Coarse-to-Fine Search Technique to Detect Circles in Images

M. Atiquzzaman

Department of Electrical and Computer Engineering, University of Dayton, Dayton, OH, USA

Detection of patterns in images is an important high-level task in automated manufacturing using machine vision. Straight lines, circles and ellipses are considered to be the basic building blocks of a large number of patterns occurring in real-world images. Real-world images frequently contain noise and occlusions resulting in discontinuous patterns in noisy images. The Hough transform can be used to detect parametric patterns, such as straight lines and circles, embedded in noisy images. The large amount of storage and computing power required by the Hough transform presents a problem in real-time applications.

The aim of this paper is to propose an efficient coarse-to-fine search technique to reduce the storage and computing time in detecting circles in an image. Variable-sized images and accumulator arrays are used to reduce the computing and storage requirements of the Hough transform. The accuracy and the rate of convergence of the parameters at different iterations of the algorithm are presented. The results demonstrate that the coarse-to-fine search strategy is very suitable for detecting circles in real-time environments having time constraints.

Keywords: Circle detection; Coarse-to-fine search; Hough transform; Image processing; Pattern recognition

1. Introduction

Recognition of patterns in images is an important task in many automated manufacturing processes using machine vision [1–3]. A large number of the patterns occurring in real-world manufacturing images can be either represented by a parametric equation or approximated by a number of parametric equations Straight lines, circles and ellipses are considered as the basic building blocks of a large number of patterns in real-world scenes. Most of the other shapes that may be present in real-world images can usually be detected by decomposing them

Correspondence and offprint requests to: Dr Mohammed Atiquzzaman, Department of Electrical and Computer Engineering, The University of Dayton, Dayton, OH 45469-0226, USA. E-mail: matiquzz@ engr.udayton.edu

into a number of straight lines and circles [4,5]. Since the standard Hough transform does not provide the length and the endpoints of a straight line, algorithms for the detection of the length and endpoints were proposed in [6,7].

Circles in real-world images are frequently found to be discontinuous and embedded in noise. The Hough transform [8] has long been known to be an efficient technique for detecting discontinuous patterns embedded in real-world noisy images. The Hough transform (HT) is essentially a voting process where each feature point votes for all the possible circles passing through the point (see Fig. 1).

The votes are accumulated in an accumulator array and the circle receiving the maximum vote is the pattern to be detected. The circle receiving the maximum vote is found from the peak in the accumulator array. Detection of circles in images using the HT has been discussed in [9–14]. A pair of 2D accumulator arrays, to reduce the storage and computation time, in detecting circles has been investigated in [11].

The transform has the drawbacks of being highly computation bound and requiring a large amount of storage. It requires an accumulator array of size $O(p^2)$ where p is the size of the accumulator array. The computational requirement increases with an increase in the size of the accumulator array and the accuracy with which the parameters are to be determined. The resolution of the accumulator array determines the accuracy with which the parameters can be determined.

A considerable amount of research has been devoted to increasing the computational and storage efficiency of the HT. The approaches generally adopted to overcome the computational requirements are mostly based on one of the following techniques.
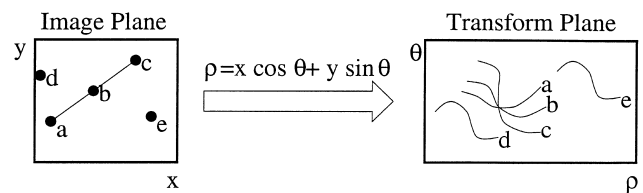

Fig. 1. Mapping of a straight line from the image space to the parameter space in the Hough transform.
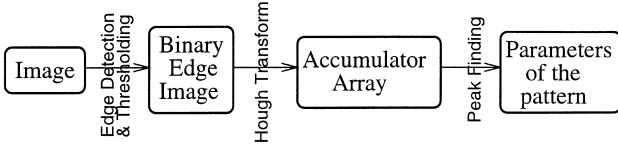
Fig. 2. The different stages in the detection of patterns in images using the Hough transform.

– Using *a priori* information to reduce the dimension of the accumulator array.
– Implementing the transform in multiprocessor systems using a large number of processors.
– Designing efficient algorithms.

A priori information regarding the gradient at the feature points has been used in [15]. The use of gradient information is known to reduce the computing time by one-sixth when compared to the method without using the gradient information. Gerig and Klein [16] illustrate a method to replace the 3D accumulator array required for a circle to three 2D arrays, in order to reduce the space requirements. However, it is not possible to store all the details of concentric circles. Multiprocessor implementations of the transform have been proposed to reduce the execution time of the transform [17–20]. However, multiprocessor systems are expensive and may not be generally available in all cases.

Coarse-to-fine search strategies [21–23], on the other hand, are efficient algorithms suitable for implementation in single processor systems. They use a number of iterations to reduce the computing and storage requirements of the standard HT. The methods are based on dynamically quantised structures for the accumulator arrays.

The objective of this paper is to propose a coarse-to-fine search technique to detect circles in images. The effectiveness of the algorithm will be judged by the accuracy with which the parameters of the circle can be determined. The speed of the algorithm will be measured by the rate of convergence of the estimated parameters towards the actual values of the parameters as a function of the number of iterations.

A necessary step preceding the Hough transform process is edge detection and thresholding, as shown in Fig. 2. The Hough transform is applied to the edge image obtained from the above process. Detection of edges and choice of an optimal threshold have been widely studied in the literature [24]. Since edge detection is not the main focus of this paper, we assume that we have available a binary edge image containing a circular shape. In this paper, we are interested in determining the effectiveness of the proposed algorithm in determining circles, using the Hough transform.

The coarse-to-fine search technique for detecting circles in images using the Hough transform is described in Section 2, followed by results and conclusions in Sections 3 and 4, respectively.

2. The Coarse-to-Fine Approach for Circle Detection

In the proposed coarse-to-fine approach for circle detection, the initial iterations are performed on reduced-size images using a coarse resolution accumulator array. The information obtained from the coarse resolution analysis is used to narrow down the search region of the parameter space. The reduced computational complexity results from the use of a simple peak-detection algorithm in addition to variable-size images and accumulator arrays in the successive iterations of the algorithm. Owing to the use of coarse resolution images and accumulator arrays at the beginning of the iterations, the parameter ranges of investigation are reduced less during the initial iterations, when large-size images and accumulator arrays are used to obtain accurate results. This results in a logarithmic reduction in the parameter range.

Let us define a circle in a binary edge image by

$$(x - a)^2 + (y - b)^2 = r^2$$

where (a, b) are the coordinates of the centre and r is the radius of the circle. The standard Hough transform [8] requires a 3D accumulator array having the ranges of a, b and $\mathbf{r}$ as $0 - \mathscr{S}(x)$, $0 - \mathscr{S}(y)$ and $0 - \frac{1}{2}\sqrt{((\mathscr{S}(x))^2 + (\mathscr{S}(y))^2}$, respectively, where $\mathscr{S}(x)$ and $\mathscr{S}(y)$ are the sizes of the binary image in the x- and y-directions. If the discretisation resolution of the a, b and r axes of the accumulator array are $\Delta(a)$, $\Delta(b)$ and $\Delta(r)$, respectively, then the size of the accumulator array is

$$\frac{\mathscr{S}(x)\,\mathscr{S}(y)\,\sqrt{((\mathscr{S}(x))^2 + (\mathscr{S}(y))^2)}}{2\Delta(a)\,\Delta(b)\,\Delta(r)}$$

The detection accuracy depends on the values of $\Delta(a)$, $\Delta(b)$ and $\Delta(r)$. This results in the accumulator arrays being exorbitantly large if the parameters are to be detected with a reasonable accuracy.

The proposed coarse-to-fine search technique iterates L times, using a different image from a set of L images at each iteration. The set of images is generated by reducing the size of the original image $(L - 1)$ times by a Gaussian subsampling process [25]. The factor by which an image is reduced at each step is σ. The first iteration uses the smallest image from the set and accumulates the votes in a small accumulator array. Because of the use of a small image and a small accumulator array in the first iteration, the values obtained for a, b and r are very rough estimates of the actual values. The second iteration uses a larger image and a larger accumulator array than those used in the first iteration. However, the parameter ranges of investigation are narrowed down during the second iteration. The estimates of a, b and r obtained in the first iteration are used to select reduced ranges of the parameters to be investigated in the second iteration. The above procedure of reducing the parameter ranges and using increasingly larger images and accumulator arrays at successive iterations is carried out until the original image has been analysed. The above coarse-to-fine analysis technique results in a reduced amount of computation, compared to the original HT using a single image and a single accumulation of the HT [21]. Section 2.1 introduces some notation which will be used to illustrate the proposed algorithm for the detection of circles.

2.1 Notation

L = number of iterations in the coarse-to-fine search algorithm.
$\mathscr{S}(x, i)$, $\mathscr{S}(y, i)$ = the x and y sizes of the image after i reductions in size. $\mathscr{S}(x, 0)$ and $\mathscr{S}(y, 0)$ are therefore, the sizes

of the original image. We will assume square images whose sizes will be represented by $\mathscr{S}(f, i) = \mathscr{S}(x, i) = \mathscr{S}(y, i) = \mathscr{S}(r, i)$ for $0 \leq i \leq L - 1$.

$\mathscr{S}(a, i)$, $\mathscr{S}(b, i)$, $\mathscr{S}(r, i)$ = the dimensions of the accumulator array along the a, b and r axes during the $(L - 1)$th iteration.

$\mathscr{S}(f, i)$ = the size of the image f along each dimension at the $(L - i)$th iteration, assuming square images.

$\Delta(a, L - i)$, $\Delta(b, L - i)$, $\Delta(r, L - i)$ = discretisation steps of a, b and r during the ith iteration.

$\mathscr{R}(a, L - i)$ = Range of a during the ith iteration.

$\mathscr{R}(b, L - i)$ = Range of b during the ith iteration.

$\mathscr{R}(r, L - i)$ = Range of r during the ith iteration.

γ = the factor (subsampling number) by which the a, b and r ranges are reduced after the last (Lth) iteration.

σ = the factor by which the image is reduced at successive iterations.

2.2   Reduction of Parameter Ranges

The algorithm uses small image and accumulator arrays during the initial iterations. An image loses detail as the size is reduced successively, and a small accumulator array means lower resolution of the estimated parameters. Consequently, a small image in conjunction with a small accumulator array results in very rough estimates of the parameters during the initial iterations. If the parameter range is reduced much (based on the rough estimates) during the initial iterations, it has been found that the actual parameters frequently fall outside the new reduced range of investigation. This problem can be overcome by reducing the parameter ranges less during the initial iterations as compared to the iterations in the final stages. Higher reductions in the parameter range can be applied when large-size images and accumulator arrays are used in the last few iterations. The above requirements are satisfied by reducing the parameter ranges logarithmically (instead of linearly) at successive iterations.

Figure 3 illustrates the linear and the logarithmic range reductions for $L = 3$. As can be seen from the figure, both the linear and the logarithmic range reductions use the full parameter ranges in the first iteration. However, the logarithmic reduction results in a much larger parameter range to be
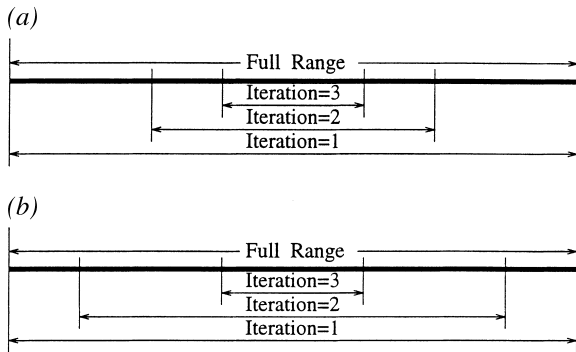
(a)

(b)

Fig. 3. Reduction of parameter ranges. (*a*) Linear reduction. (*b*) Logarithmic reduction.

investigated during the second iteration when compared to the linear reduction which reduces the range by half after the first iteration (and all subsequent iterations). During the third iteration the range of investigation is the same as the range in the linear reduction. This is achieved by reducing the range by four times after the second iteration in the logarithmic method while the linear methods only reduces the range by half after the second iteration. In this example, if the number of interations is more than three, the parameter ranges of investigation resulting from the logarithmic reduction will, in fact, be narrower than the linear reduction method. Alternatively, for a particular range of investigations at the last iteration, the logarithmic reduction of range results in requiring fewer iterations than when the linear reduction is used.

Let the a, b and r ranges be reduced by γ after the Lth iteration. To apply a logarithmic range reduction, the parameter ranges should be reduced by γ / σ^{L-i} after the ith, $1 \leq i \leq L$, iteration. If the a, b and r parameters are to be determined with an accuracy of one pixel after L iterations, γ for any parameter can be obtained from the following equation:

$$\frac{1}{\gamma/\sigma^0} \frac{1}{\gamma/\sigma^1} \frac{1}{\gamma/\sigma^2} \cdots \frac{1}{\gamma/\sigma^{L-1}} = \frac{1}{\mathscr{S}(f, 0)}$$

or,

$$\gamma = {}^{1/L}\sqrt{(\sigma^{L/2(L-1)} \mathscr{S}(f, 0))} \tag{1}$$

Without loss of generality, we will assume that the sizes of an accumulator array along the a, b and r dimensions are the same, and the ranges for a, b, and r are also reduced by the same factor after an iteration. Therefore,

$$\mathscr{R}(a, L-1) = \mathscr{R}(b, L-1) = \mathscr{R}(r, L-1) = \mathscr{S}(f, L-1) \tag{2}$$

After the first iteration, the range of a (b and r) is reduced by γ / σ^{L-1}. Therefore,

$$\mathscr{R}(a, L-2) = \frac{\mathscr{R}(a, L-1)}{\gamma/\sigma^{L-1}} \sigma = \frac{\mathscr{S}(f, L-1)}{\gamma/\sigma^{L-1}} \sigma \tag{3}$$

Similarly, it can be shown that

$$\mathscr{R}(a, L-i) = \mathscr{S}(f, L-1)\sigma^{L-1}\sigma^{L-2} \cdots \sigma^{L-i+1} \left(\frac{\sigma}{\gamma}\right)^{i-1} \tag{4}$$

As an example, using an original image of size 512×512, $L = 4$, and $\sigma = 2$, γ comes out to be 13.454 and $\mathscr{R}(a, 3) = 512$, $\mathscr{R}(a, 2) = 304$, $\mathscr{R}(a, 1) = 90.5$, and $\mathscr{R}(a, 0) = 13.5$. Therefore, the parameter ranges are reduced by 1.68, 3.36, 6.725 and 13.45 after the first, second, third and fourth iterations, respectively. For square accumulator arrays,

$$\mathscr{S}(a, L-i) = \mathscr{S}(b, L-i) = \mathscr{S}(r, L-i) = \frac{\mathscr{S}(f, L-1)}{\sigma^\mu} \sigma^{i-1}$$

where μ is an integer ≥ 1. Higher values of μ indicates a smaller accumulator array being used in the iterations. Therefore,

$$\Delta(a, L-1) = \frac{\mathscr{R}(a, L-i)}{\mathscr{S}(a, L-i)}$$

$$= \frac{\sigma^{L-1}\sigma^{L-2} \cdots \sigma^{L-i+1}}{\gamma^{i-1}} \sigma^\mu \tag{5}$$

For $\mu = 1$, $\sigma = 2$, $L = 4$ and $\mathcal{S}(f, L-1) = 512$ we have

$$\Delta(a, 3) = \Delta(b, 3) = \Delta(r, 3) = 2$$
$$\Delta(a, 2) = \Delta(b, 2) = \Delta(r, 2) = 1.189$$
$$\Delta(a, 1) = \Delta(b, 1) = \Delta(r, 1) = 0.3535$$
$$\Delta(a, 0) = \Delta(b, 0) = \Delta(r, 0) = 0.05255$$

3. Results

In this section, we present the accuracy with which the radius and the coordinates of the centre of a circle can be determined. The detection accuracy will be represented by the percentage error between the estimated (from the proposed algorithm) and the actual values of parameters. To obtain the percentage errors, we use a number of images and average the percentage errors in detecting the circles in those images. The percentage errors of a, b and r will be represented by ϵ_a, ϵ_b and ϵ_c, respectively. We will also show the speed at which the parameters converge to the actual values at the different iterations. The speed of convergence will be demonstrated by the rate at which the search interval for the parameters are reduced at successive iterations, with the peak still being within the range of investigation.

Figure 4 illustrates the 3D parameter space resulting from two points, (10, 10) and (20, 20), on a circle having a radius of 5 and the centre being at (15, 15). Figure 5 shows the binary multiresolution images used at the different iterations. The initial image is of size 512×512 and has a number of discontinuities at the edges. $L = 4$, $\sigma = 2$, and $\mu = 2$ have been used. Therefore, as seen in Fig. 5, the sizes of the images used at the different iterations are 64×64, 128×128, 256×256, and 512×512. The first iteration uses the 64×64 image; the last ones uses 512×512. Because of using $\sigma = 2$ and $\mu = 2$, the sizes of the accumulator arrays were 16×16, 32×32, 64×64, and 128×128 during iterations 1, 2, 3, and 4, respectively.

The constant-r planes of the accumulator arrays (at the different iterations) containing the maximum votes are shown
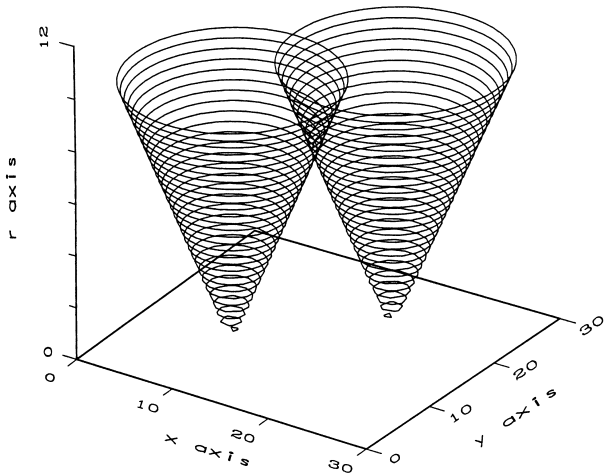

Fig. 4. Illustration of the 3D parameter space resulting from two points, (10, 10) and (20, 20), having a radius of 5 and the centre at (15, 15).

in Fig. 6. To show the sharpness of the peaks, the planes are represented in 3D forms, where the vote count is represented in the z-direction. Since the first iteration uses the full ranges of the parameter, the position of the peak in the plane during the iteration depends on the actual values of the parameters a and b. Parameter values obtained after iteration i, $1 \leq i \leq L - 1$, are used as coarse approximations of the parameter values in iteration $i + 1$. From the coarse approximations, the algorithm selects the starting and ending values of the a, b and r ranges of the accumulator arrays for the next iteration. The ranges are chosen such that the peak is close to the centre, as seen in iterations 2, 3, and 4 in Fig. 6. The peaks are also seen to be very sharp and unique in the different iterations.

Table 1 presents the percentage errors in the parameters, obtained by averaging the errors from analysing 30 different images of size 512×512 each, and using $\sigma = 2$, $\mu = 1$ and 2, and $L = 4$. The percentage error of a parameter is obtained by calculating the error between the estimated value (from the algorithm) of the parameter and the actual value of the parameter. The 30 images contain circles of random radius and centre coordinates. The sharp decrease in the percentage errors in successive iterations shows the speed of convergence of the estimated parameters to the actual parameters during the different iterations. It should be pointed out that when $\mu = 2$ is used, the accumulator arrays used in the different iterations are half the size of those when $\mu = 1$. Consequently, the errors with $\mu = 2$ are larger during the initial iterations than when $\mu = 1$ is used. However, the errors at the fourth iteration are the same for both $\mu = 1$ and 2. Note that the memory required to store the accumulator arrays with $\mu = 2$ is a quarter of the requirement when $\mu = 1$ is used. Moreover, with $\mu = 2$, the computing time is half the time required with $\mu = 1$. Therefore, in this particular case, $\mu = 2$ would be a better choice than $\mu = 1$.

The ranges of the parameters a, b and r computed by the algorithm in the different iterations are shown in Table 2. The data relate to an image of size 512×512 containing a circle of radius 210 and having its centre at (230, 235). $L = 4$, $\sigma = 2$, and $\mu = 2$ have been used. The first iteration uses the full range of the parameters and the range is then logarithmically reduced at successive iterations. In all the iterations, except the first, the algorithm has been successful in selecting the range such that the actual parameter values are almost in the centre of the range.

4. Conclusions

In this paper, we have demonstrated the effectiveness of the coarse-to-fine search Hough technique to detect circles in images. This could be used for automatic inspection of objects in automated manufacturing processes using machine vision. Binary edge images have been used to illustrate the algorithm. It has been shown that the centre and radius can be determined to an accuracy of less than 0.2% in only four iterations. Prior edge gradient information for the feature points can be used to further reduce the amount of computation [15]. The algorithm illustrated in this paper applies to the detection of circular arcs. With a change in the parametric equation, it can be used to detect ellipses and other parameterisable patterns in images.
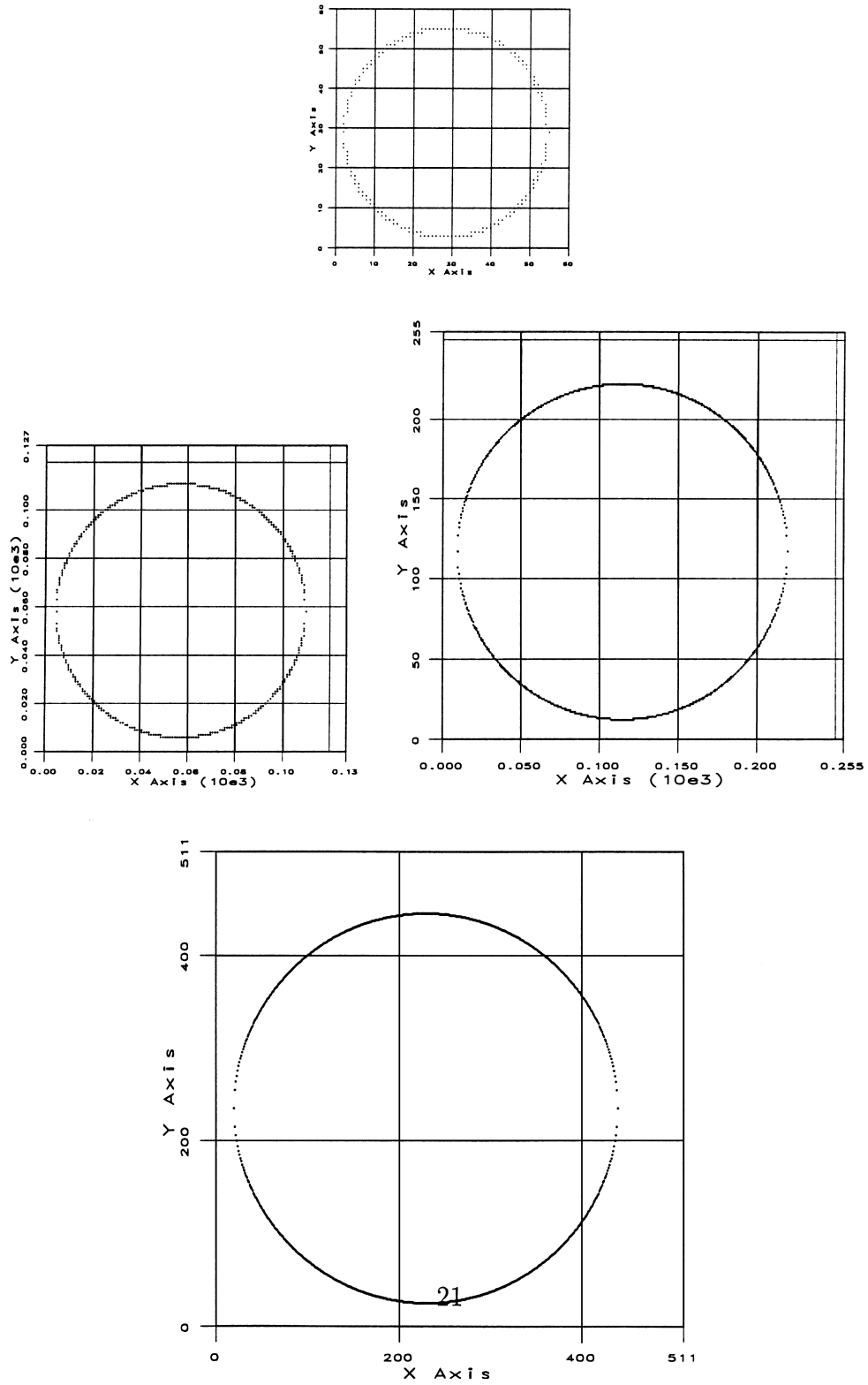
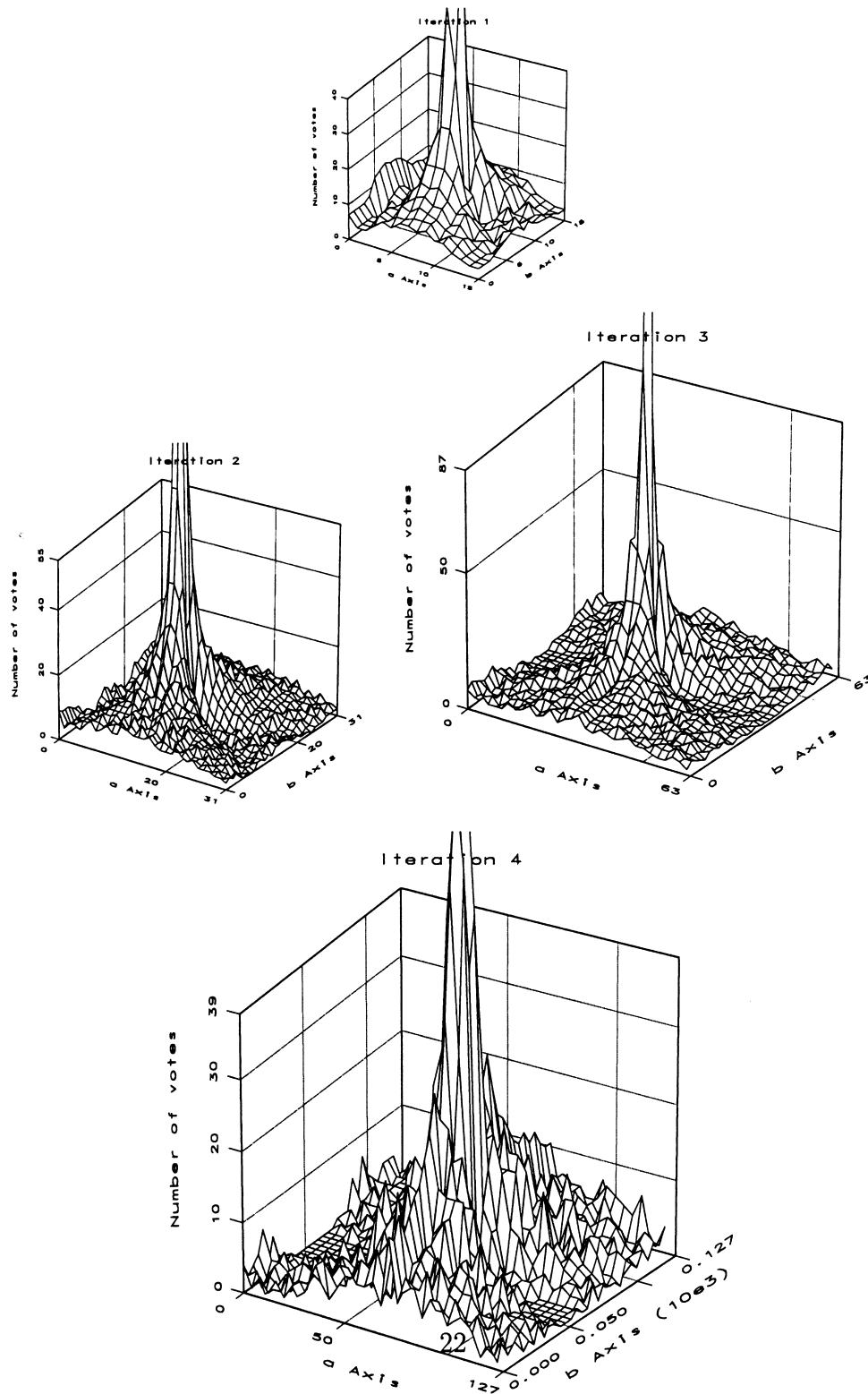**Fig. 5.** Images used during the different iterations.

Fig. 6. Accumulator array planes during the different iterations.

Table 1. Percentage errors of *a*, *b* and *r* at each iteration based on 30 images. $\mathcal{S}(f, 0) = 512$, $\sigma = 2$ and $L = 4$.

Iteration number	ϵ_a		ϵ_b		ϵ_c	
	$\mu = 1$	$\mu = 2$	$\mu = 1$	$\mu = 2$	$\mu = 1$	$\mu = 2$
1	8.52	39.02	15.92	15.92	12.66	42.75
2	2.74	4.13	0.83	3.42	3.87	3.87
3	0.17	0.25	0.17	0.27	0.69	0.85
4	0.15	0.15	0.14	0.14	0.17	0.19

Table 2. The parameter ranges of investigation at the different iterations for a circle having a radius of 210 and centre at (230, 235) in a 512×512 image.

Iteration number	Parameter range of investigation		
	a range	*b* range	*r* range
1	0–512	0–512	0–512
2	71.8–376.2	103.8–408.2	71.8–376.2
3	188.2–278.8	191.7–282.2	169.2–259.7
4	222.5–236.0	227.4–240.9	203.5–217.0

References

1. V. Allada and S. Anand, "Efficient vertex detection algorithms using the hough transform", International Journal of Advanced Manufacturing Technology, 11(6), pp. 394–405, June 1996.
2. D. M. Tsai, "Locating overlapping industrial parts for robotic assembly", International Journal of Advanced Manufacturing Technology, 12(4), pp. 288–302, April 1996.
3. S. J. Huang and C. C. Lin, "Three-dimensional non-contact measurement system", International Journal of Advanced Manufacturing Technology, 13(6), pp. 419–425, June 1997.
4. C. J. Radford, "Vehicle detection in open-world scenes using a Hough transform technique", IEE Third International Conference on Image Processing and its Applications, London, pp. 78–82, July 1989.
5. J. L. C. Sanz, E. B. Hinkle and A. K. Jain, Radon and Projection Transform-Based Computer Vision, Springer-Verlag, Berlin, 1988.
6. M. W. Akhtar and M. Atiquzzaman, "Determination of line length using Hough transform", Electronics Letters, 28(1), pp. 94–96, January 2, 1992.
7. M. Atiquzzaman and M. W. Akhtar, "Complete line segment description using the Hough transform", Image and Vision Computing, 12(5), pp. 267–273, June 1994.
8. P. V. C. Hough, "Methods and means for recognizing complex patterns", US Patent 3069654, 1962.
9. C. Kimme, D. Ballard and J. Sklansky, "Finding circles by an array of accumulators", Communications of the ACM, 18(2), pp. 120–122, February 1975.
10. H. K. Yuen, J. Princen, J. Illingworth and J. Kittler, "Comparative study of Hough transform methods for circle finding", Image and Vision Computing, 8(1), pp.71–77, February 1990.
11. R. Chan and W. C. Siu, "New parallel Hough transform for circles", IEE Proceedings Part-E, 138(5), pp. 335–344, September 1991.
12. K. Hanahara and M. Hiyane, "A circle detection algorithm simulating wave propagation", Machine Vision and Applications, 3, pp. 97–111, 1990.
13. P. Kierkegaard, "A method for detection of circular arcs based on the Hough transform", Machine Vision and Applications, 5, pp. 249–263, 1992.
14. V. F. Leavers, "The dynamic generalized Hough transform: Its relationship to the probabilistic Hough transforms and an application to the concurrent detection of circles and ellipses", Computer Vision, Graphics and Image Processing: Image Understanding, 56(3), pp. 381–398, November 1992.
15. F. O'Gorman and M. B. Clowes, "Finding picture edges through collinearity of feature points", IEEE Transactions on Computers, C-25(4), pp. 449–456, April 1976.
16. G. Gerig and F. Klein, "Fast contour identification through efficient Hough transform and simplified interpretation strategy", 8th International Joint Conference on Pattern Recognition, pp. 498–500, 1986.
17. C. Guerra and S. Hambrusch, "Parallel algorithms for line detection on a mesh", Journal of Parallel and Distributed Computing, 6, pp. 1–19, 1989.
18. M. Atiquzzaman, "Pipelined implementation of the multiresolution Hough transform in a pyramid multiprocessor", Pattern Recognition Letters, 15(9), pp. 841–851, September 1994.
19. A. L. Fisher and P. T. Highnam, "Computing the Hough transform on a scan line array processor", IEEE Transactions on Pattern Analysis and Machine Intelligence, 11(3), pp. 262–265, March 1989.
20. A. Kavianpour, S. Shoari and N. Bagherzadeh, "A new approach for circle detection on multiprocessors", Journal of Parallel and Distributed Computing, 20, pp. 256–260, 1994.
21. M. Atiquzzaman, "Multiresolution Hough transform – an efficient method of detecting pattern in images", IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(11), pp. 1090–1095, November 1992.
22. J. Illingworth and J. Kittler, "Adaptive Hough transform", IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-9(5), pp. 690–698, September 1987.
23. H. Li, M. A. Lavin and R. J. LeMaster, "Fast Hough transform", Computer Vision, Graphics and Image Processing, 36, pp. 139–161, 1986.
24. Y. Lu and R. C. Jain, "Behaviour of edges in scale space", IEEE Transactions on Pattern Analysis and Machine Intelligence, 11(4), pp. 337–356, April 1989.
25. P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code", IEEE Transactions on Communications, COM-31(4), pp. 532–540, April 1983.