A COMPARATIVE STUDY OF HOUGH TRANSFORM METHODS FOR CIRCLE FINDING

H.K. Yuen, J. Princen, J. Illingworth and J. Kittler

Department of Electronics and Electrical Engineering
University of Surrey, Guildford, GU2 5XH. U.K.

The objective of this paper is to investigate a number of circle detection methods which are based on variations of the Hough Transform. The methods considered include the standard Hough Transform, the Fast Hough Transform of Li et al, two space saving approaches which are based on those devised by Gerig and Klein and a two-stage method. We experimentally compare the performance of the methods and illustrate properties such as accuracy, reliability, computational efficiency and storage requirements.

In recent years, several methods of circle finding based on the Hough Transform (HT) have been proposed [1,2] as well as some general techniques for fast implementation of the HT [3,4]. Invariably these methods claim to improve efficiency, storage or reliability though in most cases the comparisons made with other techniques are superficial. We feel that this is about the right time to put a number of these algorithms together and examine their properties in more detail. The study is experimental and we consider both real and synthetic images. Our results show that more sophisticated variations of the HT method do not necessarily out-perform straightforward approaches.

The paper is organised as follows. In the next section we introduce the circle finding problem and the basic idea underlying the HT. This is followed by a brief description of each of the five HT based methods considered in our study. The experimental evaluation of each method is then given and the final section presents the conclusions of our work.

CIRCLE FINDING USING THE HT

If a circle in the image is described as

$$(x-a)^2 + (y-b)^2 = r^2, \qquad (1)$$

where (a, b) are the coordinate of the circle center and r is its radius, then an arbitrary edge point (x_i, y_i) will be transformed into a right circular cone in the (a, b, r) parameter space [5]. If all the image points lie on a circle then the cones will intersect at a single point in (a, b, r) corresponding to the parameters of the circle. Kimme et al [6] give probably the first known application of the Hough Transform to detecting circles in real images. In their work, they have made use of the direction of the gradient at each edge point. The centre of a circle must lie on the normal at the edge point. As a result instead of incrementing the whole circular cone, only segments of the cone need be incremented. The size the region which is incremented depends on the accuracy of the edge direction estimation.

An important part of the complete HT process is peak detection. An extremely useful technique which we have found eases the peak finding problem considerably is the post-processing method proposed by Gerig and Klein [1]. It consists of a second daa pass which takes each edge point and identifies the maximum value in the accumulator array out of all parameter values voted for by the point. The edge point is labelled with this location. In all the methods considered, this technique is used to detect the final peaks. We refer the reader to [7,10] for details.

THE STANDARD HT

The Standard Hough Transform (SHT) in this study follows the basic idea outlined in the previous section. A 3-D accumulator array is employed and edge direction information is used to limit voting to a section of the cone. In an ideal situation, the centre of the circle must lie on a line oriented normal to the edge direction. Therefore we only have to move along the normal of every edge point to find the possible locations of centers. The distance between each edge point and the estimated center is a candidate for radius of the corresponding circle. However, in practice, the edge direction is usually estimated inaccurately. As a result, the detection of the true local maximum in the accumulator array could be difficult if this simple accumulation strategy is used. If the direction error is known to be within a range of $\pm\triangle\phi$, then we may say that the center of the circle for the point (x_i, y_i) is within a certain region. This region diverges as the radius increases. To increment this region exactly in the accumulator is very difficult. It can be incremented approximately by incrementing blocks of cells along the normal line, see [10] for details.

THE GERIG AND KLEIN HT

One of the problems with the standard Hough Transform is the storage space required if the range of circle

AVC 1989 doi:10.5244/C.3.29

radii is large. Instead of using a three dimensional accumulator array for the Hough space, Gerig and Klein [1] use 3 different two dimensional accumulator arrays. Here, we call the method Gerig and Klein Hough Transform (GKHT). Two of the arrays are used for center location and one for the corresponding radius. In this case, the storage is significantly smaller than that required for the SHT. The structure of the algorithm for detecting circles is very simple. Since the locus of parameters forms a right circular cones in the Hough space, then given a value for the radius as $r = r_o$, the locus of (a, b) is just a circle of radius r_o in the (a, b) space. Therefore it is sufficient to increment all the boundary points of such a circle for each value of radius. The method proposed by Gerig and Klein does not use edge direction. There is a significant saving in storage space for this procedure. However, because only a single radius is retained for each centre it is not possible to detect concentric circles. Application of the method can be found in [1].

THE GKHT WITH EDGE DIRECTION

As the GKHT method does not use edge direction, a complete circle has to be incremented at every value of radius for each edge point. This is rather computationally demanding. In order to improve the efficiency, we have modified the GKHT to incorporate edge direction information. The modified approach is called Gerig Hough Transform with gradient (GHTG). Only a sector of the circle dependent on the edge direction need to be incremented. In order to do this efficiently, we have approximated the sector by a square [10]. The structure of the method is identical to the GKHT except that in the accumulation and post-processing only the smaller region of the complete circle is considered.

THE 2-1 HOUGH TRANSFORM

Another way to reduce the storage requirement when edge direction is available is to decompose the circle finding problem into two stages. This approach has been used in [2,3]. Here, we call it the 2-1 Hough Transform (21HT). Since the centre of a circle must lie on the normal of each point on the circle, the common intersection point of these normals is actually the centre of the circle. A two dimensional array is required to accumulate votes along the normal of each edge point. To identify the radius of circles, the distance of each point from a candidate centre is calculated and a radius histogram is produced. The detection of false peaks in the centre finding stage can lead to significant computational cost for the second stage, especially if a low threshold is used to detect small circles. The storage space required for the method is quite small, since only a single 2-D accumulator and a 1-D histogram are necessary. Application of this method can be found in [2].

THE FAST HOUGH TRANSFORM

Li et al [4] suggest using a multi-dimensional quadtree structure for accumulating the HT method. The method is called the Fast Hough Transform (FHT). Li has suggested that the FHT should require less storage and be more computationally efficient than the SHT. The FHT is based on the use of a hyperplane formulation, i.e the voting surface in parameter space is a hyperplane. It is possible to develop a hyperplane formulation for the circle finding problem [8]. However from our experimental results [10] this approach is not very suitable for finding circles due to the problem of non-linearity among parameters in the hyperplane formulation. In order to avoid this problem and increase the efficiency of the FHT, we use a different formulation which incorporates edge direction information. The modified approach is called the modified Fast Hough Transform (MFHT). The method makes use of the fact that the locus of the parameters (a, b, r) at an edge point (x_i, y_i) with normal direction ϕ_i in the three dimensional Hough space consists of two orthogonal straight lines pointing outward from the point $(x_i, y_i, 0)$. To determine whether a hypercube has been intersected by one of these two lines, we compare the perpendicular distance from the center of the hypercube to the lines with the diagonal length of the hypercube. If the former is shorter, the hypercube will receive one vote. Unlike the original FHT, there is no incremental updating for the intersection test.

In order to improve the efficiency of the algorithm, we have developed a scheme to choose a suitable threshold value adaptively based on the range of radius being searched and at the same time, reduce the searching range of radius [10]. The results presented are based on this strategy.

EXPERIMENTAL COMPARISON

There are many criteria which can be considered in any comparison of algorithms but in our study the most important points relate to the accuracy, robustness, computational complexity and storage. The accuracy is measured by comparing the absolute errors of the estimated radii and center coordinates to the true values of circles in synthetic images. We present a typical example using an image consisting of 19 randomly generated circles. To test the detection capabilities of each algorithm, a real image consisting of about 76 circles of various radii, counted subjectively, is examined. The number of missing and false circles are counted for each algorithm. To estimate computational efficiency we use the time taken to run each algorithm on our μVAX-2 computer. We realise that as a measure of efficiency this is not necessarily of general significance. However all algorithms, except the Fast Hough Transform, are very similar and therefore at least in a coarse sense we would expect the conclusions regarding efficiency to hold. The major time consumption for each algorithm occurs for transform accumulation and the implementation of the Gerig and Klein post-processing. The 2 stage method also expends a significant time for the radius histogram step. The storage requirements for each method are dominated by the accumulator array(s). Note that all the algorithms were coded in PASCAL.

Figures 1 and 2 show the edges of the synthetic and real images. The edge points in each image are identified using a method based on the Canny edge detector [9] followed by binary thinning of the resultant thresholded edge map. The edge direction of the synthetic image is smeared by a Uniform distributed noise in the range $[-5, 5]$. This smearing does not affect the performance of the GKHT method as edge direction is not used. For the other methods, three different error bounds, $0°, 5°$ and 10°, are assumed for edge direction accuracy. Figure 3 shows the corresponding means of the absolute error of the three parameters. It is interesting to note that the GKHT method achieved a zero error for the radius parameter without using the gradient. The 21HT method is very sensitive to the assumption of value of the error bound. It achieves very small errors for the parameters at the error bound of 5°. The result for the MFHT at the error bound of 0° is very good but the result at the 10° is not available due to the large storage requirement. The means of absolute error of the SHT method are generally larger than those from other methods. The result of the GHTG is good in most cases and the means remain stable as the assumption of the error bound increases. It is difficult to choose the best method based on this single example. However, based on the results from other experiments [10], the 21HT and GHTG methods seem to be more accurate than the others.

Table 1 shows for the synthetic image the number of missing and false circles found by each algorithm at different assumed edge direction error bounds. The best performance in this case is obtained with the GHTG method which only missed 2 and 1 circles at the error bounds of 0° and 5° respectively and detected all the 19 circles using the error bound of 10°, see figure 4.

We found that using the 2 stage method, errors in the first stage caused problems in the second stage histogram. It can be shown that if the magnitude of the centre error is greater than the histogram cell width a single circle will produce two peaks in the radial histogram [10]. The peaks are symmetrically located around the true radius and the distance between them depends on the centre error. This makes concentric circles difficult to distinguish from double peaks due to centre errors.

The most striking observation concerning the MFHT is that it exhausts the available storage of the VAX machine in the case of 10° error bound and therefore the result is not available. This problem relates to the large number of phantom peaks which the MFHT investigates before discovering the correct ones. The algorithm performs reasonably well in terms of efficiency, accuracy and reliability using the other error bounds.

Figure 5 shows the CPU time for the algorithms. In this case, the performance of the 21HT, GHTG and SHT are very close to each other. As expected the GKHT which does not use gradient direction takes significantly longer than GHTG.

The comparative study would not be complete without applying the algorithms to real images. Figure 6 gives the running time of the algorithms on the real image shown in figure 2. The error bound is assumed to be ±5° for the MFHT method (because of storage problem mentioned previously) and ±10° for the 21HT, GHTG and SHT methods. The time taken by the algorithms 21HT, GHTG and SHT are very close to each other. However on counting the missing and false circles, as shown in table 2, the GHTG out-performs all the other methods with only 3 circles missing out of 76. There are, however, 10 false circles detected by the algorithm. All of them are very small circles with radius 1 or 2. These false circles are found from badly detected edge points which are in fact true circles with very low grey level in the original image. Figure 7 shows the result of circle finding using the GHTG method.

In most methods, the storage requirements depend directly on the parameter ranges and the quantization of each parameter axis. We have used 256^2 images and only detect circle centres which lie inside the image, hence a and b are within $(0, 256)$. The radius was limited to lie between $(1, 35)$. Each axis was divided into 1 unit cells. Considering the storage requirements of each of the 5 algorithms, as shown in table 3 we see that the 21HT method has a clear advantage over the others. The GKHT and GHTG are the second best. The storage requirement of the MFHT is rather unpredictable. It will depend on the complexity of the image, choice of threshold value and the error bound assumption.

CONCLUSIONS

The results of our study indicate that both the GKHT and MFHT experience severe difficulties if applied to complex images. The main problem of the GKHT is the unreliability and low efficiency due to the fact that edge direction information is not incorporated in the method. The MFHT suffers from the unpredictable storage and computational requirement, and has the most complicated programming structure in comparison with the other methods. Nevertheless, both methods may still be useful if they are applied to simple images with very low noise.

It has been shown that the performance of the 21HT, GHTG and SHT are very close to each other. The main drawback for the SHT is the large storage requirement for images consisting of circles of different sizes. Although the GHTG is restricted to non-concentric circles, we have found, from other experimental results [10], that the GHTG generally peforms better than the 21HT method in the sense of robustness. This is due to the fact that the 21HT is a 2 stage method and any error occurring in the first stage of center finding will cause difficulties in the peak finding of the second radius finding stage.

ACKNOWLEDGEMENTS

This work was carried out as part of ALVEY contract MMI/078, a collaborative project involving Surrey University, Heriot-Watt University and Computer Recognition Systems of Wokingham.

REFERENCES

1. **Gerig G. and Klein F.** "Fast contour identification through efficient HoughTransform and simplified interpretation strategy," 8th IJCPR, Paris, pp 498-500, 1986.

2. **Davies E.R.** "A modified Hough scheme for general circle location," Pattern Recognition Letters, vol 7, no. 1, pp 37-44, 1988.

3. **Illingworth J. and Kittler J.** "The adaptive Hough Transfprm," IEEE Trans. Pattern Analysis & Machine Intelligence, vol 9, no. 5, pp 690-697, 1987.

4. **Li H., Lavin M.A. and LeMaster R.J.** "Fast Hough Transform: A hierarchical approach," CVGIP, 36, pp 139-161, 1986.

5. **Duda R.O. and Hart P.E.** "Use of the Hough Transformation to detect lines and curves in pictures," Comm. of the ACM, vol 15, no. 1, pp 11-15, 1972.

6. **Kimme C., Ballard D. and Sklansky J.** "Find circles by an array of accumulators," Comm. of the ACM, vol 18, no. 2, pp 120-122, 1975.

7. **Princen J., Yuen H.K., Illingworth J. and Kittler J.** "A comparative study of Hough Transform Algorithms: Part I - Line detection methods," Department of Electronic and Electrical Engineering, University of Surrey, U.K.

8. **Illingworth J., Kittler J. and Princen J.** "Shape detection using the Adaptive Hough Transform," in Proceedings, NATO Advanced Research Workshop on Real-Time Object and Environment Measurement and Classification, Maratea, Italy, September 1987, Springer-Verlag, New York/Berlin.

9. **Canny J.** "A computational approach to edge detection," IEEE Trans. Pattern Analysis & Machine Intelligence, vol 8, no. 6, pp 679-698, 1986.

10. **Princen J., Yuen H.K., Illingworth J. and Kittler J.** "A comparative study of Hough Transform Algorithms: Part II - Circle detection methods," Department of Electronic and Electrical Engineering, University of Surrey, U.K.

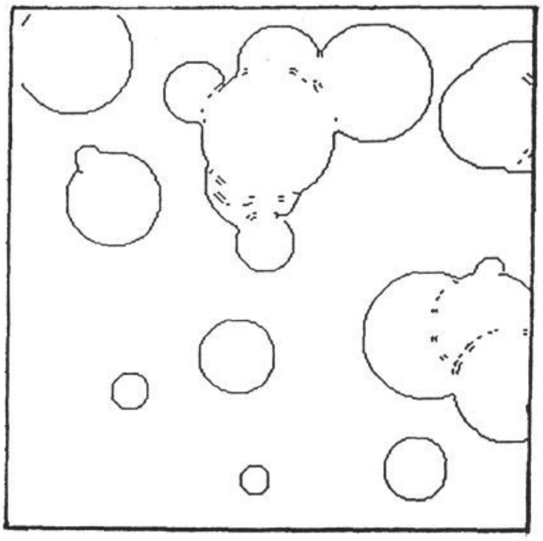

Figure 1. The edge image of the synthetic image.

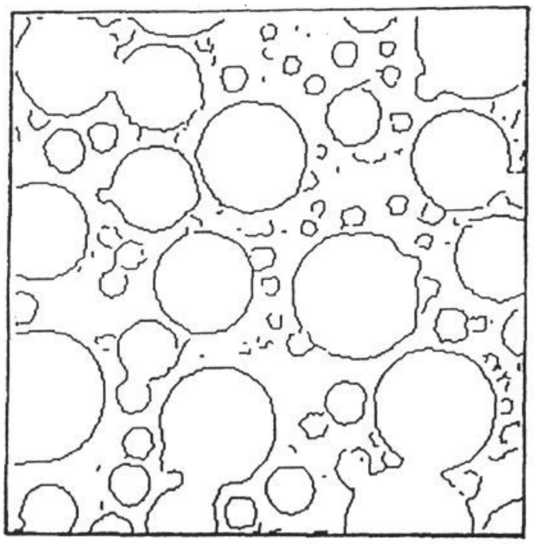

Figure 2. The edge image of the real image.

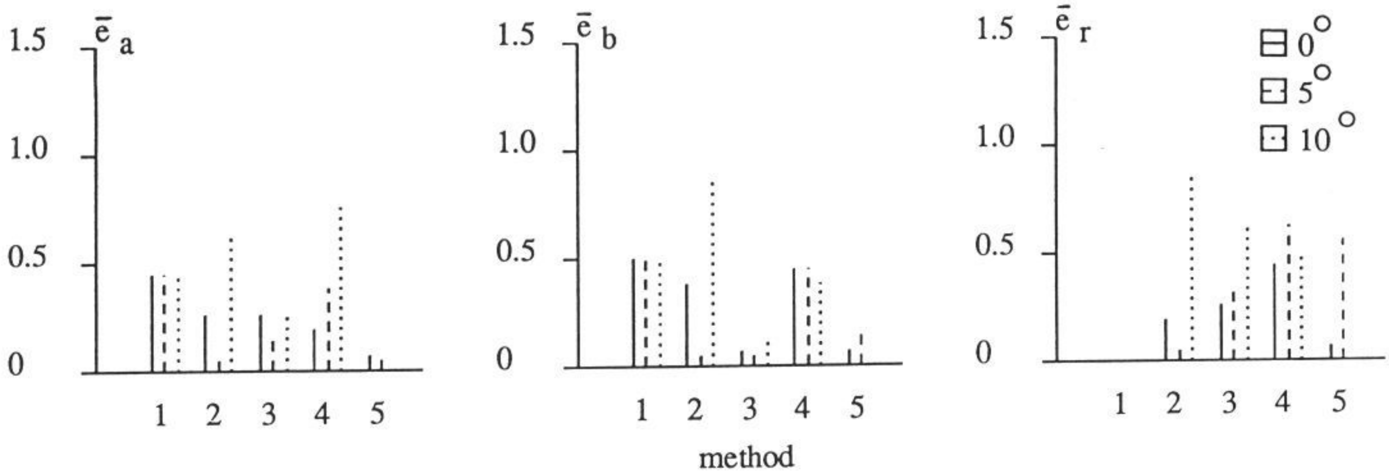

Figure 3. The means of the absolute error, $\bar{e}_a$, $\bar{e}_b$ and $\bar{e}_r$, of the parameters a, b and r respectively for method (1) GKHT, (2) 21HT, (3) GHTG, (4) SHT and (5) MFHT at error bound of (a) 0°, (b) 5° and (c) 10°.

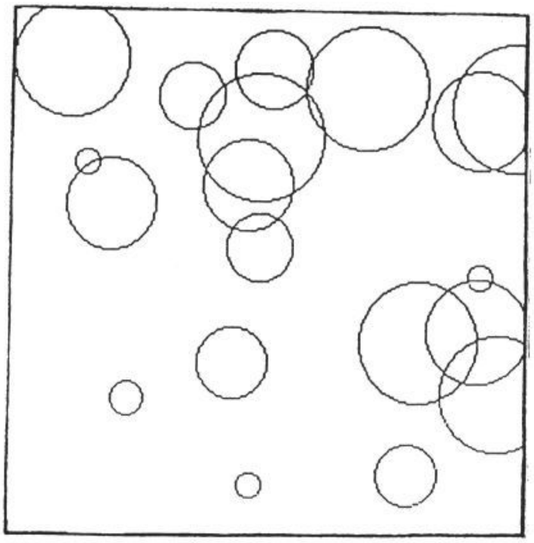

Figure 4. The resultant circles of the synthetic image detected by the GHTG method at the error bound of 10°.

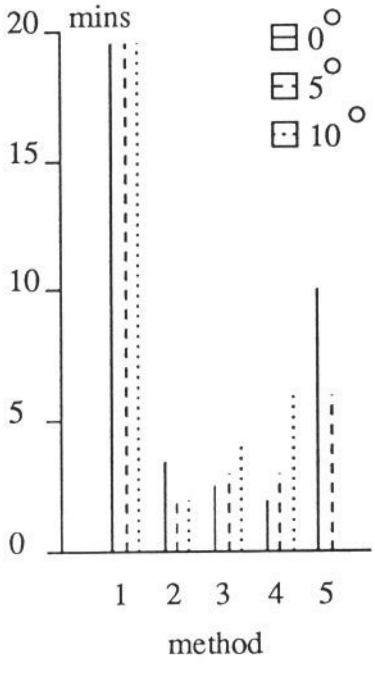

Figure 5. The CPU time of each algorithm on the synthetic image.

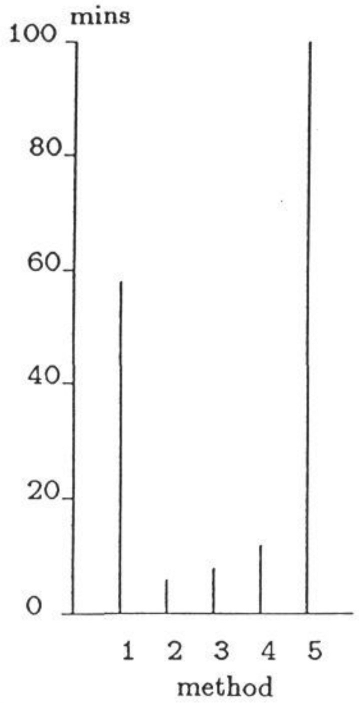

Figure 6. The CPU time of each algorithm on the real image.

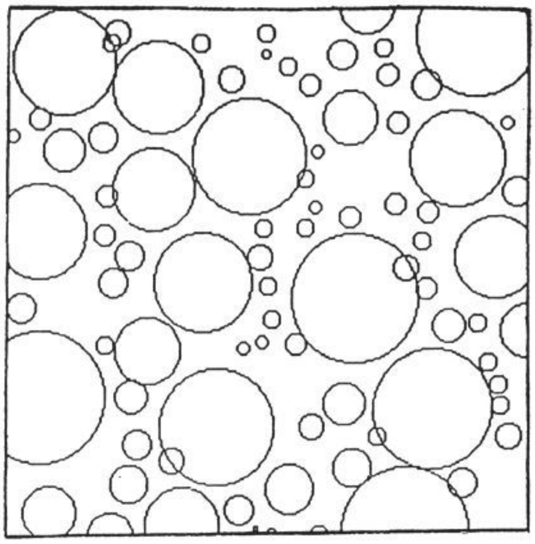

Figure 7. The resultant circles of the real image detected by the GHTG method.

	Missing			False		
Method	0°	5°	10°	0°	5°	10°
GKHT	3	3	3	0	0	0
21HT	6	4	1	0	0	1
GHTG	2	1	0	0	0	0
SHT	4	1	2	0	0	0
MFHT	7	3	-	0	0	-

Table 1. The number of missing and false cirlces in the synthetic image detected by each method at different error bounds.

Method	Missing	False
GKHT	57	16
21HT	41	2
GHTG	3	10
SHT	35	2
MFHT	12	6

Table 2. The number of missing and false circles in the real image detected by by each method.

Method	GKHT	21HT	GHTG	SHT	MFHT
Storage	197K	131K	197K	4588K	-

Table 3. The storage requirement for each algorithm.