Fig.3.

$$(2) \begin{cases} \alpha + \beta + \delta = 90^\circ \\ \tan \beta = b/h \\ \tan(\alpha + \beta) = (b+d)/h \\ \tan \gamma = w/(b+d) \end{cases}$$

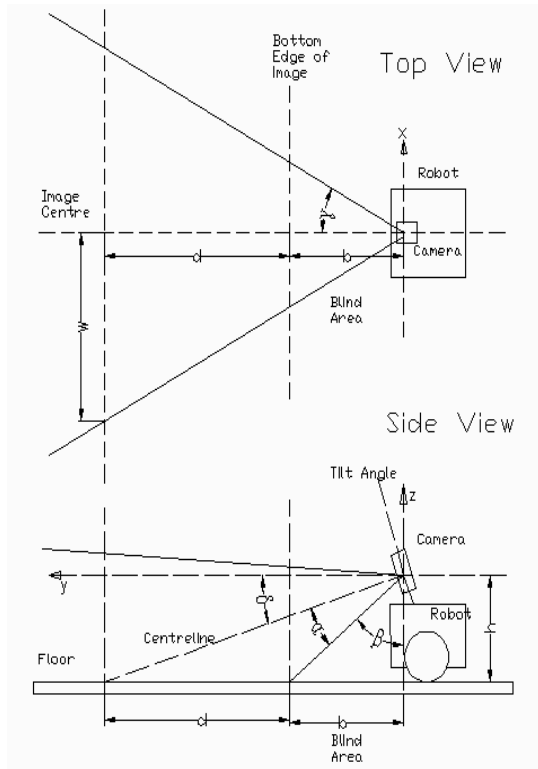


Figure 3 Geometrical Relations

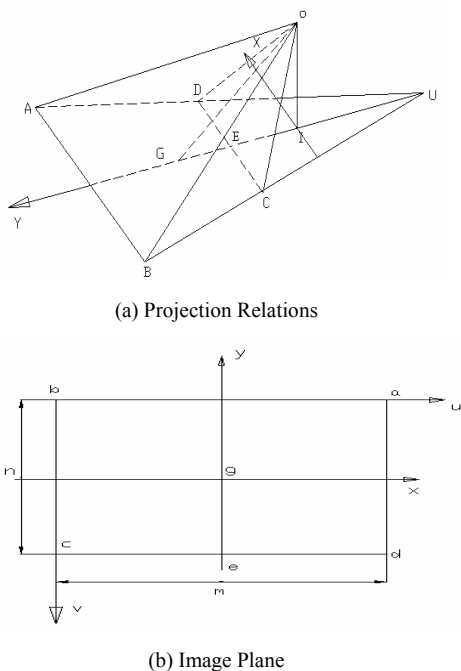


Figure 4 Camera Projection Model

According to the pinhole model, the monocular vision system could be simplified as the camera projection model, as Fig.4 shows. In Fig.4, 'O' is the optical center of the camera lens, 'G' is the intersection point of the optical axis and the ground, 'I' is the vertical projection of point 'O' on the ground, 'ABCD' represents the trapezoid field of view, 'EI' corresponds to 'b' in Fig.3, 'ABCDE' in (a) corresponds to 'abcde' in (b), 'm' and 'n' represent the width and height of the image (the image resolution is $m \times n$), 'x-y' and 'u-v' denote the robot coordinate frame and the image coordinate frame respectively.

For an arbitrary image coordinate pixel (u, v), the distance along y axis can be calculated as,

$$y = h \tan(\beta + 2\alpha(n-1-v)/(n-1)) \quad (3)$$

The x value corresponding to the u coordinate is:

$$x = y \tan(\gamma(2u-m+1)/(m-1)) \quad (4)$$

where, α , β , γ can be calculated from (2) and $m \times n$ is the image resolution.

Formula (3) and (4) decide the transformation relations between the image coordinate frame and the robot coordinate frame. Obstacle detection can obtain a one dimension array of obstacles' boundary $\{(u, v)\}$, and with (3) and (4), we could get the corresponding $\{(x, y)\}$. Here, the value of y is the distance between the robot and the obstacles, and the value of x denotes the position of the obstacles along x axis. Then the distance measurement based on monocular vision is realized by the above transformation and calculation.

C. Local Path Planning Algorithm

The color image taken by a camera shows the area where a mobile robot will move, and the size of the area and the location of the obstacles in the area can be determined by the approaches mentioned above, that is to say we can get obstacles' locations in the image by image segmentation in HSI space and get the size of the moveable area and the obstacles' location in the robot coordinate frame with the transformation equation.

Because the robot moves on a flat ground, a 2-dimension grid map is used to model the actual environment corresponding to the area taken by the camera and sequence numbers from 0 are used to mark all the grids. Each grid has two possible values: 1 or 0, if the grid is occupied by obstacles, the value of the grid is 1, otherwise 0.

The area in the image taken by the camera is trapezoidal. Directly dividing the trapezoidal area into grids would greatly increase the calculation complexity. So we first transform the trapezoidal area into a rectangular one, and then divide the rectangle into grids. The detailed grid map making steps are as follows:

- ①. Construct a rectangular area. The width and height respectively equals to the longer edge and the height of the trapezoid.
- ②. Divide the rectangle into grids with a certain size.
- ③. Set the value of each grid.

Because the transformed rectangular area includes the invisible area of the camera, and in order to correctly represent

what the camera or the robot ‘see’ the values of the grids in the invisible area of the rectangle are all set to 1.

In experiments, we found that the size of grid (shortly grid size) is a key factor affecting the result of grid map searching. If the grid size is too large, the result turns out that the area occupied by obstacles is larger than the actual situation and no moveable path can be gotten for the robot. Fig.5 has four grid maps corresponding to the original image showed in Fig.1 with different grid size. (a), (b), (c), (d) each has a grid size equals to a robot size, 1/4 robot size, 1/25 robot size and 1/100 robot size respectively. It obviously shows that (a) and (b) doesn’t agree with the actual environment, (c) basically agrees with the actual environment, and (d) is most consistent with the actual environment. The grid size chosen to model the environment depends on the field of view of the camera and the size of the robot. Because much smaller grid size will make the searching more time-consuming, a compromise between the grid size and the searching time should be reached.

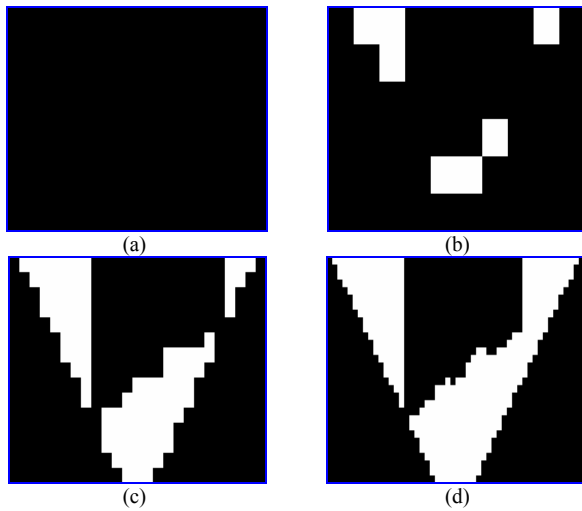


Figure 5 Grid Map with Different Grid Size

There are several searching algorithms used in grid map based path planning, such as A*. However, these algorithms are developed for global path planning, and a starting point and an end point in the grid map should be assigned firstly. For a mobile robot moving in an unknown environment, there isn’t a definite local destination to go to, so those existing searching algorithms can not be used here. A new approach for searching a movable path in local grid map is proposed in following.

Assume the array of the obstacles’ boundary is $obh[u]$, image resolution is $m \times n$. Firstly, ‘orient’ is defined as the sign of the searching direction and it can be calculated from (5).

$$orient = \frac{1}{n/2} \left(\sum_{u=0}^{n/2-1} obh(u) - \sum_{u=n/2}^{n-1} obh(u) \right) \quad (5)$$

If $orient > 0$, the searching direction is right up, if $orient < 0$, the searching direction is left up.

The searching begins at the bottom of the grid map and the starting position is:

$$start = -x_0 / gridwidth \quad (7)$$

Where, $gridwidth$ is the width of a grid, x_0 is the x coordinate of the left up vertex of the rectangle.

When the grid size is smaller than the robot, it is a necessary step to decide whether the width of successive grids without obstacles is larger than the width of the robot so that the robot could go through safely. After the whole grid map has been searched, a series of grid sequence numbers denoting where the robot should move to in the next step could be gotten and that is the safe local path for the robot.

III. EXPERIMENTS AND ANALYSIS

In order to see how the navigation method could work in real environment, an experiment system was built and experiments were done.

A. Experiment System

Fig.6 shows how the experiment system works and Fig.7, Fig.8, and Fig.9 show the components of the system.

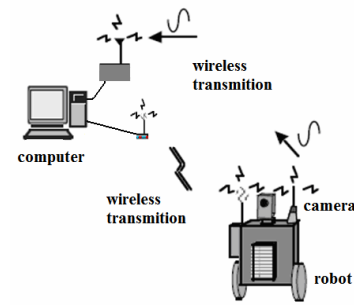


Figure 6. The Work Principle



Figure 7. Robot and Vision System



Figure 8. Wireless System

The robot in Fig.7 is a mobile robot developed by our lab. The robot has three degree of freedom on the ground and the speed is 0-100 mm/s.

The monocular vision system consists of a MINTRON 6432PD color camera, a IMAVISION DH-QP300 image frame grabber, a PC configured with Pentium(R) 4 CPU, 3.2GHz, 512MB RAM, and a tripod.

The wireless system is composed of a wireless image transmission system and a wireless communication system. The wireless image transmission system uses the 1.2GHz PT-609 wireless video transmission module as showed in Fig.8 (a). The wireless communication system uses nRF905 based PTR8000 as showed in Fig.8 (b).

B. Software Designing

The software were developed with VC++6.0 and CCS 2.0. The interface is showed in Fig.9.

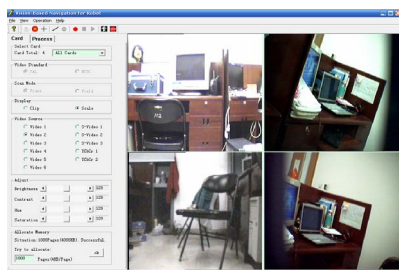


Figure 9 The software interface

C. Automatic Navigation Experiment

The following figures show the sequential images captured at interval and the navigation instructions calculated by the algorithm when the robot was navigated by the monocular vision system.



Figure 10 One image from the real-time experiment

In Fig.10, the calculation result of the proposed method navigated the robot with the instruction “Go straight ahead!” while that’s no obstacle detected within the range of the vision field.

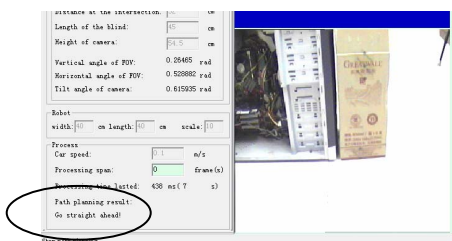


Figure 11 A later image from the real-time experiment

In Fig.11, the calculation result of the proposed method navigated the robot to go on moving ahead when there’s still a safe distance to the obstacle.

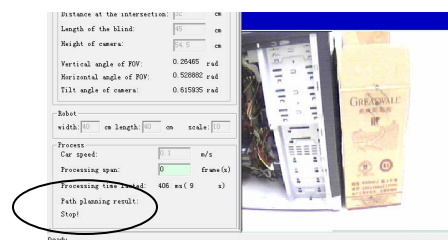


Figure 12 The robot was instructed to stop going ahead

Fig.12 showed that with the proposed method, obstacle within the unsafe area was detected, and the robot instructed to stop going a head in this direction.

These real-time results showed that the proposed monocular vision navigation method could give right instructions to guide the robot moving safely in unknown environment.

D. Processing Time Experiment

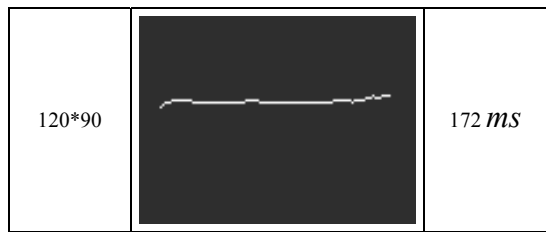
In real-time navigation, there are two factors that would impact the processing time of the algorithm: image resolution and environment.

◆ Image resolution

Image resolution is a key factor which impacts the processing time. Table 1 lists the relationship between the image resolution and the processing time.

Table 1 Image Resolution and Processing Time

Image Resolution	Processing Results	Processing Time
768*576		18266 <i>ms</i>
400*300		2359 <i>ms</i>
320*240		1516 <i>ms</i>
200*150		516 <i>ms</i>





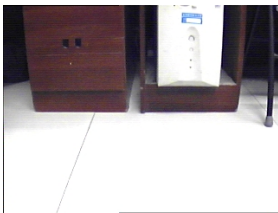

We could conclude from table 1 that the smaller the image resolution is the shorter the processing time would be.

◆ Environment

Table 2 shows the relationship between the environment and the processing time. What we could see from the list is as follows:

- a. The more complicated the environment is, the shorter the processing time becomes.
- b. The nearer the obstacles to the robot is, the shorter the processing time would be.

Table 2 The Environment and Processing Time

The Environment	Processing Time
	2031 <i>ms</i>
	1141 <i>ms</i>
	1140 <i>ms</i>
	1547 <i>ms</i>

IV. CONCLUSION

The paper proposed a monocular vision based navigation method for a mobile robot moving in an unknown and non-specific environment. The main characteristics of the method include that processing image in HSI color space and using histogram comparison to decrease the affect of lighting and detect obstacles' boundary, designing a special manner to mount a camera on a robot so as to get the distance between the robot and the obstacles by a transformation equation, using a self-defined grid map and a searching approach to model the environment 'seen' by the camera and get the local safe path. Experiments done in real and non-specific environments showed the effectiveness of the method.

Acknowledgment

This work is supported in part by National Natural Science Foundation of China under the grant No. 50705003 and National High Technology Research and Development Program of China under the grant No. 2007AA04Z252.

REFERENCES

- [1] Iwan Ulrich, and Illah Nourbakhsh. Appearance-based Obstacle Detection with Monocular Color Vision. Proceedings of the AAAI National Conference on Artificial Intelligence, Austin, TX, July/August 2000
- [2] Urban Bergquist. Color Vision and Hue for Autonomous Vehicle Guidance. Sweden: Linkoping University, Dec. 1999
- [3] Guiherme N. DeSouza, and Avinash C. Kak. Vision for Mobile Robot Navigation A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 2, Feb. 2002
- [4] T. Taylor, S. Geva, and W. W. Boles. Monocular Vision as A Range Sensor. CIMCA 2004 Proceedings, 12-14 July 2004, Gold Coast-Australia
- [5] G. Cheng, and A. Zelinsky. Real-Time Visual Behaviors for Navigating a Mobile Robot. IEEE/RSJ International Conference on Intelligent Robots and Systems, Osaka, Japan, 1996
- [6] S. Wesolkowski, S. Tominaga. Shading and Highlight Invariant Color Image Segmentation Using the MPC Algorithm. SPIE Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts VI, San Jose, USA, January 2001, pp. 229~24
- [7] Gudrun J. Klinker, Steven A. Shafer, and Takeo Kanade. A Physical Approach to Color Image Understanding. International Journal of Computer Vision. 4, 7-38 (1990)
- [8] Wladyslaw Skarbek, and Andreas Koschan. Color Image Segmentation-A Survey. Technical Report 84-32, Technische Universitt Berlin, Oct.1994: 9~16
- [9] W. Elemenreich, L. Schneider, and R. Kirner. A Robust Certainty Grid Algorithm for Robotic Vision. 6th IEEE International Conference on Intelligent Engineering Systems (INES), May 2001, Opatija, Croatia
- [10] Lei Guo, Youchun Xue, Keqiang Li etc.. Study on Real-time Distance Detection Based on Monocular Vision Technique. Journal of Image and Graphics, Nov.2006, Vol.11, No.1
- [11] Xizhu Bo, Bingrong Hong. A Method of Collision Avoidance Planning for Multi-Robots in Dynamic Environment. Robot, Sept.2001, Vol.23, No.5
- [12] Hongbin Yu, Xiaolan Li. Fast Path Planning Based on Grid Model of Robot. Microelectronics and Computer, 2005, Vol. 22, No. 6
- [13] afael C. Gonzalez. Digital Image Processing (Second Edition). Beijing: Publishing House of Electronic Industry, 2003.3