A New Fast and Robust Circle Extraction Algorithm

Euijin KIM, Miki HASEYAMA, and Hideo KITAJIMA
School of Engineering Hokkaido University
N-13, W-8, Kita-ku, Sapporo-shi 060-0828, Japan
kim@media.eng.hokudai.ac.jp

Abstract

This paper presents a new algorithm that is capable of extracting circles from complicated and heavily corrupted images. The algorithm uses a least-squares fitting algorithm for arc segments. The arcs are segmented by using the short straight lines which are extracted by a fast line extraction algorithm. The arc segments are used to yield accurate circle parameters. Tests performed on synthetic and real-world images show that the algorithm quickly and accurately extracts circles from complicated and heavily corrupted images.

1 Introduction

Extracting circles from digital images has received much attention for several decades in computer vision. Therefore, many algorithms have been presented. The Hough transform (HT) is the best known algorithm. In spite of its popularity owing to its simple theory of operation, the HT has some problems. We point out, for instance, the computational burden in the 3-dimensional parameter space. To solve these problems, several modified algorithms have been proposed based on the HT [1, 2, 3, 4, 5]. However, the improvements still have some problems such as the choice of thresholds, finding peaks in the parameter space, interference from abundant edge pixels, and poor tangents and gradients of the edge pixels.

On the other hand, several algorithms which do not use histograms in the parameter space have been presented such as a least-squares fitting algorithm [6], an algorithm using geometrical symmetry properties [7], circle extraction using a genetic algorithm [8] and the UpWrite [9]. These non-HT-based algorithms may extract circles faster that the HT-based algorithms because they do not use histograms in the parameter space.

All the above mentioned algorithms process the image pixel by pixel. Thus, they sometimes fail in the following situations:

- The circular objects are embedded in a complicated background.
- The input image has many extraneous edge pixels.
- The input image is heavily corrupted.

These situations interfere with the construction of a robust histogram in the parameter space as well as the pairing of edge pixels used for the estimation of circles for HT-based and non-HT-based algorithms.

In the work presented here, a new circle extraction method is proposed which does not need a parameter space. It fits short straight lines to a circle by least-squares. The short straight lines are determined by using a fast line extraction algorithm [10]. This is possible because circles are represented as short straight lines in digital images. Also, the line-fitting approach can be used to improve the accuracy and speed of the algorithm in complicated and heavily corrupted images.

2 Algorithm description

2.1 The least-squares fitting algorithm

The least-squares fitting algorithm is one of the most commonly used methods for finding the circle parameters $(\tilde{R}, \tilde{x}, \tilde{y})$. We review this algorithm given in reference [6]. Given a set of pixels $(x_1, y_1), \ldots, (x_N, y_N)$ which represents a contour that is assumed to belong to a circular arc, we use the least-squares fitting algorithm to approximate the curved segment defined by the N pixels. The arc center $(\tilde{x}, \tilde{y})$ and radius $\tilde{R}$ are estimated by minimizing the sum of the squared errors between the radius and distances from the pixels to the center.

$$min \quad e(\tilde{R}, \tilde{x}, \tilde{y}) = \sum_{i=1}^{N} [R^2 - \{(x_i - \tilde{x})^2 + (y_i - \tilde{y})^2\}]^2$$

Setting to zero the partial derivatives of $e(\tilde{R}, \tilde{x}, \tilde{y})$ with respect to $\tilde{R}, \tilde{x}$ and $\tilde{y}$, we obtain [6]

$$\tilde{x} = \frac{c_1 b_2 - c_2 b_1}{a_1 b_2 - a_2 b_1} \tag{1}$$

$$\tilde{y} = \frac{a_1 c_2 - a_2 c_1}{a_1 b_2 - a_2 b_1} \tag{2}$$

$$\tilde{R}^2 = \frac{1}{N}\{\sum_{i=1}^{N} x_i - 2\sum_{i=1}^{N} x_i\tilde{x} + N\tilde{x}^2 + \sum_{i=1}^{N} {y_i}^2 - 2\sum_{i=1}^{N} y_i\tilde{y} + N\tilde{y}^2\} \quad (3)$$

where

$$\begin{aligned}
a_1 &= 2(\sum_{i=1}^{N} x_i \sum_{i=1}^{N} x_i - N\sum_{i=1}^{N} {x_i}^2) \\
a_2 &= 2(\sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i - N\sum_{i=1}^{N} x_i y_i) \\
b_1 &= a_2 \\
b_2 &= 2(\sum_{i=1}^{N} y_i \sum_{i=1}^{N} y_i - N\sum_{i=1}^{N} {y_i}^2) \\
c_1 &= (\sum_{i=1}^{N} {x_i}^2 \sum_{i=1}^{N} x_i - N\sum_{i=1}^{N} {x_i}^3 + \sum_{i=1}^{N} x_i \sum_{i=1}^{N} {y_i}^2 \\
&- N\sum_{i=1}^{N} x_i {y_i}^2) \\
c_2 &= (\sum_{i=1}^{N} {x_i}^2 \sum_{i=1}^{N} y_i - N\sum_{i=1}^{N} {y_i}^3 + \sum_{i=1}^{N} y_i \sum_{i=1}^{N} {y_i}^2 \\
&- N\sum_{i=1}^{N} {x_i}^2 y_i)
\end{aligned}$$

The disadvantages of this least-squares fitting algorithm are well known. For example, it is sensitive when outliers are present with a moderate level of noise such as extraneous edge pixels. Therefore we cannot adopt this method without some improvements on this least-squares fitting algorithm for complicated and heavily corrupted images. To avoid these limitations we use short straight lines which are extracted by using a fast line extraction algorithm [10].

2.2 The fast line extraction (FLE) algorithm

We explain that the major difference between the proposed algorithm and other algorithms. Most conventional algorithms compute tangents and gradients for each pixel. The proposed algorithm uses, in contrast, the short straight lines which approximate a circle. We have proposed a fast line extraction algorithm (FLE) [10] using the directions of line segments. The FLE can extract short straight lines from a circular contour with high accuracy and speed. It can classify the short straight lines of the circle into groups I,II,III,IV as shown in Fig. 1. Fig. 2 shows a circle separated into segments belonging to the four groups. Each segment consists of two lines as illustrated by the solid lines in the figure. Thus, there are 16 short straight lines present in the circle.

We use this classification in order to find the circle parameters and reduce computational burden. Further information on the FLE is found in reference [10]. Simultaneously, we also can easily obtain the mid point $(cx = (x_1+x_2)/2, cy = (y_1 + y_2)/2)$ and tangent (θ_{line}) from the extremity points $(x_1, y_1), (x_2, y_2)$ of each of the extracted short straight lines by the following equation:

$$\theta_{line} = \tan^{-1}(\frac{y_1 - y_2}{x_1 - x_2}). \quad (4)$$

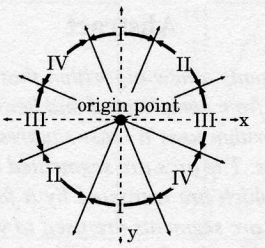


Figure 1: *The assigned four groups of classification.*

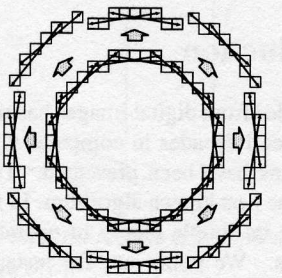


Figure 2: *The 16 extracted short straight lines of a circle.*

2.3 Arc segmenting

Each arc is segmented using the extracted short straight lines that have been assigned to one of the four groups. The approximate circle parameters $(\tilde{R}, \tilde{x}, \tilde{y})$ and the tangent (θ_{line}) can be estimated by equations (1), (2), (3) and (4) for each individual short straight line. Each of the four groups can be further subdivided into two arcs, one facing the other across from the estimated center point$(\tilde{x}, \tilde{y})$.

For example, Fig. 3 shows the two subdivided groups for group I and III lines by using the estimated center points. Each subdivided group for the four groups is shown in Fig. 4. The range of each subdivided group is $45°$. This is very important because by classifying the short straight lines into one of the eight groups, it is easy to segment the arcs in the same group. We record the short straight lines, which are

extracted from one of the four groups, to form a set $\mathbf{LS_1}$ $= \{Line_i, i = 1, 2, \ldots\}$. Simultaneously, $(\tilde{R}, \tilde{x}, \tilde{y})$, θ_{line} and a subdivided group as shown in Fig. 3, are estimated through each $Line_i$.

2.3.1 Generating a pair $(Line_i, Line_j)$

It is not sufficent to estimate a part of a circle as a single $Line_i$. We pair two lines in one of the four groups if the pair $(Line_i, Line_j)$ satisfies the following three constraints:

1. The distance between the two mid points (cx_i, cy_i) and (cx_j, cy_j) of the pair $(Line_i, Line_j)$ can not exceed a maximum distance $(D1)$.

2. The interior angle (θ_{in}) between the pair $(Line_i, Line_j)$ have to be within a limited range. The interior angle (θ_{in}) is obtained from an inner-product of the two vectors between the intersection point and the each mid point (cx_i, cy_i) and (cx_j, cy_j) of the two short straight lines as shown in Fig. 5. The permissible value of θ_{in} is determined by $135° < \theta_{in} < 180°$ because the range is limited to $45°$ for each of the eight groups.

3. If the pair $(Line_i, Line_j)$ satisfies the two constraints mentioned above, we start to estimate the combined circle parameters $(\tilde{R}, \tilde{x}, \tilde{y})$ by equations (1), (2) and (3). To confirm the accuracy of estimated parameters, we use the difference between the two tangents (θ_{line}) and the two estimated tangents (θ_{est}) which are calculated for the two mid points (cx_i, cy_i) and (cx_j, cy_j) of the line pair by the following equation:

$$\theta_{est(k)} = \tan^{-1}\left(\frac{cx_k - \tilde{x}}{\tilde{y} - cy_k}\right), (k = i, j). \quad (5)$$

The pair $(Line_i, Line_j)$ can be merged according to the relationship between the permissible error θ_{er} and the difference by the following equation:

$$|\theta_{est(k)} - \theta_{line(k)}| < \theta_{er}, (k = i, j). \quad (6)$$

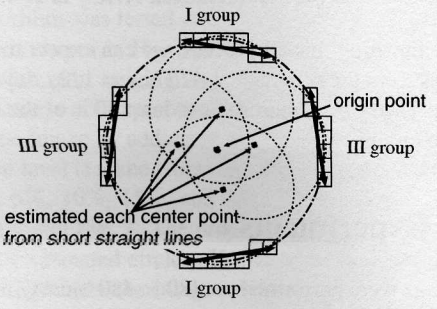
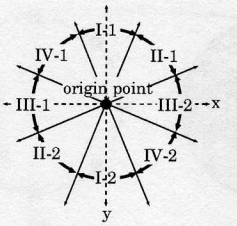


Figure 3: *Two subdivided groups for each group I and III*



1 : I-1
2 : II-1
3 : III-2
4 : IV-2
5 : I-2
6 : II-2
7 : III-1
8 : IV-1

Figure 4: *The eight groups of classification from four groups.*

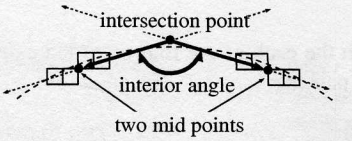


Figure 5: *The interior angle between two lines.*

The pair $(Line_i, Line_j)$ which passes this test, is used in a combinatorial calculation for $Line_i$ in $\mathbf{LS_1}$. We have to update the new tangent (θ_{arc}) and new mid point $(cx(arc)_i, cy(arc)_j)$ of the estimated arc_i. We also record the two points $(cx_{1i}, cy_{1i}) = (cx_i, cy_i)$, $(cx_{2i}, cy_{2i}) = (cx_j, cy_j)$ of the pair as two extremity points of the arc. The results which are combined with the pairs facilitate a subsequent procedure and move into a set $\mathbf{LS_2} = \{arc_i = (Line_i, Line_j), i = 1, 2, \ldots\}$.

2.3.2 Arc merging

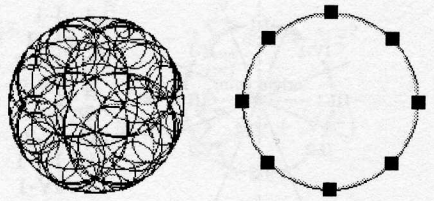
We make arc segments using a pair (arc_i, arc_j) from the set $\mathbf{LS_2}$ with the following three constraints. The three constraints are similar with the previous three constraints.

1. The distance between the two mid points $(cx(arc)_i, cy(arc)_i)$ and $(cx(arc)_j, cy(arc)_j)$ of the pair (arc_i, arc_j) can not exceed a maximum distance $(D2)$.

2. The interior angle (θ_{in}) between the pair with mid points $(cx(arc)_i, cy(arc)_i)$, $(cx(arc)_j, cy(arc)_j)$ have to be within a limited range. The range is determined by $135° < \theta_{in} < 180°$.

3. If all these conditions are satisfied, the algorithm estimates the combined circle parameters $(\tilde{R}, \tilde{x}, \tilde{y})$. As described previously, we can calculate θ_{est} with the following equation:

$$\theta_{est(k)} = \tan^{-1}\left(\frac{cx_k - \tilde{x}}{\tilde{y} - cy_k}\right), (k = 1i, 2i, 1j, 2j) \quad (7)$$



(a) (b)

Figure 6: *The extracted short straight lines from a circle and 8 arc segments classifying them into eight groups.*

with the each of the four recorded extremity points of the pair (arc_i, arc_j). The equation:

$$|\theta_{est(k)} - \theta_{line(k)}| < \theta_{er}, (k = 1i, 2i, 1j, 2j) \quad (8)$$

can be used to confirm the accuracy of the estimated parameters.

Then the merged circle parameters $(\tilde{R}, \tilde{x}, \tilde{y})$, the new tangent, the new mid point, and the two new extremity points for the pair (arc_i, arc_j) are recorded into the set $\mathbf{LS_2}$ as a new arc_i and arc_j is removed. This procedure is performed iteratively and processes the rest of the four groups, subsequently.

The results of all the segmented arcs in the each group are moved into a set $\mathbf{AS} = \{ARC_i = arc_i, i = 1, 2, \ldots\}$. Fig. 6(a) shows the estimated circles for each of the 55 short straight lines (the circle consists of 396 pixels) from a circle with a radius of 70 pixels. Also, by using the previous procedure we can obtain the 8 arc segments in the eight groups which are represented by the black squares shown in Fig. 6(b).

2.4 Clustering arc segments

Once the segmented arcs are available, the next stage is to perform an interpretation process consisting of three steps which are similar to the previous procedure. Functions $H_{1,2,3}(ARC_i, ARC_j)$ in each step give a value assessing the possibility that the clusters of arc segments ARC_i, ARC_j may be part of a circle.

2.4.1 Step 1

Starting from the initial set **AS**, find the relationship between the arc segments from any part of a contour and between different contours. In the function $H_1(ARC_i, ARC_j)$, we use the same circumference condition.

- $H_1(ARC_i, ARC_j)$: First, we check θ_{in} between $ARC_i, ARC_j (i \neq j)$. If θ_{in} exists within the range given by the following equation:

$$135^\circ - D4 \leq \theta_{in} \leq 135^\circ - D4, \quad (9)$$

the circle parameters $(\tilde{R}, \tilde{x}, \tilde{y})$ are estimated by equations (1), (2), and (3). To confirm the accuracy of parameters, we calculate θ_{est} using the with four extremity points by equation (7). If the relationship between each tangent θ_{line} from the four extremity points and θ_{est} satisfy equation (8), the merged circle parameters $(\tilde{R}, \tilde{x}, \tilde{y})$ and a new θ_{arc} and the two extremities $(cx_{1i}, cy_{1i}), (cx_{2i}, cy_{2i})$ for the pair (ARC_i, ARC_j) are recorded in the set **AS** as a new ARC_i and ARC_j is removed.

When the termination condition is satisfied, the results are a set **AS** of clusters of arc segments which may represent part of the same circumference.

2.4.2 Step 2

Starting from the initial set **AS** generated in step 1, the procedure is almost the same except that the value of θ_{in} is set at 90° instead of 135°, between the arc segments in the function $H_2(ARC_i, ARC_j)$.

2.4.3 Step 3

Finally, in step 3 a criterion decides whether a segment, or a cluster of segments, represents a circle, clustering them together. The parameters of the circle are then estimated using both candidates clusters ARC_i, ARC_j.

- $H_3(ARC_i, ARC_j)$: After finishing steps 1 and 2, the estimated cluster of arc segments and their circle parameters are recorded in the set **AS**. If the two arcs satisfy by the following equation:

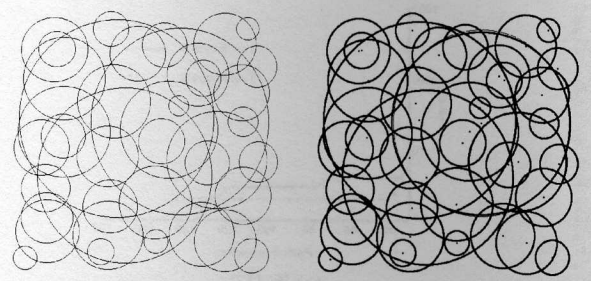
$$(|\tilde{R}_i - \tilde{R}_j|, |\tilde{x}_i - \tilde{x}_j|, |\tilde{y}_i - \tilde{y}_j|) < D3, \quad (10)$$

we finally estimate the circle parameters $(\tilde{R}, \tilde{x}, \tilde{y})$ as an extracted circle and removed ARC_j in **AS**.

After finishing this all process, we can expect to extract the circles from the set $\mathbf{AS} = \{ARC_i, i = 1, 2, \ldots\}$ if the number of edge pixels represents at least 40% of the total pixels of the circumference.

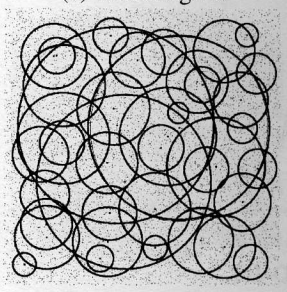
3 Experiments

All tests were performed on 480×480 binary images using a 600–MHz Pentium–III processor. The constraining thresholds in all experiments are set at $D1 = 40$ pixels, $D2 = 100$ pixels, $D3 = 20$ pixels, and θ_{er} and $D4 = 10^\circ$.

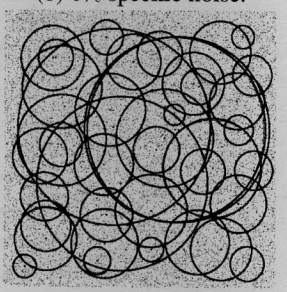


(a) Test image 1.

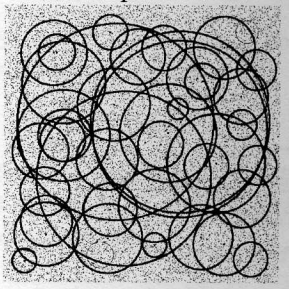
(b) 0% speckle noise.



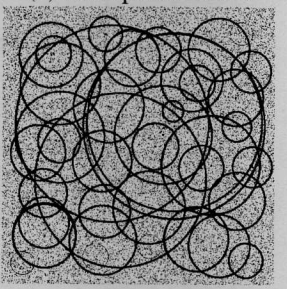
(c) 5% speckle noise.



(d) 10% speckle noise.



(e) 15% speckle noise.



(f) 20% speckle noise.

Figure 7: *Experiment (1).*

3.1 Experiment (1)

The algorithm was tested on the complicated synthetic image that is shown in Fig. 7(a). Test image 1 consists of 40 circles with radii ranging from 18 to 170 pixels. To prove the robustness of the proposed algorithm, we randomly corrupted the image by adding speckle noise. The definition of the noise level is taken from reference [9]. The noise levels are 0%, 5%, 10%, 15%, and 20%. The results are shown in Table 1 and Fig. 7. The proposed algorithm quickly and accurately extracted circles. However, not all of the circles are extracted from the images with noise levels from 10% to 20% because the short straight lines of the small circles were not extracted due to the influence of the heavy speckle noise as shown in Fig. 8.

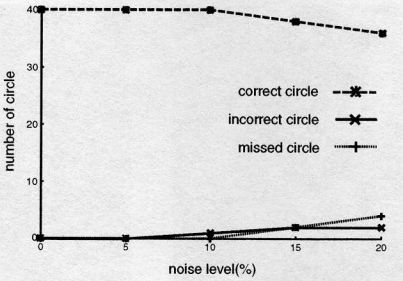


Figure 8: *Extracted circles from each noise level.*

Table 1: *Experimental results (1) : A (number of pixels), B (number of lines), C (incorrect circle), D (missed circles), E (correct circles), F (noise level), G (execution time).*

	7(b)	7(c)	7(d)	7(e)	7(f)
A	12040	22708	32894	42537	51646
B	1609	1567	1578	1781	2348
C	0	0	1	2	2
D	0	0	0	2	4
E	40	40	40	38	36
F	0 %	5 %	10 %	15%	20 %
G	0.71(s)	0.71(s)	0.71(s)	0.78(s)	1.0(s)

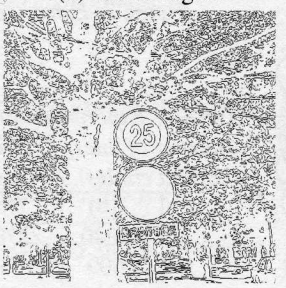
3.2 Experiment (2)

In this section, we compared our algorithm to HT-based and non-HT-based algorithms for real-world images as shown in Fig. 9(a) and (b). Test images 2 and 3 were extracted with a Canny edge operator [11] as shown in Fig. 9(c) and (d). Test image 2 consists of some coins with occluded pixels ranging from 30% to 50%. It also has other circular objects with concentric circles. Test image 3 has traffic signs containing two concentric circles and one circle which are embedded in a complicated background condition. First, some representative HT-based algorithms, T–HT [1], D–HT [3], Y–HT [5], were performed . Since Y–HT was developed to extract ellipses, we modified it a little to extract only circles, but the original idea is the same which is a combinational computation. Fig. 10 shows the relationship between the peak threshold and the number of selected candidate center points for the test images. From Fig. 10, it can be seen that finding a peak value is very difficult. If we choose a high threshold, an occluded circle can not be selected as a candidate center point. On the other hand, if we use a low threshold, many false candidate center points will be selected. Table 2 shows the time required to calculate the parameter space. In the non-HT-based algorithms, we first ran the algorithm using geometrical symmetry properites [7]. The key of the algorithm is to detect vertical and horizontal lines. Since the algorithm does not use a histogram in the parameter space, the speed is very fast if the input images have perfect cir-

(a) Coins and circular objects.

(b) Traffic signs.

(c) Test image 2.

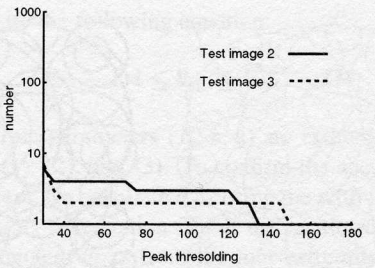


(d) Test image 3.

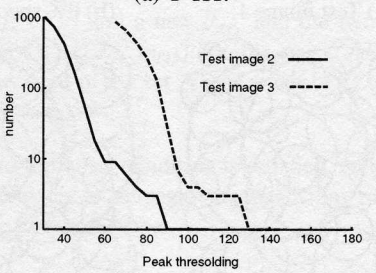
Figure 9: *Real-world images.*

Table 2: The execution time of each HT-based algorithm.

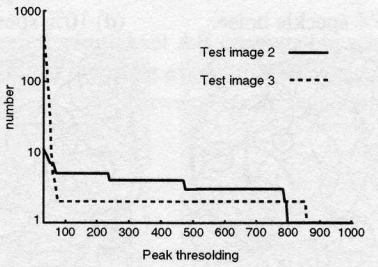
	T–HT	D–HT	Y–HT
Test image 2	16.25(s)	2.37(s)	62(s)
Test image 3	80.0(s)	4.11(s)	297(s)



(a) T–HT.



(b) D–HT.



(c) Y–HT.

Figure 10: The relationship between peak thresholding and the number of candidate center points.

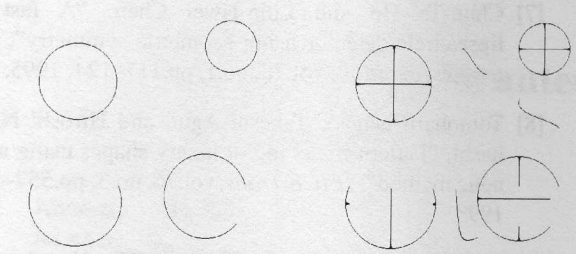
cles and un-complicated background conditions as shown Fig. 11(a) and (b). However, we failed to detect vertical and horizontal lines from test image 2 and 3 because of many extraneous edge pixels as shown in Fig. 11(c) and (d). The remaining non-HT-based algorithms are the circle extraction method using a genetic algorithm (CGA) [8] and the UpWrite [9]. A primitive CGA was developed to detect arbitrary objects. We changed some parameters for extracting only circles. The resulting images are shown in Fig. 12. The CGA extracted some circles which have perfect circumferences. However, the CGA failed to extract for occluded circles with complicated background from the images shown in Fig. 12(a) and Fig. 12(b). The reasons are that the fitness function is very sensitive in complicated backgrounds because the CGA uses the fitness value to confirm circles with a distance transformation. The UpWrite uses the spot algorithm and the program was downloaded from the Internet [12]. The UpWrite is very slow because the spot algorithm has a long computational time. Also, in the case of complicated background interfere it is difficult to make an accurate local model as shown in Fig. 12 (c) and (d). Table 3 shows the execution times for the CGA and the UpWrite algorithms.

Table 3: The execution time of the CGA and UpWrite algorithm.

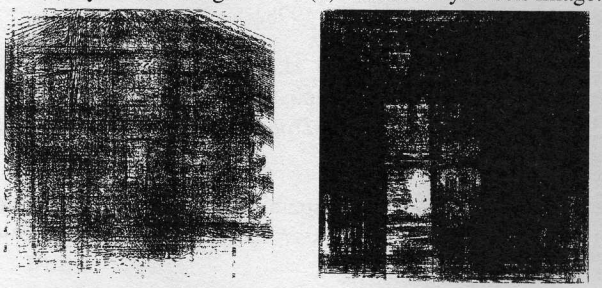
	UpWrite	CGA
test image 2	108(s)	122(s)
test image 3	466(s)	67(s)

3.3 Experiment (3)

Our proposed algorithm was used in experiment (3) for test image 2 and 3. The results are shown in Table 4 and Fig. 13(a) and (b). The proposed algorithm extracted all circles even containing some occlusion, concentric circles even when embedded in complicated background images with accuracy, speed, and robustness. Also, to show that the algorithm is robust corrupted images with 10% speckle noise were tested. From Fig. 13(c) and (d) it can be seen that the

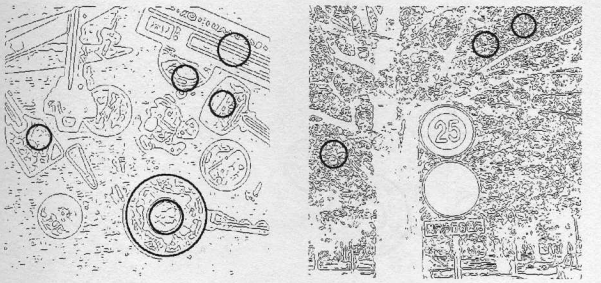


(a) Synthesis image. (b) Result of synthesis image.

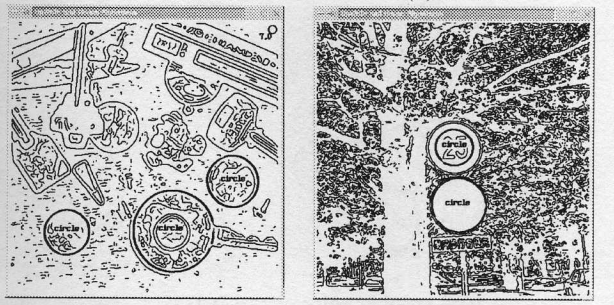


(c) Result of test image 2. (d) Result of test image 3.

Figure 11: Results of the horizontal and vertical scanning transform procedure.



(a) CGA. (b) CGA.



(c) UpWrite. (d) UpWrite.

Figure 12: Result of the CGA and UpWrite algorithms.

Table 4: *Experimental results (2) : A (number of pixels), B (number of lines), C (correct circles), D (noise level), E (execution time).*

	A	B	C	D	E
test image 2	15923	935	6	0%	0.25(s)
test image 3	34158	1383	3	0%	0.35(s)
test image 2	36360	1126	6	10%	0.44(s)
test image 3	52838	3324	3	10%	1.25(s)

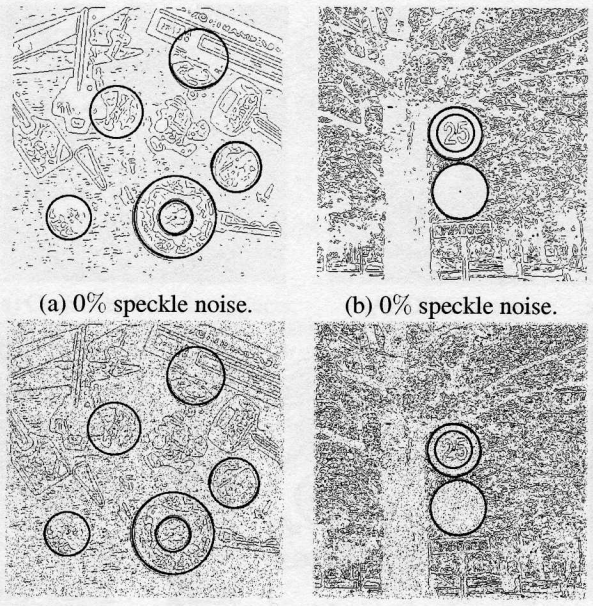
proposed algorithm extracted the circles with high speed and accuracy.

4 Discussion

In this section, we discuss the results of the experiments. Experiment (1), shown in Fig. 7, tests the complicated and variously sized circles. Our proposed algorithm can extract the circles with accuracy, speed, and robustness. Experiment (2), shown in Fig. 9, uses complicated real-world images to test accuracy, speed, and robustness for HT-based and non-HT-based algorithms. Y–HT using combinatorial computation can accumulate robust histograms for perfect and occluded circles with complicated background images. However, the speed as shown in Table 2 is very slow. D–HT is a very simple algorithm and faster than other HT-based algorithms. However, it is very difficult to decide a suitable peak threshold. In particular, since the algorithm utilized line-beam accumulating from each edge pixel, histograms respond sensitively to extraneous edge pixels. The geometrical algorithm is also very sensitive to occluded circles and extraneous edge pixels. Though the CGA and UpWrite do not use a parameter space, the accuracy, speed, and robustness are very low. All these previous algorithms may extract circles accurately and quickly under some restricted conditions. Our algorithm does not use a histogram parameter space like the non-HT-based algorithms. However, we can obtain the results with accuracy, speed, and robustness because our algorithm uses short straight lines instead of using each edge pixel. Our algorithm is very flexible for various real-world images and we did not need to control the parameters for each of the test images 1,2, and 3.

5 Conclusion

This paper has presented a new circle extraction algorithm-using short straight lines that is capable of extracting circles with high accuracy and speed, and is robust in the presence of heavy noise. The accuracy and speed of most conventional methods depend on the number of edge pixels because they process the image pixel by pixel. Our algorithm uses short straight lines instead of individual edge pixels. This concept facilitates the estimation of circle parameters by using the least-squares algorithm. The proposed algorithm can



(a) 0% speckle noise. (b) 0% speckle noise.

(c) 10% speckle noise. (d) 10% speckle noise.

Figure 13: Results of the proposed algorithm.

be extended to extracting ellipses.

References

[1] S. Tsuji and F. Matsumoto, "Detection of ellipses by a modified Hough transformation", *IEEE Trans. Comput*, vol.C-27, no.8, pp.777–781, 1978.

[2] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes", *Pattern Recognition*, vol.13, no.2, pp.111–122, 1981.

[3] E. R Davies, "A modified Hough scheme for general circle location", *Pattern Recognition Letter*, vol.7 no.1, pp.37–43, 1988.

[4] N. Guil and E. L. Zapata, "Lower order circle and ellipse Hough Transform", *Pattern Recognition*, vol.30, no.10, pp.1729–1744, 1997.

[5] H. K. Yuen, J. Illingworth, and J Kittler, "Detecting partially occluded ellipses using the Hough transform", *Image and Vision Computerting*, vol.7, no.1, pp.31–37, 1989.

[6] S. M. Thomas and Y. T. Chan, "A simple approach for the estimation of circular arc center and its radius", *Computer Vision, Graphics, and Image Processing*, vol.45, pp.362–370, 1989.

[7] Chun-Ta Ho and Ling-Hwei Chen, "A fast ellipse/circle detector using geometric symmetry", *Pattern Recognition*, vol.28, no.1, pp.117–124, 1995.

[8] Tomoharu Nagao, Takeshi Agui, and Hiroshi Nagahashi, "Pattern matching of binary shapes using a genetic method", *IEICE Trans*, vol.76, no.3, pp.557–565, 1993.

[9] R. A. McLaughlin and M. D. Alder, "The Hough transform versus the UpWrite", *IEEE Trans, PAMI*, vol.20, no.4, pp.396–400, 1998.

[10] Euijin Kim, Miki Haseyama, and Hideo Kitajima, "Fast line extraction from digital images using line segments", *IEICE Trans*, vol.84, no.8, pp.1566–1579, 2001.

[11] J. Canny, "A computational appoach to edge detection", *IEEE Trans, PAMI*, vol.8, no.6, pp.679–698, 1986.

[12] http://ciips.ee.uwa.edu.au/Paper/Journal_Papers/1998/02/Index.html