Visual Servoing for Robot Navigation

P. Hong, H. Sahli, E. Colon, Y. Baudoin
Royal Military Academy
Free University of Brussels
HUDEM Project

ABSTRACT
This paper presents an integrated visual servoing system for robot navigation. This system is able to pursue a moving object by controlling a camera mounted on a pan/tilt head, so that the moving object is maintained in the center of the image. The visual system has four capabilities: the target detection, the target motion model online identification, camera control for target tracking and target position estimation. In order to minimize the time required for the image target detection, the target is made of elementary features: colored circular object. The target detection consists of two stages algorithm: (i) a color classification stage, and (ii) a knowledge-based shape detection stage. The color classification stage utilizes the distribution of the target color in the HSV color space in order to obtain an initial set of candidate regions. The second stage of the detection scheme uses mathematical morphology operators for circular object detection. The camera control exploits the detection in conjunction with an affine fit between 2 consecutive images. After the affine fit has been made, the camera control parameters are estimated. Due to the fact that the perspective projection is a many to one mapping, we designed the servomotor-camera-target system as a time variant system. A two phases control strategy is implemented. During the first phase (initialization) the target dynamics is estimated. The second phase consists of a state feedback control strategy. The target depth is estimated by using the appearance similarity between the target and its image. This system runs continuously in time and updates the target localization at a frame-rate of 170μs on a Pentium 400 MHz PC.

1. INTRODUCTION

This report summarizes our development in the visual servoing system for robot navigation. In our approach a colored target is put on the top of the robot and a fixed camera is used to detect and track the target. Then from the camera parameters and the target image we can estimate the robot 3D position for robot navigation (see Figure 1).

In the following sections we will discuss the implemented methods for target detection, motion model identification, target tracking and target position estimation. A hue method is

used for target detection, and an observer-based-full-state-feedback control is used for target tracking.

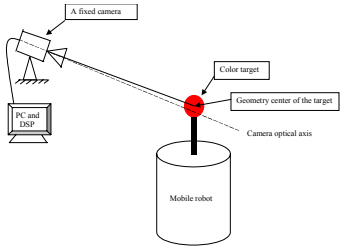


Figure 1. Robot 3D Position Estimation

2. TARGET DETECTION

In order to detect a given color in an outdoor scene we have to deal with illumination changes. We suggest a new approach for colored target detection on which the hue color is used to distinguish between different colored objects.

2.1 Color Detection

In a learning phase a hue interval corresponding to the color of the target object is estimated, then this interval is used for (R,G,B) pixels classification. In the RGB space we define two planes, which correspond to the estimated hue interval. The target detection is simply obtained by scanning the image and testing if a given RGB pixel lies between the two defined RGB planes of a given color target object. Figure 2 shows the detected red target region. For a real time application speed is vital important. We do this classification in RGB space directly. In this method, we first describe the normal of vertical planes in a specified RGB value. Then, for each pixel, in RGB space we calculate the dot product of its vector and the normal of the vertical plane. By comparing the calculation result with zero we can know which hue section this pixel belongs to. With this method we can detect the color target, as shown in Figure 2.

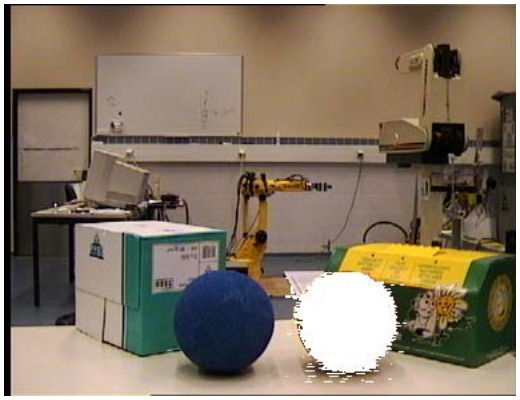


Figure 2. Detected Red Object

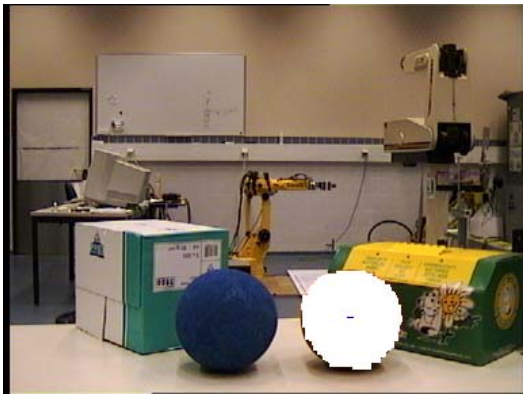


Figure 3. After Morphology Filtering

2.2 The Noise Suppression and Image Segmentation

In an image, the worst case happens for color detection when the pixel is too dark or too close to white. For these pixels their hue values that are calculated from their (R,G,B) data are very sensitive to noise and these pixels do not have much color information. We use threshold to omit them. Figure 2 is obtained after using this method. From Figure 2 we can see that the image still contains a lot of noise.

To solve this problem, we use morphology filtering. During the color detection we create a corresponding binary image. For this binary image we use morphology filtering to do image segmentation. A square mask of 5 by 5 pixels is used as the structuring element in our application. Figure 3 shows the processed result. The white color shows the detected region after image segmentation; and all the pixels in this region are considered as the detected pixel. The region is connected and well represents the target shape.

2.3 Hue Adaptation

The two hue limitations are regularly adapted to cope with illumination variations.

2.4 Position and Size Estimation

The target position in an image is estimated by calculating the geometry center ($\bar{x}$, $\bar{y}$) of the detected pixels. The geometry center is shown as a short blue line in Figure 3. We use two kinds of definition to define the detected target size: one is the radium of the target image and it is used for target depth estimation; the other is the second order moment of the detected target boundary pixels and it is used for window (Region Of Interest) tracking, that is

$$\mu_x^2 = \frac{1}{N} \cdot \sum_{i=1}^{N} (\tilde{x}_i - \bar{x})^2 \tag{2-1}$$

$$\mu_y^2 = \frac{1}{N} \cdot \sum_{i=1}^{N} (\tilde{y}_i - \bar{y})^2 \tag{2-2}$$

where

-(μ_x^2 , μ_y^2) the second order moment in x and y direction respectively.

-($\tilde{x}_i$, $\tilde{y}_i$) the coordinates of i th pixel which is on the boundary.

- N the total number of pixels which are on the boundary.

3. TARGET TRACKING

In fact, the target-tracking problem can be regarded as a visual servoing problem. In our system the target is mounted on a mobile robot. A calibrated camera fixed at the origin of the world frame is controlled through its pan (α) and tilt (β) angles to bring the target image center onto the image plane center. Figure 4 depicts the camera platform and Figure 5 shows the defined camera control parameters.

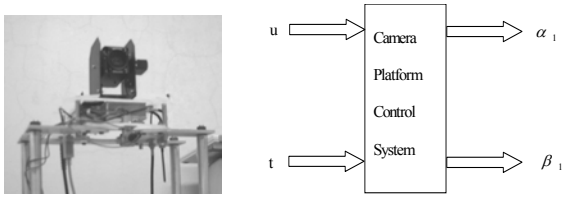


Figure 4. Camera Platform Control System

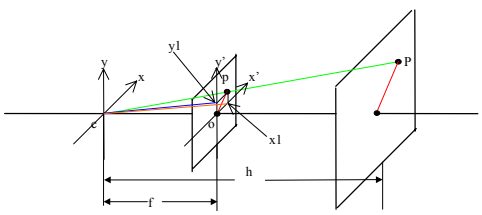
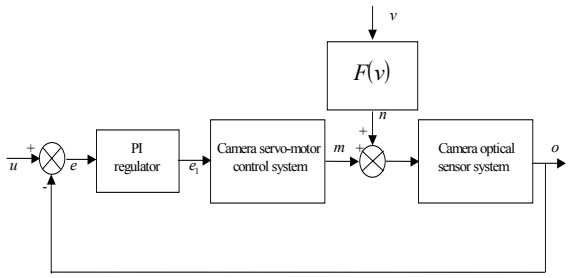


Figure 5. The Camera Control Parameters: $\alpha = \angle ocx_1$ and $\beta = \angle ocy_1$.

Due to the fact that the robot moves with an unknown model, the servomotor-camera-target system is a time variant system. The target motion model has to be identified in real time.

In order to meet the system dynamic characteristic requirements we developed a two phases control strategy. An initialization phase, in which the mobile dynamics is estimated, is used to track the target with a PI regulator. The block diagram of this phase is given in Figure 6.



u: image plane center.
v: target movement.
n: the noise caused by the target movement.
$F(v)$: transfer function, which represents the relationship between v and n.
m: the movement of the camera optical axis.
o: target image center.

Figure 6. Initialization System Block Diagram

The second phase control consists of a feedback control strategy shown in Figure 7. The plant is modeled as a dynamic system shown in Figure 8.

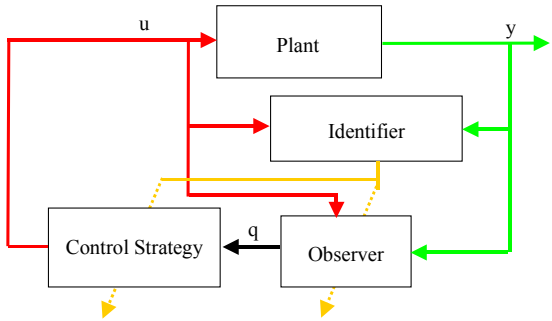


Figure 7. Observer Based State Feedback Control

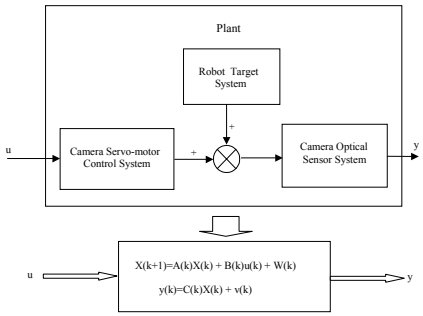


Figure 8. Dynamic System Model

In our implementation a second order difference model is considered. The system functions are

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -a_0(k) & -a_1(k) \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k) \tag{3-1}$$

$$y(k) = \begin{bmatrix} b_0(k) & b_1(k) \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \tag{3-2}$$

where

- (x_1, x_2) is the state vector corresponding to the camera angles (pan or tilt) and angular velocity.
- (a_0, a_1, b_0, b_1) are the system parameters to be estimated.

These parameters are estimated using LSM method from a set of input image frames and camera control parameters.

The feedback control strategy is implemented with system state vector estimation using Kalman Filtering. The detail of the observer-based-full-state-feedback-control system configuration is shown in Figure 9.

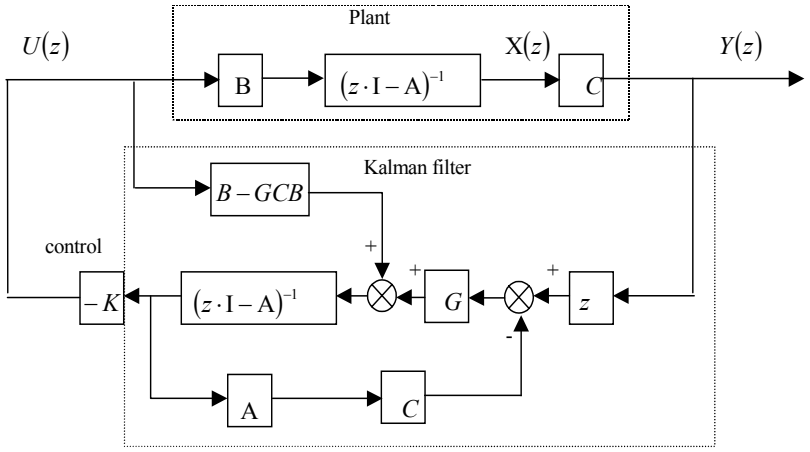


Figure 9. The Observer Based Full State Feedback Control System

where

A the plant system matrix given by $[a_0, a_1]$

B the plant input matrix $[0,1]$

C the plant output matrix given by $[b_0, b_1]$

G the Kalman filter gain matrix

K the control gain matrix defined as the difference between the required and identified system parameters of characteristic functions (the pole-assignment method).

4. WINDOW'S POSITION AND SIZE ESTIMATION

For a good target detection and tracking we need to use high sampling speed and to increase the signal-to-noise ratio. To achieve these purposes one way is to reduce the image size. Instead of using the whole image for target detection and tracking we create an image window (a Region Of Interest) and let its size only be a little bigger than enough to contain the target image. Only the image inside of the window will be processed for target detection and tracking. First, this will be of benefit to the target tracking process, because this will increase the sampling speed of the system. Second, with this kind of window the target has almost occupied the whole window area, the signal to noise (which is caused by the environment, for instance the color of other objects) ratio will be very high. From this point of view, this will make the detection and tracking methods much more robustness.

The best way is to use $\bar{x}$ and $\bar{y}$ of the next target image as the window's center and to use the second order moments as the estimator of window's size, that is,

$$l = C_1 \cdot \mu_x^2 + 2 \cdot \varepsilon \tag{4-1}$$

$$h = C_2 \cdot \mu_y^2 + 2 \cdot \varepsilon \tag{4-2}$$

where

l window's length

h window's height

C_1, C_2 scale factors

ε tolerance

Due to the camera's tracking activities, it is more difficult for the window's parameter prediction; and the system is a time variant one. We use Least-Mean-Square (LMS) adaptive filter as the predictor. In the following, we only use the position prediction in X direction as an example to describe the method; the same principle holds for both direction and window's size prediction.

In our application, we define the real position estimation output as the desired system output, that is,

$$d(k) = \bar{x}(k) \tag{4-3}$$

We also define the difference of the real position output and the control command as the filter input, that is,

$$u(k) = \bar{x}(k) - x_{co}(k) \tag{4-4}$$

where

$\bar{x}(k)$: the estimated target position in X direction at instant k

$x_{co}(k)$: the control signal of X direction at instant k

The filter's output is used as the position prediction of X direction.

5. TARGET POSITION ESTIMATION

The origin of world frame is set at the center of the camera. The camera platform is kept horizontal. Then, the position of the target can be described by 3 parameters: the horizontal angle, the vertical angle and the distance between camera and target. Angles are calculated using the pose of the camera and the orientation angles of the target image in the camera coordinate system. The distance between camera and target is estimated by simple similar triangle relationship of real target size, detected target image size and effective camera focal length.

6. THE EXPERIMENTAL RESULTS

In our experiments, the task of image acquisition and processing is done by a PC with a image grabber. The image processing time is associated with image size. We use the zooming function of the camera to keep the target image in the same size. For a good target image size, say 76 pixels in diameter, in a 384x288 image, the tasks of image processing, system model identification, system state observation, full state feedback control and window's position and size prediction are done by a personal computer (with 400 MHz Pentium microprocessor) within 0.17 second.

The following figures show a test result of the observer-based-full-state-feedback-control phase. The target moved (with an angle, not directly) first towards the camera and then went away from the camera. All the units of magnitude used in the following target tracking discussions are in pixels. All data are recorded under the condition that both camera tracking and window prediction function are active, but without zooming.

Figure 10 shows the target tracking error in X direction. This error shows the tracking ability of this system. The error shows a little increment when the target moves closer to the camera; and it decreases when the target moves away from the camera. The window position prediction error in X direction is shown in Figure 11. From the error chart we can see that

the prediction is good, even if there is an influence of the tracking activity. The time variant property has been removed by LMS filter.

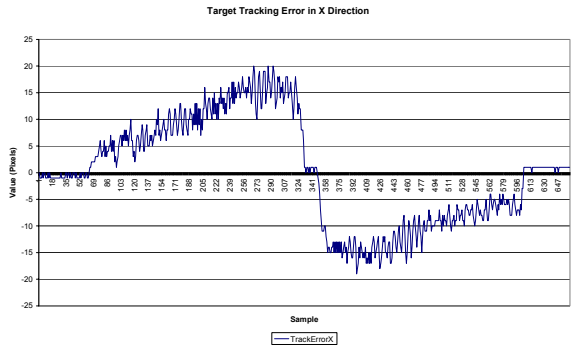


Figure 10. The Tracking Error in X Direction

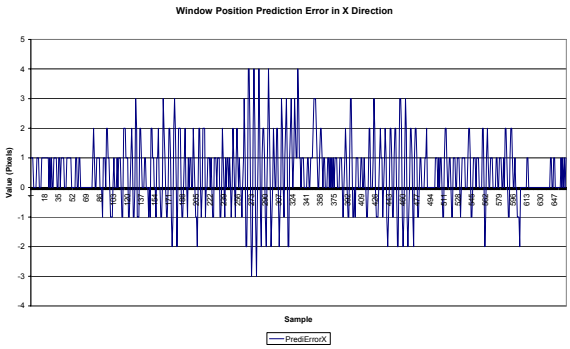


Figure 11. The Window Position Prediction Error in X Direction

The properties of window's position prediction error are summarized in Table 1.

	Mean	STDVar.	Maxi.	Mini.
Predict. Error X	0.4795	1.1058	4	-3
Predict. Error Y	0.4599	0.4988	1	0

Table 1. Prediction Error Properties

Figure 12 and Figure 13 show the prediction results of window height and length, respectively. From the figures we can see that the window's size becomes larger when the target is closer to the camera and it becomes smaller when the target goes away from the camera. Obviously, there are four, nearly symmetric, noise pulses in these two figures. They are caused by the background of the test scene.

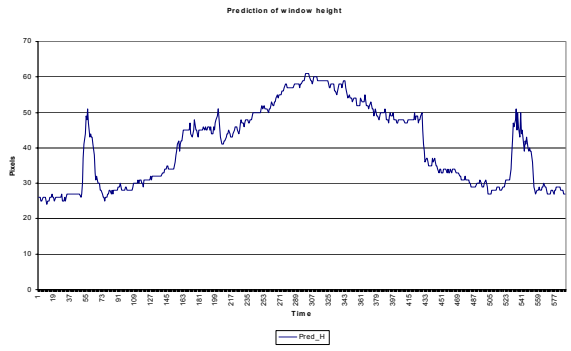


Figure 12. The Prediction of Window Height

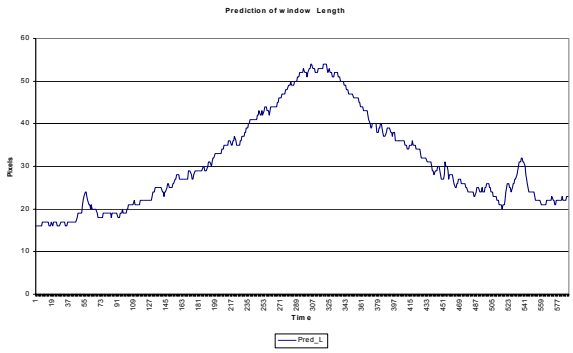


Figure 13. The Prediction of the Window Length

We also made some indoors, outdoors static and dynamic target position estimation trials with this system. Figure 14 shows one scene of an outdoors static trial. The target detection function works well (detected pixels are painted white).

Figure 14. Target Detection under Outdoor Environment

Figure 15 shows the estimated position errors caused by distance estimation error (assumed that the angle estimations are perfect) and by angle estimation error (assumed that the distance estimations are perfect), respectively. For a target that is about 8.4 meters away, the maximum position error caused by distance estimation error is less than 0.2 meter (the upper curve). The lower curve shows the value and variation of position estimation error caused by angle estimation error only. The maximum position error caused by angle estimation error is less than 0.03 meter, for the same target position. The variation caused by angle estimation is much less than that caused by distance estimation. It is obvious that the position error caused by distance estimation error is much bigger than the position error caused by angle estimation error.

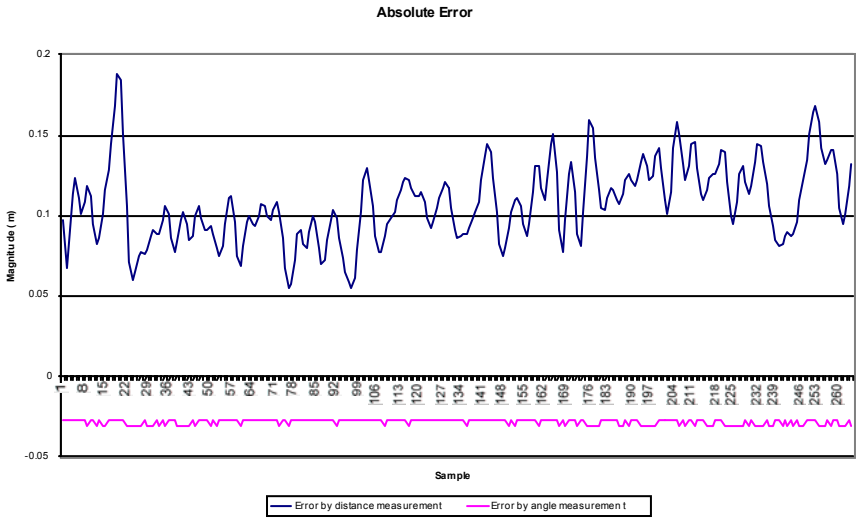


Figure 15. The Position Error Caused by Distance Estimation Error and by Angle Estimation Error

After the static position estimation trials, we put the target on the mobile robot and did some indoor and outdoor dynamic position estimation trials.

Figure 16 shows the indoor position estimation results compared with the real ones. Curves indicated by vision are estimated by our vision system. In the chart, we can see that the estimated robot positions are very close to its real positions.

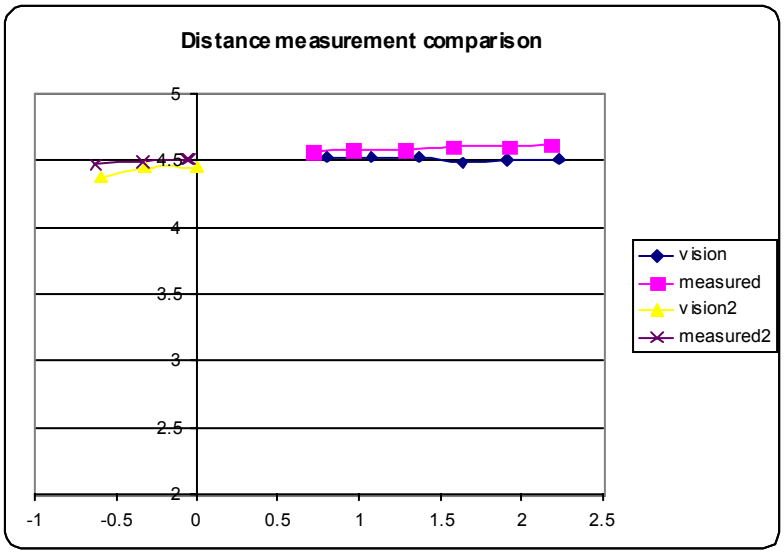


Figure 16. The Position Estimation Result (indoor)

Figure 17 shows the outdoor position estimation results compared with the real ones. We also can see that the estimated robot positions are very close to the real ones.

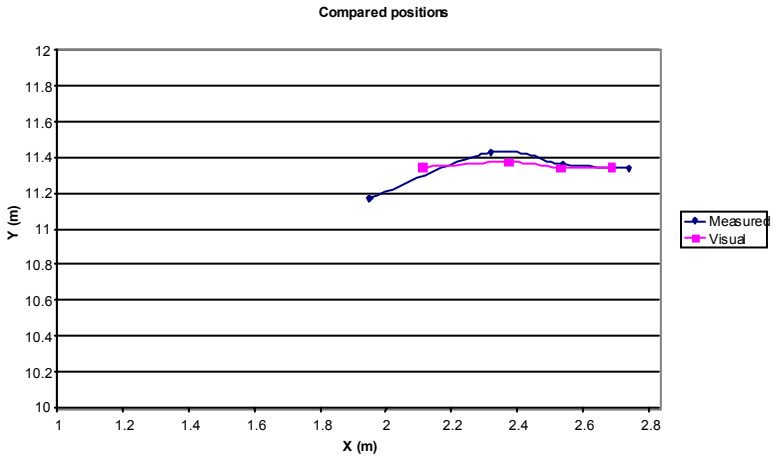


Figure 17. The Position Estimation Result (outdoor)

Improvement

To improve the position estimation accuracy further, we should solve the problems caused by the change of illumination. Under direct sunshine, the target cannot be fully detected and there are much more noises due to the illumination and the environment. Figure 18 shows an example of one of these situations.

Figure 18: Detection results Under Direct Sunshine

From this figure, it can be seen that, even with an umbrella to avoid direct sunshine on the target, the upper part of the ball is not detected. The reason for this result is that the upper part of the target becomes brighter and the lower part of the target becomes darker. Due to the strong sunlight, the saturation values of the pixels on the upper part of the target are reduced. For these pixels, under the fixed saturation threshold, the program can no longer detect them. However, this picture also gives us the cue of solution of this problem-it likes a circle cut by a line. Therefore, instead of using the detected target area to estimate the radium of our target image, we let an ellipse fit the edge points of the detected target image region using Least Square method, as shown in Figure 19.

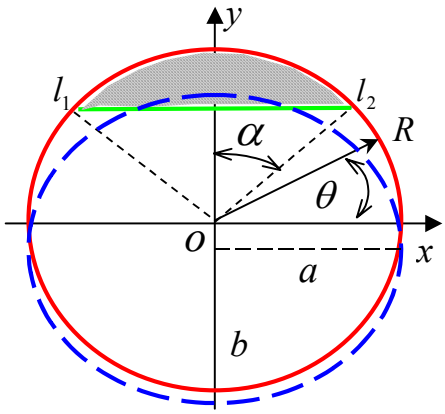
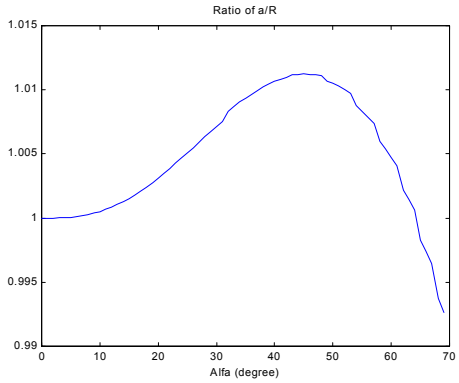
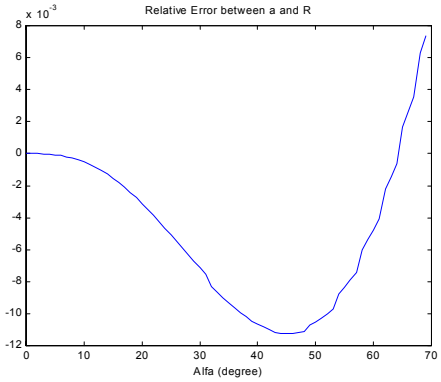


Figure 19. The Ellipse Fitting Method

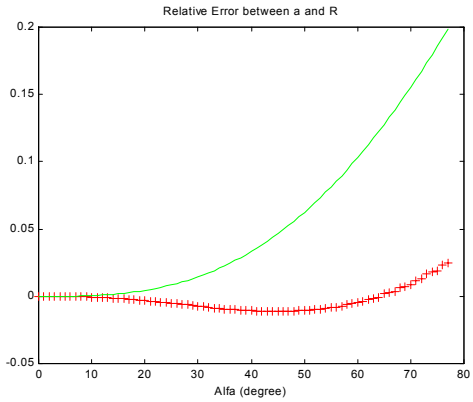
Where R is the radium of the target image (a circle), $2\cdot\alpha$ is the angle spanned by the line $l_1 - l_2$, a is the long semi-axis of the ellipse and b is the short semi-axis of the ellipse. The long semi-axis of the ellipse is used as the estimate of the radium of the target image, and Figure 20 shows the computer simulation results for different angle α. Figure 20(a) shows the Ratio of a over R. Figure 20(b) shows the Relative Error between a and R. Figure 20(c) shows the comparison between the ellipse method (red-cross curve) and the area method (green curve). The ellipse method gives very good estimation results.



(a): Ratio of a over R



(b): Relative Error between a and R



(c): Comparison of the Relative Estimation Error between the two Methods

Figure 20. Computer Simulation Results

From Figure 20(b) we can see that if we shift the calibration point (the image used for calibration has already been deformed, as shown in Figure 21) when we calibrate the visual system, we can make the relative estimation error less than 0.5% for vary large region of α. Figure 21 is used for visual system calibration and Figure 22 is used for target depth estimation. The estimation results are given in Table 2. From these two images, we can see that the detected target image regions are quite different from each other, but the estimation results are very good, as shown in Table 2.

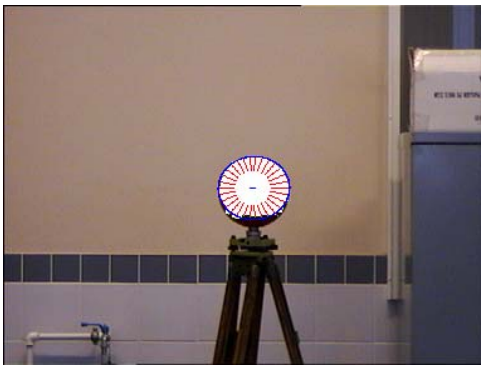


Figure 21: Image Used for Calibration

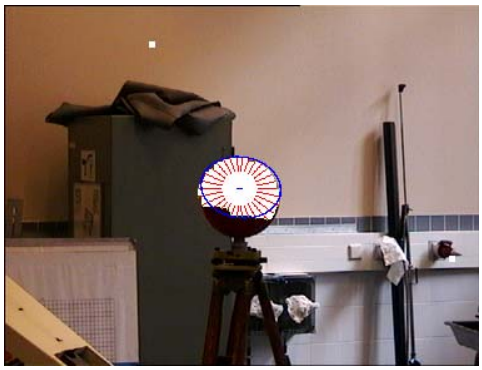


Figure 22: Target at Estimated Point

Real Distance (m)	Measurement Mean (m)	Standard Variance (m)	Relative Error
4.532	4.534	0.0464	0.054%

Table 2. Estimation Results of Ellipse Method

7. CONCLUSIONS

Because the parameters of the system model are estimated in real time, it is enough to use only a linear second order difference model to approximate the real model of the system in our application. The functions of target tracking and target position estimation worked very well in indoor and outdoor trials. We also developed a simple, but very robust ellipse fitting method for distance measurement to improve the accuracy of our system and received good results.

REFERENCES

1) F. Bumbaca and K. C. Smith, Design and Implementation of a Colour Vision Model for Computer Vision Applications, *Computer Vision, Graphics, and Image Processing 39*, 1987, pp. 226-245.

2) R. C. Gonzalez and R. E. Woods, *"Digital Image Processing"*, Addison-Wesley, 1992.

3) N. Hollinghurst and R. Cipolla, "*Uncalibrated stereo hand-eye coordination*", Image and Vision Computing, Volume 12 Number 3, April 1994, pp. 187-192.

4) Mengxiang Li, "*Camera Calibration of a Head-Eye System for Active Vision*", Lecture Notes in Computer Science, Vol. 800, Jan-Olof Eklundh (Ed.)

5) Charles L. Phillips and H. Troy Nagle, "*Digital Control System Analysis and Design*", Second Edition, Prentice-Hall Inc.,1984.

6) D. E. Catlin, "*Estimation, Control, and the Discrete Kalman Filter*", Springer-Verlag New York Inc., 1989.

7) W. S. LEVINE, "*The Control Handbook*", CRC Press Inc., 1996.

8) W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, "*Numerical Recipes in C*", The Art of Scientific Computing, Cambridge University Press, 1992.

9) Simon Haykin, "*Adaptive Filter Theory*", Third Edition, Prentice Hall, 1996.

10) Markus Vincze, "*Robust tracking of ellipses at frame rate*", PATTERN RECOGNITION, The Journal of the Pattern Recognition Society, Volume 34, Number 2, pp. 487-498, February 2001.