

# Decrease the Dimension of Detecting Circles and Ellipses with Hough Transform

<sup>1</sup> Chiao-Ssu Liao (廖喬恩), <sup>2</sup> Chiou-Shann Fuh (傅楸善), <sup>3</sup> Wei-Lun Huang (黃偉倫)

Department of Computer Science and Information Engineering,  
National Taiwan University, Taipei, Taiwan.

E-mail: {r97085, fuh, d97012}@csie.ntu.edu.tw

## ABSTRACT

Hough transform is often used to detect circles and ellipses. Many kinds of shapes could be detected only if equations could be offered. But as the number of unknown variables increases in our equations, it takes more time to detect. For example,  $(x-a)^2 + (y-b)^2 = r^2$  is used to detect circles, and its dimensions are three. Equation  $(x \times \cos(\theta) - y \times \sin(\theta) + x_0)^2 / a^2 + (x \times \sin(\theta) + y \times \cos(\theta) + y_0)^2 / b^2 = 1$  is used for solving ellipses' cases, and its dimensions are five. As the dimensions increase, the complexity of our program will also increase. Detecting shapes became inefficient because of the curse of dimensionality. In this paper, we try to decrease dimensions of detecting circles and ellipses with Hough transform.

## 1. INTRODUCTION

Hough transform offers a good way to recognize shapes even if abundant information about the objects were not given to us. With the feature, we can detect circles, lines, and even ellipses by using transformation models and complete detection through voting procedure. Unfortunately, the unknown variables are more than three and even five in the case of the circle's radius or the ellipse's axes are uncertain. The algorithms will be too complex to be used in real-time detecting system with the growth of unknown variables. An idea is assumed here to decrease the dimension of detecting uncertain radius, semi-major axis and rotation angles for the circle and the ellipse.

## 2. BACKGROUND

In the case of detecting uncertain circles, we use the equation  $(x-a)^2 + (y-b)^2 = r^2$ . After transformation, it becomes the voting procedure of three variables: a, b,

and r. The complexity will be  $O(n^3)$ . When it refers to the case of ellipses, the equation is  $(x \times \cos(\theta) - y \times \sin(\theta) + x_0)^2 / a^2 + (x \times \sin(\theta) + y \times \cos(\theta) + y_0)^2 / b^2 = 1$ . After transformation, variables increase to five:  $x_0$ ,  $y_0$ , a, b, and  $\theta$ . Moreover, complexity will be  $O(n^5)$ . A way will be found to decrease the complexity of algorithms based on Hough transform which can help us to detect circles and ellipses in some uncertain situation and rotation angles. The most difficult of this report is decreasing dimension of the algorithm. In order to accomplish the goal, we have to decide an approaching estimate with the concept of connected components to make estimate trend accurate.

## 3. PREPROCESSING BEFORE DETECTING

We will detect unknown radius circles with Hough transform. The goal of our experiment is to use less time than original Hough transform. Before detection, the first step is to do the edge detection. Now we use a picture of two coins [3] as an example:



Fig. 1 Original coin picture.

After edge detection:

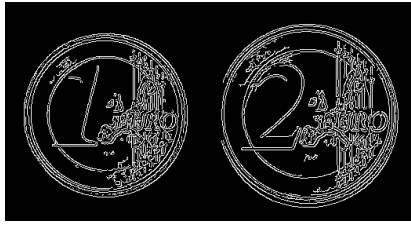


Fig. 2 After edge detection.

Dilation will be the next step to deal with the picture after edge detection which can fill the broken segment in to offer better information.

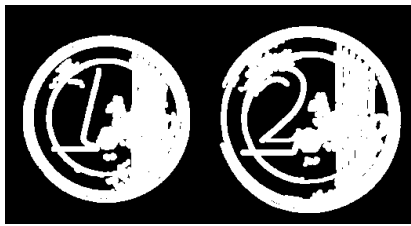


Fig. 3 After dilation.

Using `cvcontour()` or `blob()` to find the connected component:

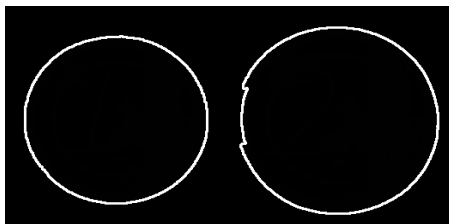


Fig. 4 After finding connected component.

Then we can use the width and height of the components to speculate the radius of every circle.

#### 4. ORIGINAL VS. DECREASING DIMENSION FOR DETECTING CIRCLES

Pictures of two formats are offered to detect circle:

Size 1: 750\*550 pixels

Size 2: 320\*256 pixels

The patterns are the same in the two pictures which are shown as follows:

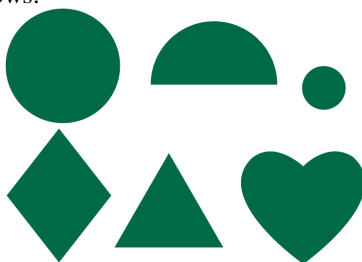

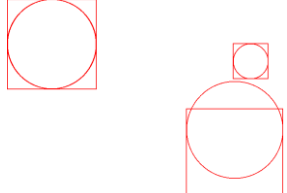


Fig. 5 Patterns in the pictures.

Original Hough transform: It is necessary to input the range of possible radius. We give it a range from 10 to the  $\max(\text{size of image.width, size of image.height})/2$ .

After decreasing dimension: We will save every connected component and extract the width and height information. It is used to judge if it is possible a circle and the radius is half of the length. The speed of comparison is shown in Table 1:

Table 1 The comparison of before and after decreasing dimension.

	Original Hough transform
Size 1	376328 ms
Size 2	20875 ms
Detecting result (using size 1)	
	After decreasing dimension
Size 1	421 ms
Size 2	93 ms
Detecting result (using size 1)	

When using Hough transform to detect circle, the parameter about the number of voting can increase the accuracy. In other words, Hough transform judges if it is a circle by the number of voting which can be given by us. For example, if we set it to be 10, then every pixel its voting number must be larger than 10 will be regarded as circles. Because of using the method of connected component, the parameter could be adjusted by the length of component. With this method, the small size component will not be ignored. In our experiment, it is set as  $1.5 \times$  square component length. The heart pattern shows in Fig. 5 could be detected correctly if we change the parameter as the scale we referred before.

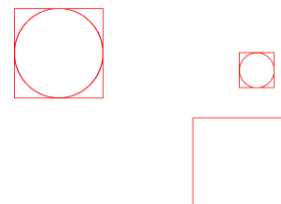


Fig. 6 More accurate result after adjusting.

## 5. DETECTING ELLIPSES WITH DECREASED DIMENSION METHOD

Now it is tried detecting ellipses using Hough transform. The method is similar to before, and the steps are: edge detection, dilation, finding connected components. Then if the width and height of the contour are different, the shorter one will be chosen as the new square. The new square is the resize contour comes from original contour.

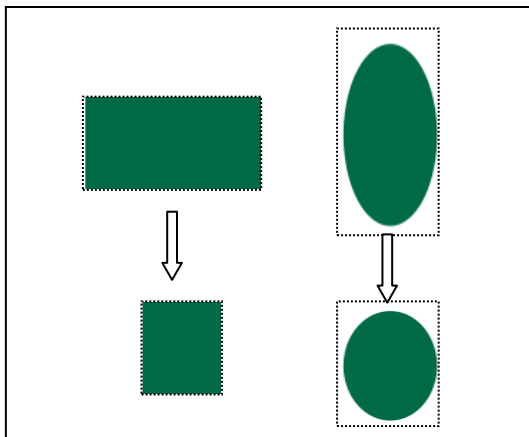


Fig. 7 Resize rectangle contour with shorter edge.

Detecting un-rotated ellipses are easier than rotated ones because we can use the width and height of the un-rotated ellipse's blob as the semi-major axis and semi-minor axis which can not be known for the rotated ellipses. Let us start with the un-rotated cases:

Pictures of two formats are offered as before to detect un-rotated ellipses:

Size 1: 750\*550 pixels

Size 2: 320\*256 pixels

The patterns are the same in the two pictures shown below:

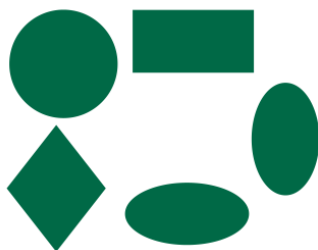


Fig. 8 Patterns for detecting ellipses (Ellipses in the picture are un-rotated).

As shown before, it takes much more time to detect circles on original Hough transform than the other because of dimensions. In the case of ellipses, dimensions will increase to four or more. The speed could be estimated to be much slower than improved one according to the data we measured before. Using the method of resize, rectangle contours become square ones, and ellipses will be circles for Hough transform. The parameter about the number of voting is adjusted to be 2\* square component width for getting accuracy result. The detecting results of two sizes are shown in Table 2:

Table 2 The result of detecting circles and un-rotated ellipses.

	Size 1
Detecting Result	
Time	1984 ms
	Size 2
Detecting Result	
Time	906 ms

We will try to detect rotated ellipses with the method we used to detect circles and un-rotated ellipses. If the blob is judged not to be circles or ellipses, we will calculate the center of this blob. Then, the blob is searched by column to find the highest point its pixel is not null of every column. To find the farthest distance from center of the blob to the highest point in every column of the object and the fraction of

$|x \text{ value of the point} - x \text{ value of the center}|$  and the farthest distance is  $\cos(\theta)$ . With the value of  $\cos(\theta)$ , the angle of rotation could be speculated. The location of the farthest point and the center point could be the information for rotation with clockwise or counterclockwise.

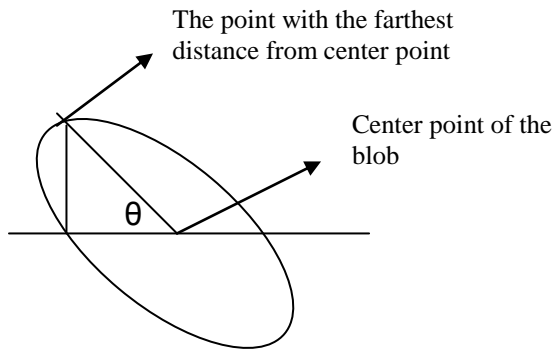


Fig.9 The relation between center point, farthest point from the center point, and rotating angle  $\theta$  .

The result of detecting rotated ellipses using the method we referred before is shown as follows:

Table 3 The result of detecting circles and ellipses.

Size	750*550 pixels
Original picture	
Time	3437.5ms
Result	

We can find circles, un-rotated ellipses, and rotated ellipses in the picture shown in Table 3. Rotated ellipses are drawn as un-rotated ellipses and it takes 3437 ms. Using the original Hough transform referred in Table 1 which can only detect circles to detect the same picture shown in Table 3, it takes 368296 ms. A lot of time is saved obviously.

## 6. CONCLUSION

We used edge detection, dilation, connected components to decrease dimension of detecting circles and ellipses for Hough transform. The speed of detecting circles and ellipses could be improved, especially for eclipses' cases. It is shown above that with this method, we can detect circles 200-800 times faster than original Hough transform for uncertain radius cases. The dimensions for detecting ellipses are five which must take much more time than the method we used in the Table 3. If we compare the cases for ellipses of decreased dimensions with the ones for circles without decreased dimensions, cases of detecting ellipses are about 107 times faster than cases without decreased dimensions. The time we save will be more obvious as the image size grows. The result provides us with a possible way to decrease dimensions when using Hough transform.

## 7. REFERENCES

- [1] 廖俊鑑, 以改良式霍夫轉換為基礎的快速圓形／圓弧偵測方法之研究, 2005.
- [2] R. M. Haralick and L. G. Shapiro, Computer and Robot Vision, Vol. I, Addison Wesley, Reading, MA, 1992.
- [3] 葉高華, 歐元硬幣的秘密, <http://richter.pixnet.net/blog/post/17652578>, 2008.
- [4] Ballard, D. H., "Generalizing the Hough transform to Detect Arbitrary Shapes," Pattern Recognition, Vol. 13, No. 2, pp. 111-122, 1981.
- [5] Atiquzzaman, M., "Multiresolution Hough Transform – an Efficient Method of Detection Patterns in Images," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 11, pp. 1090-1095, 1992.
- [6] Bennett, N., R. Burrige and N. Saito, "A Method to Detect and Characterize Ellipses Using the Hough Transform," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 7, pp. 652-657, 1999.