International Journal of Innovative Computing, Information and Control
Volume **8**, Number **1(A)**, January **2012**

ICIC International ©2012 ISSN 1349-4198
pp. 151–166

A FAST RANDOMIZED METHOD FOR EFFICIENT CIRCLE/ARC DETECTION

Shih-Hsuan Chiu[1], Kuo-Hung Lin[1], Che-Yen Wen[2], Jun-Huei Lee[1]
and Hung-Ming Chen[1]

[1]Department of Materials Science and Engineering
National Taiwan University of Science and Technology
No. 43, Sec. 4, Keelung Rd., Taipei 106, Taiwan
{ schiu; D9304301; D9104301; D9704004 }@mail.ntust.edu.tw

[2]Department of Forensic Science
Central Police University
No. 56, Shujen Rd., Takang Village, Kueishan Hsiang, Taoyuan County 33304, Taiwan
cwen@mail.cpu.edu.tw

Received August 2010; revised December 2010

Abstract. *Circle/arc detection plays an important role in image processing and machine vision. The Hough transform has been applied to circle/arc detection, and many multi-step based methods have been proposed for improving its performance (computation and storage space). The multi-step iterative procedure to find candidate circles/arc includes: picking initial points, finding correspondent searching points with some predefined geometric properties, and obtaining candidate circles/arcs. However, the number and distribution of the initial points are keys for efficient detection. In this paper, we propose a new circle/arc detection method, Fast Randomized method for Efficient Circle/arc Detection (FRECD). It just requires one "neighbor" point of target circles/arcs as the initial point. Besides, the proposed FRECD does not use storage for voting space. From the experimental results, the proposed FRECD provides better performance than previous multi-step based methods.*

Keywords: Multi-step based methods, Circle/arc detection, Randomized method

1. **Introduction.** Pattern recognition has been applied in many fields. In digital signal processing, the SOM (self organizing maps) based feature extraction method was introduced to analyze EEG (Electroencephalograph) signals from motor imagery [1]. In machine vision, a neural network was applied to recognize the motions pattern [2]. In medical image analysis, an automatic detection of candidate regions was proposed by using density and shape features [3]. For geometric patterns, circle/arc detection plays an important role in image processing and machine vision, such as iris detection/segmentation [4], steering wheel (circular pattern) detection for intelligent transportation system [5], ball detection for sports video [6], circle localization for PCB optics image [7] and the detection of safety helmet for ATM's surveillance system [8]. The Hough transform is the most popular technology used for pattern detection. It transfers spatial data into a parameter space. Each spatial point is represented by specific parameters to vote for. However, its heavy computation and storage requirements make it difficult to be applied to real-time systems. For example, a circle pattern contains three parameters: (x', y') and r', which are respectively the coordinate of the circle center and radius. The order of the computational complexity for Hough transform to execute the voting procedure is three.

Many modified methods have been proposed to reduce the heavy load (computation and storage requirements) of the Hough transform. Those methods can generally be classified as the gradient-based methods [9,11] and the probabilistic methods [12-23]. Since the gradient information of edge contours is sensitive to noise, the gradient-based methods may fail while vague noise appears near the target circles/arcs [12]. On the other hand, the probabilistic methods accomplish the circle/arc detection processing into the following multi-step: (1) picking initial points (n_i), (2) finding searching points (n_s) based on geometric properties, and (3) obtaining candidate circles/arcs. In general, the probabilistic methods provide better performance (less computation and storage requirements) than the gradient-based methods [12]. However, the number and distribution of the initial points are keys for efficient detection. Our method is proposed based upon these points of view:

(1) The number of initial points

The RHT (randomized Hough transform) randomly picks three initial points ($n_i = 3$) to solve circle parameters [15-17]. Its improved version, the efficient randomized circle detection algorithm (RCD) [18], uses four initial points ($n_i = 4$) and applies a distance criterion to check whether the initial points lie on the same circle/arc. However, no matter three or four initial points, the probability of all initial points lying on the same circle/arc under complex background (or noise) is low. The iterations will explosively be increased while noise is increasing. For example, a $W \times H$ image (total M pixels) contains a circle (with an N point circumference), and the probability for picking n_i initial points on the same circle can be approximated as $(N/M)^{n_i}$, since $M \gg N$. We can see that the probability is far less than one while n_i increasing [18].

A simple way to increase the above probability, $(N/M)^{n_i}$, is to use less number of initial points (i.e., $n_i < 3$). The semi-random detection method (SRD) [19] randomly picks two initial points ($n_i = 2$) at a valid row and uses them to find searching points ($n_s = 1$ or 2) by following the vertical direction. Any three points (from n_i and n_s) can be used to obtain a candidate circle based on a right triangle. A SRD-like method which only utilizes one initial point ($n_i = 1$) for circle detection is developed [20]. However, all above methods are not suitable for arc (or dot circle) detection.

The so-called efficient voting method [21] picks two initial points ($n_i = 2$) and uses one of them to find a searching point ($n_s = 1$) within a pseudo circle region. All the three points will vote for only one candidate circle which has the highest existing rate. The fast randomized Hough transform (FRHT) [8,22] picks only one initial point (P_1) and treats it as the center of a window region. Within this window, any two points (not P_1) with the same distance to P_1 will be used to solve a candidate circle based on chord properties. Since the probability of picking one initial point, $(N/M)^1$, is much larger than that of picking four points, $(N/M)^4$, the method consumes less iterations than the method that needs four initial points. However, all above methods require storage for voting space.

(2) The distribution of initial points

The main constraint of the above mentioned methods, no matter what the number of initial points is, their initial point(s) must lie on the targeted circle/arc for a successful detection. Zhang and Cao intended to use only one initial point inside a targeted circle region to increase the probability of picking an initial point [23]; however, it may fail while the initial point is picked outer the target circles. It is also not suitable for arc (or dot circle) detection.

According to above points of view, we can conclude that an efficient multi-step based method for circles/arcs detection should possess the following four properties:

(1) It needs as few initial points as possible. Since the probability for picking n_i initial points on the same circle can be approximated as $(N/M)^{n_i}$, we can obtain a higher probability with smaller n_i. For example, $(N/M)^1$ is larger than $(N/M)^4$.
(2) It can pick the initial point(s) which is (are) neighboring to target circles/arcs, rather than lie on the target circles/arcs.
(3) It can detect both circles and arcs.
(4) It does not use storage for voting space.

In this study, the proposed fast randomized method for efficient circle/arc detection (FRECD) can satisfy above four properties. This paper is organized as follows: Section 2 summarizes several geometric properties of a circle pattern. Section 3 is the proposed fast randomized method for efficient circle/arc detection (FRECD). Section 4 shows the experimental results and discussions. Finally, the conclusions will be addressed in Section 5.

2. **Properties of a Circle Pattern.**

The parameters of a circle can be decided by three points. In Figure 1(a), there are three points a, b and e on a circle, and their coordinates are (x_a, y_a), (x_b, y_b) and (x_e, y_e), respectively. To simplify the explanation of the proposed method, we summarize some well-known geometric properties of a circle as follows:

Property I:

Any two points on a circle can obtain a middle-perpendicular line. The middle-perpendicular line $L_{\perp ab}$ with respect to L_{ab} can be denoted:

$$y - y_m = t \cdot (x - x_m), \tag{1}$$

where (x_m, y_m) is the coordinate of the middle point m of a and b, and the slope t is equal to $-(x_a - x_b)/(y_a - y_b)$.

Property II:

The coordinate of a circle center (x', y') can be obtained by the intersection of two middle-perpendicular lines (e.g., $L_{\perp ab}$ and $L_{\perp be}$). The radius r' is the distance from the circle center to any point a (b or e).

Property III:

The distances from any points on $L_{\perp ab}$ to a and b are the same; e.g., $\overline{S_l a} = \overline{S_l b} = h_l$. Similarly, $\overline{S_o b} = \overline{S_o e} = h_o$ and $\overline{S_i b} = \overline{S_i e} = h_i$ can be obtained with respect to the middle-perpendicular line $L_{\perp be}$.

Property IV:

For a middle-perpendicular line of a circle, we can find symmetric point pairs. For example in Figure 1(b), we can find the symmetric point pair, $d(x_d, y_d)$ and $c(x_c, y_c)$, with respect to $L_{\perp ab}$. Given the points a, b and c, the coordinate (x_d, y_d) can be obtained by the following steps:

1) Obtaining the line equation of $L_{\perp ab}$ according to Equation (1) and representing it as:

$$Ax + By + C = 0, \tag{2}$$

where $A = t$, $B = -1$, $C = y_m - t \cdot x_m$.

2) Supposing $c(x_c, y_c)$ and $d(x_d, y_d)$ form a symmetric point pair with respect to $L_{\perp ab}$, we can obtain Equation (3) from Equation (2):

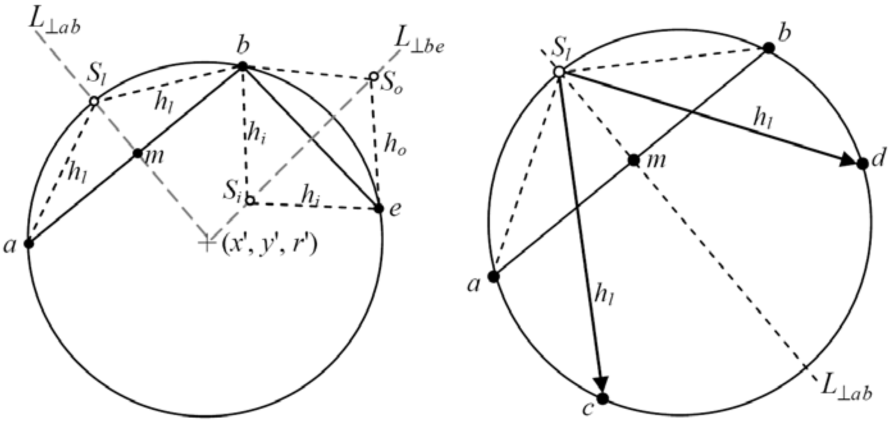
$$A\left(\frac{x_c + x_d}{2}\right) + B\left(\frac{y_c + y_d}{2}\right) + C = 0. \tag{3}$$

Since $L_{\perp ab}$ and L_{cd} are perpendicular, their slopes are negative reciprocals of each other:

$$\frac{y_c - y_d}{x_c - x_d} \cdot \left(-\frac{A}{B}\right) = -1. \tag{4}$$

3) The coordinate $(x_d,\ y_d)$ of point d can be computed from Equations (3) and (4):

$$\begin{cases} x_d = x_c - \dfrac{2 \cdot A(A \cdot x_c + B \cdot y_c + C)}{A^2 + B^2} \\ y_d = y_c - \dfrac{2 \cdot B(A \cdot x_c + B \cdot y_c + C)}{A^2 + B^2} \end{cases} \tag{5}$$



(a) (b)

FIGURE 1. The illustration of properties of a circle pattern: (a) perpendicular lines of chords (e.g., $L_{\perp ab}$ and $L_{\perp be}$) and symmetric chords (e.g., h_l, h_o and h_i); (b) symmetric point with respect to a perpendicular line; e.g., the c and d are the symmetric point pair with respect to $L_{\perp ab}$

3. Fast Randomized Method for Efficient Circle/Arc Detection.

Given a $W \times H$ edge image with E edge points, we denote D as a set of all edge points:

$$D = \{P_i(x_i, y_i) | i = 1, 2, \ldots, E\}, \tag{6}$$

where $P_i(x_i,\ y_i)$ denotes the coordinate of an edge point. As shown in Figure 2, the proposed fast randomized method for efficient circle/arc detection (FRECD) is summarized as follows:

Step 1: Initialization

Let I be the iteration number, and $I = 0$ initially. T_i is the predefined iteration number that we can tolerate. Let N be the current detected circle number, and $N = 0$ initially. T_n is the target circle number. In the meanwhile, we create a cell sequence $C_d(j)$, where $j = 1, \ldots,\ d$, where

$$d = \sqrt{W^2 + H^2}. \tag{7}$$

Step 2: Initial point selection

We randomly pick a point S as an initial point (any point from edge, noise or background point) and let $I = I + 1$. If $I = T_i$, we stop the whole procedure; otherwise, we forward to Step 3.

Step 3: Window creation

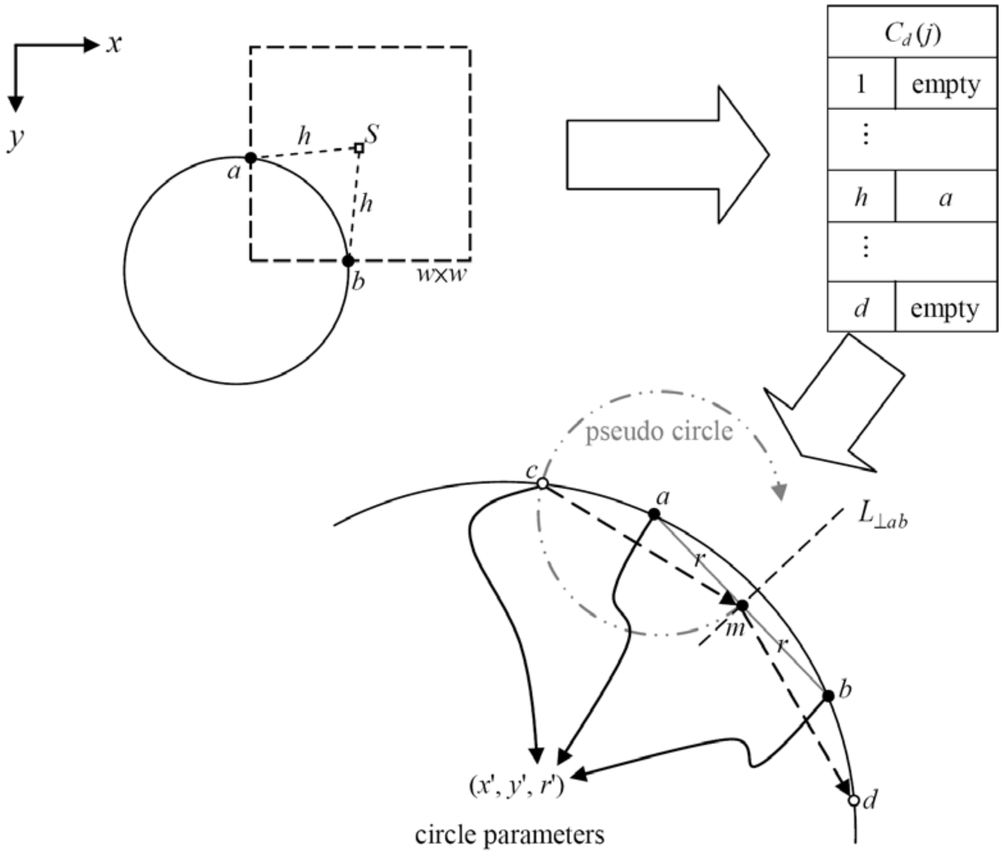


Figure 2. The illustrations of the proposed FRECD. The point S is the initial point; the points a and b are the searching points. Within the pseudo circle region, a third point c is used to check whether valid according to Property IV. If the point c is valid, a candidate circle/arc is obtained by using the three points (i.e., a, b and c).

We create a window region with $w \times w$ size that is centered at the initial point S.

Step 4: Symmetric point searching

Within the window, we pick edge points P_i ($P_i \in D$ and $P_i \neq S$) one by one and compute the distance between P_i and S. In Figure 2, a point, a, is picked ($a \in D$), and the distance h is computed between a and S. If the correspondent cell $C_d(h)$ is empty, we store the point a to the cell $C_d(h)$ and continue to pick the next point. When we find a point b with the same distance h (i.e., a and b form a symmetric point pair and may be on the same circle), we will forward to Step 5 and check their geometric properties. Once we cannot find any symmetric point pair, we empty the cell sequence and return to Step 2.

Step 5: Geometric property checking

In Figure 2, if the symmetric point pair (points a and b) is on the same circle, we can find their middle point m (x_m, y_m), where

$$\begin{cases} x_m = (x_a + x_b)/2 \\ y_m = (y_a + y_b)/2 \end{cases} . \tag{8}$$

The distance r between a (or b) and m is

$$r = \|(a, m)\| = \|(b, m)\|, \tag{9}$$

where $|| \bullet ||$ denotes Euclidean distance between two points; e.g., $||(a, m)||$ denotes the Euclidean distance between point a and m. We define a pseudo circle with radius r and center a. If a point, c, on the pseudo circle whose distance to the center a satisfies:

$$||(c, a)|| - r \leq \Delta_r, \tag{10}$$

where Δ_r is the maximum tolerant error between $||(c, a)||$ and r. In all experiments of this study, we allow $\Delta_r = 1$. We will check whether its symmetric point $d(x_d, y_d)$ exists (i.e., $d \in D$) based upon Property IV. If $d(x_d, y_d)$ can be found, we will consider the three point (a, b, c) can form a candidate circle. The parameters (x', y', r') can be obtained by Properties I and II. We continue the process to Step 6. If all points within this pseudo circle region are searched, we empty $C_d(j)$ and return to Step 4.

Step 6: Circle detection

We use a parameter (existing rate) to decide if a circle/arc exists. The edge points on the candidate circle of (x', y', r') can be described as:

$$D' = \left\{ P_j | \left| \sqrt{(x_j - x')^2 + (y_j - y')^2} - r' \right| \leq \varepsilon_r \right\}, \tag{11}$$

where $D' \subseteq D$, and ε_r is the maximum tolerant error ($\varepsilon_r = 1$ in this study) near to the radius r'. We regard the candidate circle as the target circle while

$$|D'| \geq 2\pi r' T_r, \tag{12}$$

where $|D'|$ is the cardinality of D', and $T_r \in [0, 1]$ is the predefined existing rate of the obtained candidate circle. If the candidate circle is confirmed as the target circle by Equation (12), we remove all elements of D' out of D and empty the cell sequence; meanwhile, we let the current detected circle number be $N = N + 1$. If $N = T_n$, we stop the whole procedure; otherwise, return to Step 2.

4. **Experiments and Discussions.**

Since the efficient randomized circle detection (RCD) and the fast randomized Hough transform (FRHT) are similar to the proposed method (FRECD). They randomly pick four (one) initial points with a predefined distance criterion (symmetric properties) to detect target circles/arcs. In the following description, we utilize geometric distribution theory to analyze the probability distribution and computation comparison with the RCD and FRHT. In addition, we use several synthetic and realistic images to compare the performance of the RCD, FRHT, and proposed FRECD. All experiments are performed on Windows Vista Operation System with Intel Core 2 Duo CPU (2.20 GHz) and executed (C++ programming) by using Borland C++ Builder 6.0.

4.1. **Probability analysis.**

Since an efficient initial point picking algorithm is crucial to the performance of detection methods, the probability point of views of the geometric distribution of initial points is discussed. While the RCD and FRHT need their initial point(s) must lie on the target circle/arc for a successful detection, the proposed method just requires one neighbor point (even noise or background) of target circles/arcs as the initial point.

We use V (valid picking) and $F = 1 - V$ (failed picking) to denote the probability of event of randomly picking valid initial point(s) and invalid one(s), respectively. Let K be a random variable with a geometric distribution, the cumulative distribution function (CDF) can be denoted as [24]:

$$\begin{aligned} C_K(k) =& V + F^1 V + F^2 V + \cdots + F^{k-1} V \\ =& V \cdot \left(1 + F + F^2 + \cdots + F^{k-1}\right) \end{aligned}$$

$$\begin{aligned} &=(1-F)\cdot\left(\frac{1-F^k}{1-F}\right) \\ &=1-F^k, \end{aligned} \tag{13}$$

where $C_K(k)$ denotes the accumulated probability, and the random variable K denotes the number of failed picking until valid picking occurs exactly once.

If a $W \times H$ image (total M pixels) contains a circle with N edge points, the probability of randomly picking a valid initial point on the circle is:

$$\mathrm{P} = N/M \approx 2\pi \cdot r/M, \tag{14}$$

where $M = W \times H$ and r denotes the radius of the target circle.

For the RCD, it needs to pick randomly four valid initial points on a target circle/arc (valid picking for a candidate circle/arc), and the V (the probability of valid picking) of the RCD is:

$$V_{\mathrm{RCD}} \approx \left(\frac{2\pi \cdot r}{M}\right)^4 \approx \mathrm{P}^4. \tag{15}$$

For the FRHT, it only needs one valid initial point, and the V of the FRHT is:

$$V_{\mathrm{FRHT}} \approx \frac{2\pi \cdot r}{M} \approx \mathrm{P}. \tag{16}$$

For the proposed FRECD, it only needs one "neighbor" initial point (even noise or a background point). As shown in Figure 3, the "neighbor" initial point (S) can be picked in the region between the inner and outer dot circles. Therefore, the V of the FRECD is:

$$V_{\mathrm{FRECD}} \approx \frac{1}{M}\int_{r-w/2}^{r+w/2}(2\pi \cdot r)dr = w\frac{2\pi \cdot r}{M} = w \cdot \mathrm{P}, \tag{17}$$

where $w \in Z^+$ and denotes the window size. We can see that V_{FRECD} is much larger than V_{RCD} and V_{FRHT}. According to Equation (13), the CDFs of the three methods can be represented as:

$$C_{K,\mathrm{RCD}}(k) = 1-(1-V_{\mathrm{RCD}})^k = 1-(1-\mathrm{P}^4)^k, \tag{18}$$

$$C_{K,\mathrm{FRHT}}(k) = 1-(1-V_{\mathrm{FRHT}})^k = 1-(1-\mathrm{P})^k, \tag{19}$$

$$C_{K,\mathrm{FRECD}}(k) = 1-(1-V_{\mathrm{FRECD}})^k = 1-(1-w\cdot\mathrm{P})^k. \tag{20}$$

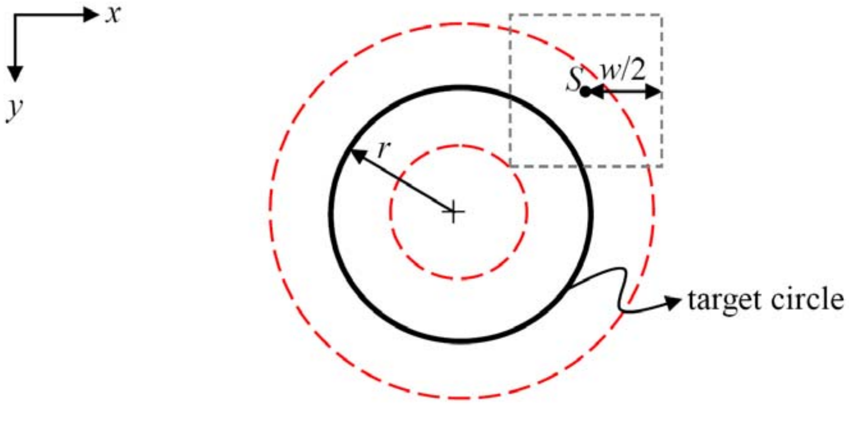


Figure 3. The illustration of the distribution of picking a "neighbor" initial point (e.g., S). It can be picked in the region between the inner and outer dot circles.

Physically, the $C_K(k)$ denotes the probability of picking valid initial point(s) at the kth iteration, and it means the past $(k-1)$ iterations of pickings are failed. We can use Equations (18)-(20) to obtain the plots of the three CDFs. As shown in Figure 4, we set P as two different values: 0.0035 and 0.0007, and it can be seen that the $C_{K,FRECD}(k)$ is higher than the others. That is, the probability of picking valid initial point(s) at the kth iteration of the proposed FRECD is higher than the others.

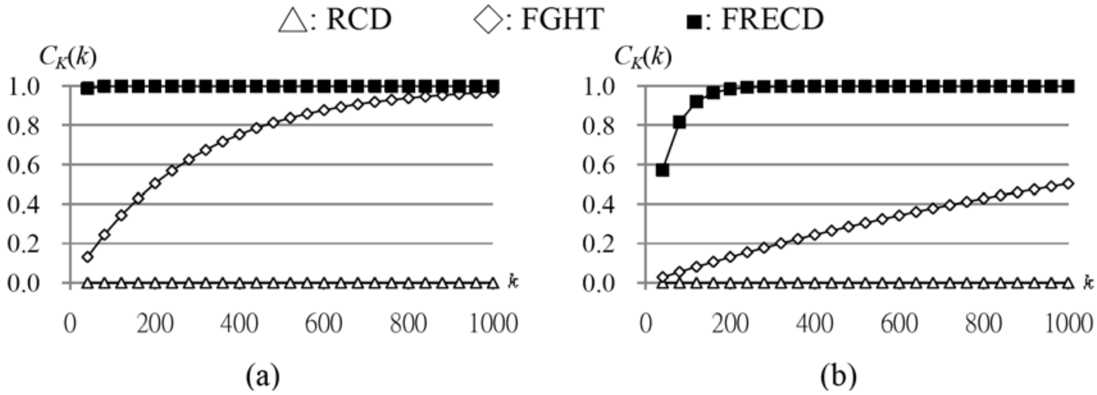


FIGURE 4. The distribution of cumulative density function, $C_K(k)$, while P is set as: (a) 0.0035; (b) 0.0007

4.2. **Computation analysis.** We use a $W \times H$ image (total M pixels) that contains a circle with $N \approx 2\pi \cdot r$ edge points on the circumference to estimate total computation for picking valid initial points and finding searching points. The computation (T) can be approximated as:

$$T \approx k \cdot I_f, \tag{21}$$

where $I_f \in Z^+$ denotes the iteration times of finding searching points during the past k iterations of pickings. As the discussions in Section 4.1, the probabilities of picking valid initial point(s) for the three methods are shown in Equations (15)-(17). In general, the minimum iteration times of the RCD for picking valid initial points $(n_i = 4)$ on the same circle/arc at the kth iteration is approximated as $1/V_{\text{RCD}}$. From Equation (15), we can obtain:

$$k(\text{RCD}) \approx \left(\frac{M}{N}\right)^4. \tag{22}$$

Since the RCD does not need to find any searching points, the total computation for RCD is approximated as:

$$T(\text{RCD}) \approx \left(\frac{M}{N}\right)^4. \tag{23}$$

For the FRHT, the minimum iteration time of the FRHT $(n_i = 1)$ for picking one valid initial point at the kth iteration is approximated as $1/V_{\text{FRHT}}$. From Equation (16), we can obtain:

$$k(\text{FRHT}) \approx \frac{M}{N}. \tag{24}$$

The FRHT needs to set a window region to find searching points $(n_s = 2)$. In the worst case, there are M pixels within the window region; i.e., the window covers the whole image. Therefore, the total computation is approximated as:

$$T(\text{FRHT}) \approx \frac{M}{N} \cdot M \approx \frac{M^2}{N}. \tag{25}$$

Similarly, the minimum iteration of the FRECD ($n_i = 1$) for picking a neighbor initial point at the kth iteration is approximated as $1/V_{\text{FRECD}}$. From Equation (17), we can obtain:

$$k(\text{FRECD}) = \frac{M}{wN}. \tag{26}$$

The proposed FRECD also needs a window region ($w \times w$) to find searching points ($n_s = 3$). In the worst case, all points in the window are considered as searching points, $I_f = w^2$; therefore, the total computation is approximated as:

$$T(\text{FRECD}) \approx \frac{M}{wN} \cdot w^2 = \frac{Mw}{N}. \tag{27}$$

The ratio between $T(\text{FRECD})$ and $T(\text{RCD})$ is obtained as:

$$\frac{T(\text{FRECD})}{T(\text{RCD})} \approx \left(\frac{Mw}{N}\right)\left(\frac{N}{M}\right)^4 \approx w\left(\frac{N}{M}\right)^3. \tag{28}$$

Since the value of (N/M) is less than one, we may conclude that the proposed FRECD will consume less computation than the RCD. Similarly, the ratio between $T(\text{FRECD})$ and $T(\text{FRHT})$ is obtained as:

$$\frac{T(\text{FRECD})}{T(\text{FRHT})} \approx \left(\frac{Mw}{N}\right)\left(\frac{N}{M^2}\right) \approx \frac{w}{M}, \tag{29}$$

where $w \in Z^+$. Since M (image size) is much larger than w (window size), we can conclude that the proposed FRECD will consume less computation than the FRHT.

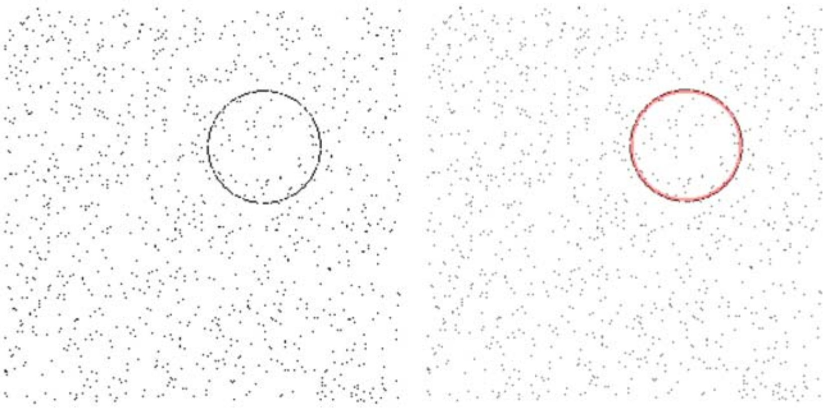
In what follows, we will use synthetic images to verify the feasibility (probability and computation analysis) of the proposed FRECD. These synthetic images contain several target circles/arcs which are artificial created and imposed with different levels of pepper noises. Besides, we use some realistic images that contain some objects with circular patterns to repeat the experiments. Table 1 shows the parameter settings of the three methods in our experiments: w denotes the window size, and it is set as 30 in all experiments heuristically. Δ_r and ε_r are two tolerance values described in Equations (10) and (11), respectively. T_r denotes the existing rate of the obtained candidate circle, which is described in Equation (12).

Table 1. The parameter settings of the three methods. The "×" denotes that the parameter is not needed.

Methods	w (window size)	Δ_r (pixel)	ε_r (pixel)	T_r (existing rate)
RCD	×	×	1	0.7
FRHT	30	×	1	0.7
FRECD	30	1	1	0.7

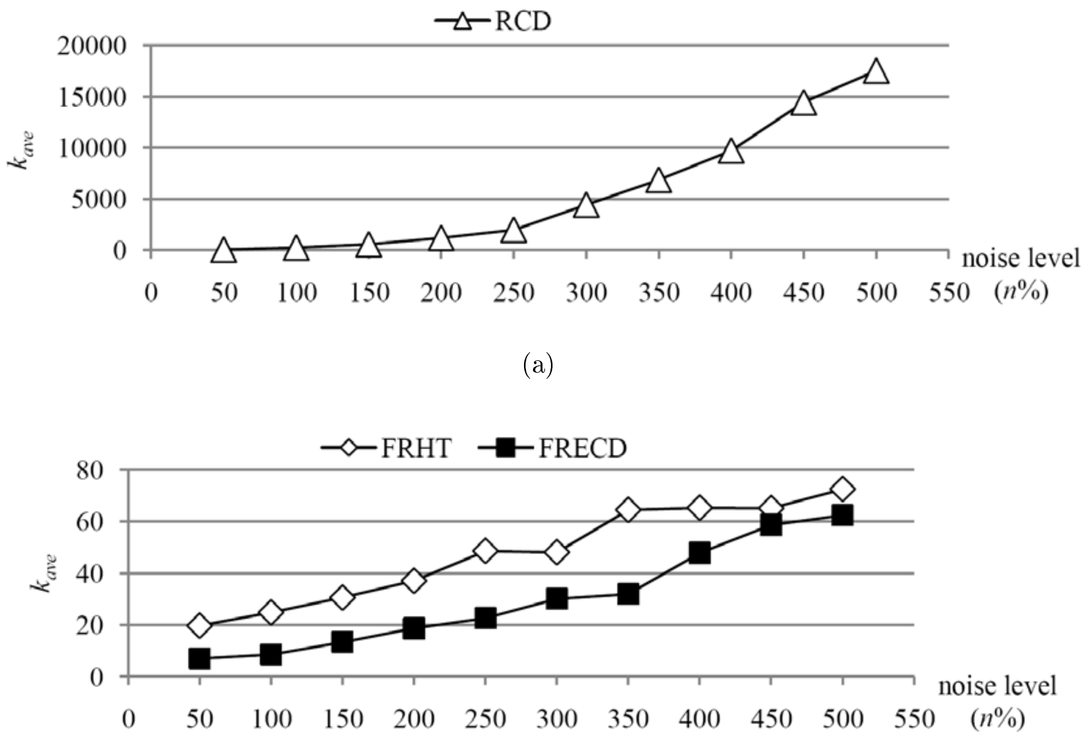
4.3. **Experiments for synthetic images.** An original synthetic image with a target circle is our test image. The radius r' of the circle is 37 pixels, and there are 232 edge points on the circle. In Figure 5(a), 500% pepper noise (1160 points) is imposed on the original circle image, and the detected result by using the proposed FRECD is shown in Figure 5(b).

We impose different levels of pepper noise on the original image. The thresholds T_i (the largest iteration that we can tolerate) and T_n (the excepted number of target circles) are set to be ∞ and 1, respectively. We obtain noisy images by changing the noise levels ($n\%$) from 50% (i.e., adding 116 noise points) to 500% (i.e., adding 1160 noise points). The experimental result shows that all the three methods can detect the target circle



(a) (b)

Figure 5. A synthetic image test (one target): (a) 500% noise imposed on the original image with one target circle; (b) the detection result of (a) by using the proposed FRECD



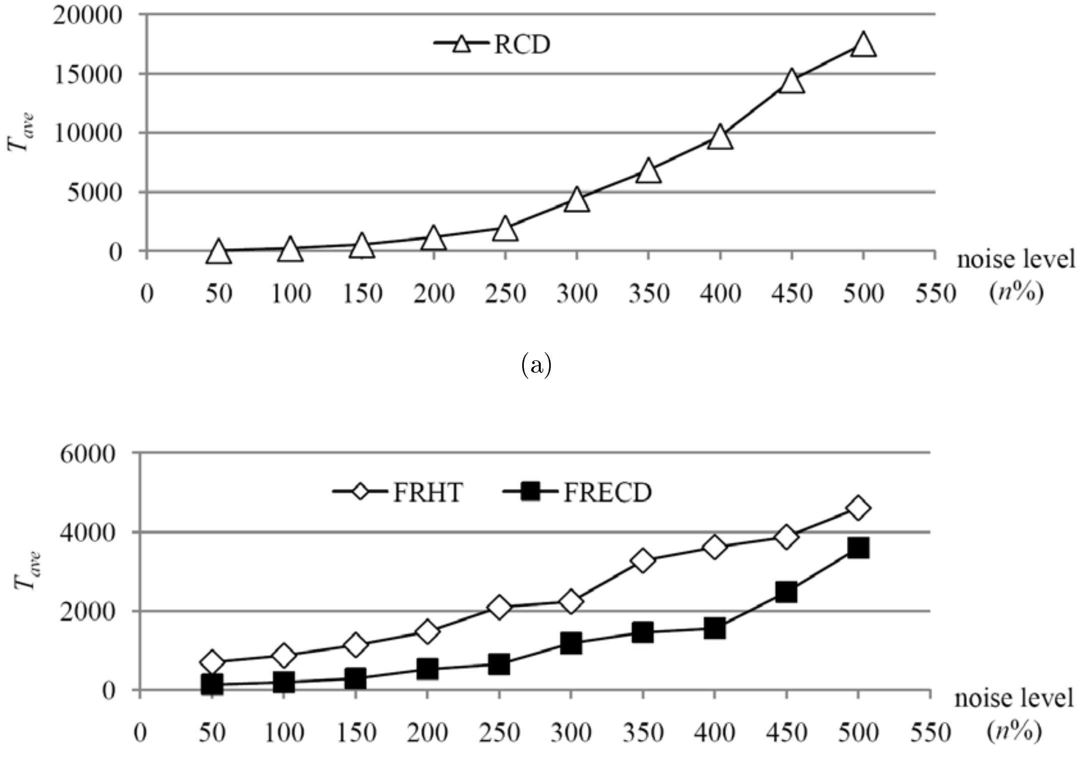
(a)

(b)

Figure 6. The experimental result of the synthetic image in Figure 5(a). The average iteration times (k_{ave}) of picking valid initial point(s) at the kth iteration of the three methods: (a) RCD; (b) FRHT and the proposed FRECD.

successfully. Figure 6 illustrates the average iteration times of picking valid initial points at the kth iteration, k_{ave} (obtained from 100 tests), for detecting the target circle. In

Figure 6(a), it can be seen that k_{ave} of the RCD is much larger than the others. In Figure 6(b), the proposed FRECD provides significant better performance than the FRHT in consuming less iteration times for picking a valid initial point: "neighbor" initial point. In the meanwhile, we obtain the average computation, T_{ave}, from the 100 tests. As shown in Figure 7, the proposed FRECD obviously consumes less computation than the others in each noise level.



(a)

(b)

Figure 7. The experimental result of the synthetic image in Figure 5(a). The average computation times, T_{ave} (obtained from 100 tests), of the three methods: (a) RCD; (b) FRHT and the proposed FRECD.

In Figure 8(a), we create three different target arcs in a 256 × 256 image (i.e., $T_n = 3$) and impose 500% pepper noise (3965 points) with the original image. If we let $T_i = \infty$ (the largest iteration that we can tolerate), all three methods can detect all targets successfully by changing the noise levels (n%) from 50% to 500%; Figure 8(b) shows the detected result by using the proposed FRECD.

Figure 9 illustrates the average iteration times of picking valid initial points at the kth iteration, k_{ave} (obtained from 100 tests), for detecting the target arcs. Since the probability of valid picking of the RCD is quiet low, it requires much iteration in picking valid initial point(s). In Figure 9(a), we can see that the k_{ave} of the RCD is much larger than the others. In Figure 9(b), the proposed FRECD provides significant better performance than the FRHT in consuming less iteration times for picking a valid initial point: "neighbor" initial point. In the meanwhile, we obtain the average computation, T_{ave}, from the 100 tests. As shown in Figure 10, the proposed FRECD obviously consumes less computation than the others in each noise level.

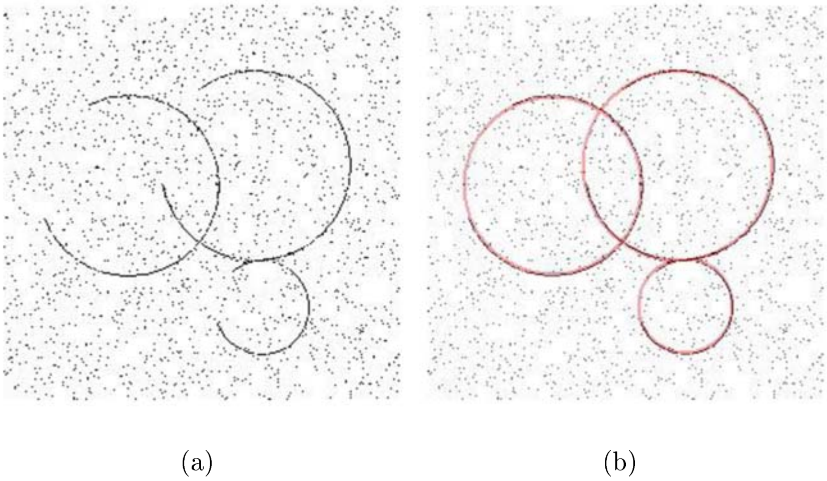


Figure 8. A synthetic image test (three targets): (a) 500% noise imposed on the original image with three target arcs; (b) the detection result of (a) by using the proposed FRECD

Table 2. The detection result of the moon image in Figure 11(a). The T_i, T_n and T_r are set as 10^7, 1 and 0.5, respectively. The average results: k_{ave} (iteration times of picking valid initial point(s) at the kth iteration) and T_{ave} (computation) are obtained from 100 tests.

Methods	k_{ave} (±S.D.)	T_{ave} (±S.D.)
RCD	16,265 (±12,865)	16,265 (±12,865)
FRHT	273 (±187)	225,782 (±154,983)
FRECD	132 (±113)	9,274 (±2,836)

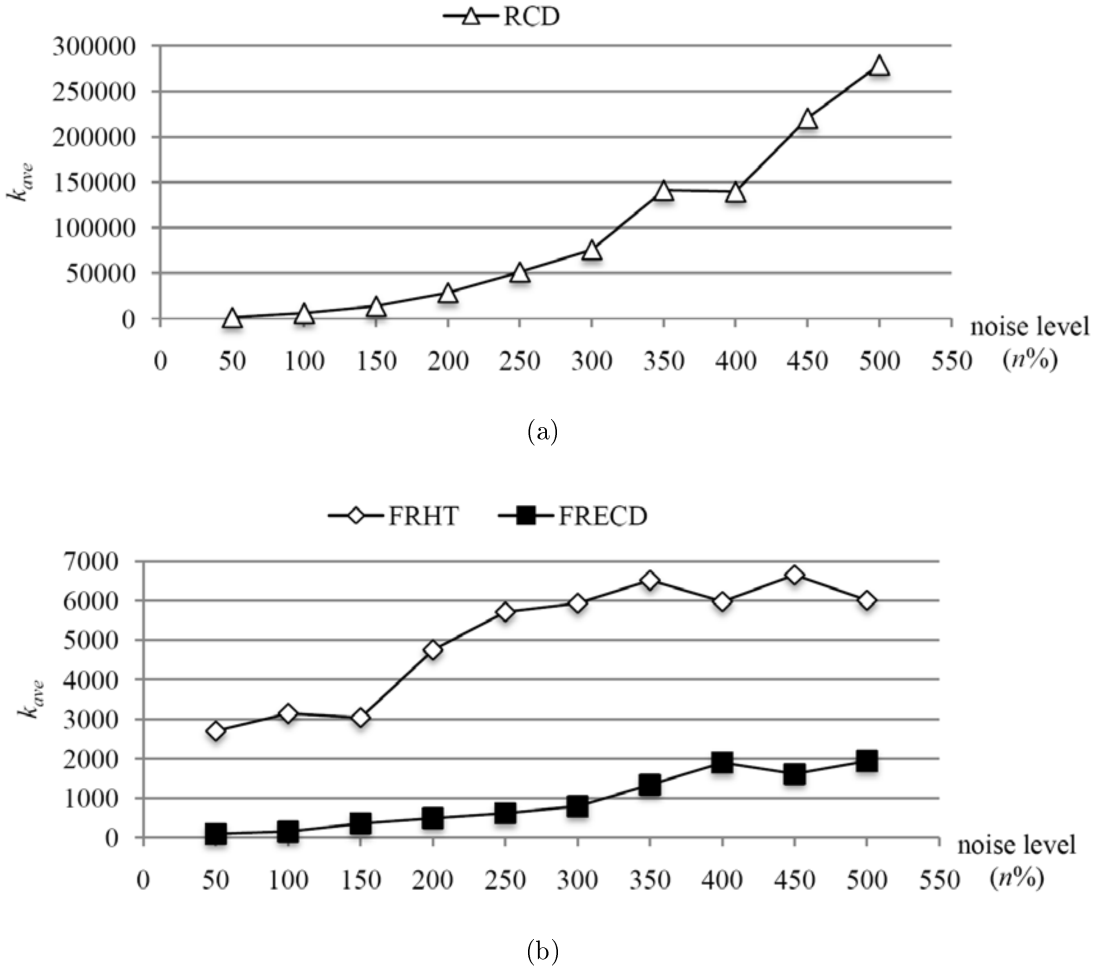
4.4. **Experiments for realistic images.** Several realistic images are used to compare the performance of proposed FRECD with the FRHT and RCD. Figure 11 shows the edge images of three realistic images: (a) the moon; (b) a traffic sign and (c) an NBA player. We use these three images for 100 tests to obtain the average results: k_{ave} (iteration times of picking valid initial point(s) at the kth iteration) and T_{ave} (computation).

In Figure 11(a), an arc pattern is on the moon edge, and Figure 12(a) shows its detection result. In Table 2, all of the three methods can detect the arc pattern successfully while T_i is set as 10^7. The k_{ave} (132 ± 113) of the proposed FRECD is much less than that of the other methods. In addition, the T_{ave} ($9,274 \pm 2,836$) of the proposed FRECD consumes the least computation than the others.

In Figure 11(b), there are two circles on the traffic sign. When $T_i = 10^7$, all of the three methods can detect the target circles successfully, and Figure 12(b) shows the detection result. In Table 3, the k_{ave} (985 ± 694) and T_{ave} ($44,211 \pm 31,142$) of the proposed FRECD are much less than that of the other methods.

In Figure 11(c), a circle pattern is on a complex background. When $T_i = 10^7$, all of the three methods can detect the target circle successfully, and Figure 12(c) shows the detection result. In Table 4, the k_{ave} (472 ± 353) and T_{ave} ($23,075 \pm 17,257$) of the proposed FRECD are much less than those of the other methods.

4.5. **Discussions of the probability of picking valid initial points.** In practice (similar to the synthetic and realistic experiments), the value of P (i.e., N/M) is ranged



(a)

(b)

Figure 9. The experimental result of the synthetic image in Figure 8(a). The average iteration times (k_{ave}) of picking valid initial point(s) at the kth iteration of the three methods: (a) RCD; (b) FRHT and the proposed FRECD.

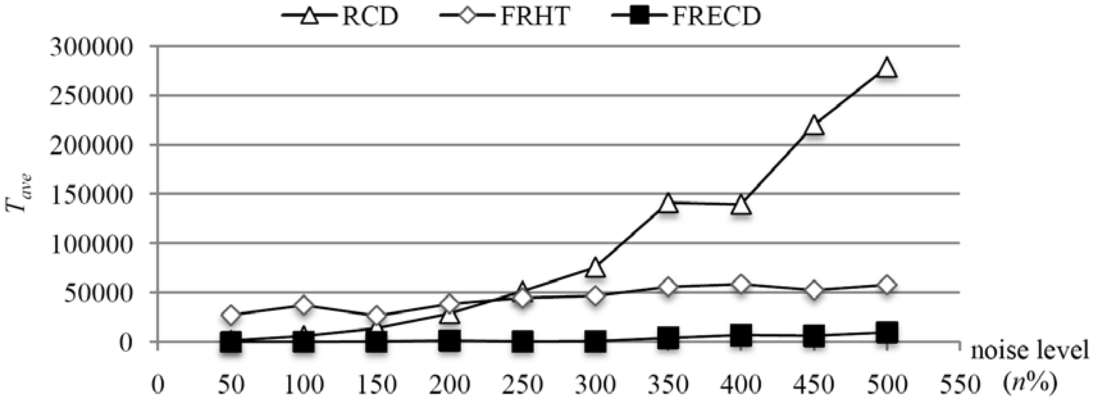


Figure 10. The experimental result of the synthetic image in Figure 8(a). The average computation, T_{ave} (obtained from 100 tests), of the three methods: RCD, FRHT and the proposed FRECD.

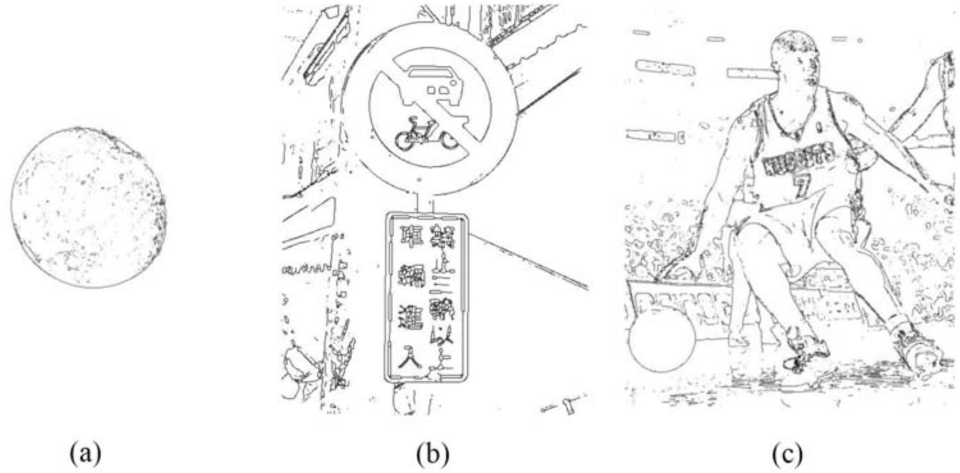


Figure 11. The edge images of three realistic images: (a) moon; (b) a traffic sign; (c) an NBA player

Table 3. The detection result of the traffic sign image in Figure 11(b). The T_i, T_n and T_r are set as 10^7, 2 and 0.7, respectively. The average results: k_{ave} (iteration times of picking valid initial point(s) at the kth iteration) and T_{ave} (computation) are obtained from 100 tests.

Methods	k_{ave} (±S.D.)	T_{ave} (±S.D.)
RCD	71,927 (±36,423)	71,927 (±36,423)
FRHT	5,989 (±2,431)	27,928,200 (±18,725,900)
FRECD	985 (±694)	44,211 (±31,142)

Table 4. The detection result of the NBA player image in Figure 11(c). The T_i, T_n and T_r are set as 10^7, 1 and 0.7, respectively. The average results: k_{ave} (iteration times of picking valid initial point(s) at the kth iteration) and T_{ave} (computation) are obtained from 100 tests.

Methods	k_{ave} (±S.D.)	T_{ave} (±S.D.)
RCD	8,786,500 (±1,123)	8,786,500 (±1,123)
FRHT	14,578 (±11,598)	4,434,380 (±3,528,730)
FRECD	472 (±353)	23,075 (±17,257)

from 0.0007 to 0.0043. We use these values to demonstrate the performance of required iteration before picking valid initial point(s).

(1) For the synthetic image experiment in Figure 5(a), the image size is 256×256 (i.e., M), and there are about 232 (i.e., N) edge points on the target circle, the P is approximated as $232/(256 \times 256) \approx 0.0035$. From Figure 4(a), we can see $C_{K,FRECD}(k)$ is the highest one. This is the reason that the FRECD consumes less iteration than others in picking a valid initial point, as shown in Figure 6(b). Furthermore, $C_{K,RCD}(k)$ of the RCD is far less than the others while $x \leq 1,000$. This explains why the RCD consumes much more iteration times than the others, as shown in Figure 6(a) and Figure 7(a).

(2) For the realistic image experiments in Section 4.4, the P values of the moon (in Figure 11(a)) and traffic sign images (in Figure 11(b)) are $1005/(591 \times 394) \approx 0.0043$ and $1343/(450 \times 600) \approx 0.0049$, respectively. Since those P's are close to the

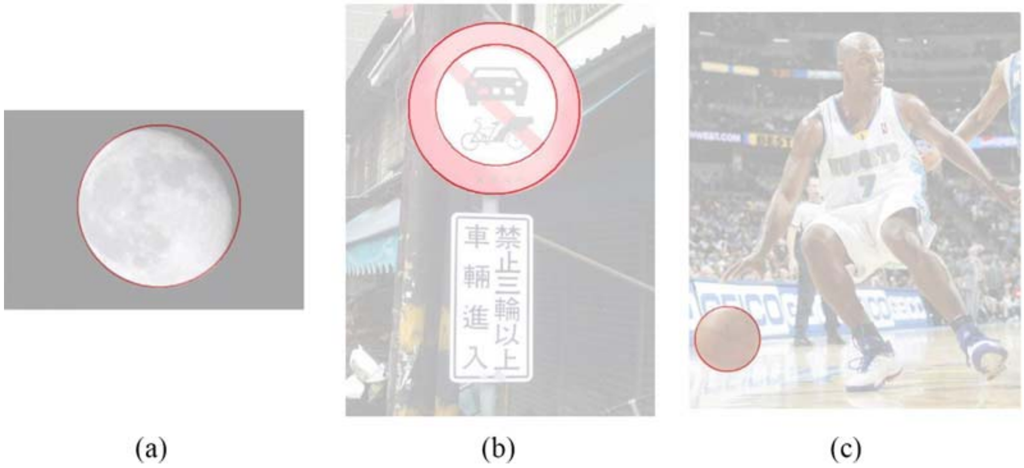


Figure 12. The edge images of three realistic images: (a) moon; (b) a traffic sign; (c) an NBA player

value (0.0035) in above synthetic image experiment, we can obtain similar results: the proposed FRECD has higher cumulated probability than the others, and the detection result shows that the proposed FRECD consumes the least iterations than the others. In Figure 11(c), there is a target circle (the basketball) with complex background. We have a small P value as $414/(666 \times 800) \approx 0.0007$. In Figure 4(b), we can see the proposed FRECD still has obviously higher cumulated probability than the others. That means the proposed FRECD has better performance than the other two methods in picking a valid initial point, especially in cases with noise or complex background.

5. **Conclusions.** In this paper, we propose a new circle/arc detection method, the fast randomized method for efficient circle/arc detection (FRECD). It only requires one "neighbor" point (even noise or background) of target circles/arcs as the initial point. From the probability and computation analysis, we can conclude that the FRECD consumes much less iteration times (for picking a valid initial point) and computation (T) than the RCD and FRHT. Moreover, the proposed FRECD does not use storage for voting space, and the experimental results indicate the proposed FRECD method provides better performance than previous multi-step based methods.

In order to be an efficient multi-step based method, the FRECD uses a bigger window size to obtain a higher probability of picking a valid initial point; however, the computation costs (i.e., computation T) will increase simultaneously (please see Equation (27)). Therefore, how to decide a proper window size becomes an important issue. For future work, we will find a way to decide an adaptive window size (w) instead of a predefined value.

REFERENCES

[1] T. Yamaguchi, K. Nagata, P. Q. Truong, M. Fujio and K. Inoue, Pattern recognition of EEG signal during motor imagery by using SOM, *International Journal of Innovative Computing, Information and Control*, vol.4, no.10, pp.2617-2630, 2008.

[2] H. Tamura, T. Gotoh, D. Okumura, H. Tanaka and K. Tanno, A study on the s-EMG pattern recognition using neural network, *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(B), pp.4877-4884, 2009.

[3] H. Kim, Y. Katsumata, Y. Itai, J. K. Tan and S. Ishikawa, Automatic detection of GGO candidate regions by using density and shape features, *International Journal of Innovative Computing, Information and Control*, vol.6, no.1, pp.255-263, 2010.

[4] J. Huang, X. You, Y. Y. Tang, L. Du and Y. Yuan, A novel iris segmentation using radial-suppression edge detection, *Signal Processing*, vol.89, no.12, pp.2630-2643, 2009.

[5] Y. Luo, J. Liu and X. Shi, An improved method of RHT to localize circle applied in intelligent transportation system, *Proc. of 2008 International Conference on Audio, Language and Image Processing*, Shanghai, China, pp.335-338, 2008.

[6] T. D'Orazio, N. Ancona, G. Cicirelli and M. Nitti, A ball detection algorithm for real soccer image sequences, *Proc. of International Conference on Pattern Recognition*, Bari, Italy, vol.16, no.1, pp.210-213, 2002.

[7] N. Qiao, Y. Ye, Y. Huang, L. Liu and Y. Wang, Method of circle detection in PCB optics image based on improved point Hough transform, *Proc. of SPIE – The International Society for Optical Engineering*, Chengdu, China, vol.7283, pp.72832Y-72832Y-5, 2009.

[8] C. Y. Wen, S. H. Chiu, J. J. Liaw and C. P. Lu, The safety helmet detection for ATM's surveillance system via the modified Hough transform, *Proc. of IEEE Annual International Carnahan Conference on Security Technology*, Taipei, Taiwan, pp.364-369, 2003.

[9] A. A. Rad, K. Faez and N. Qaragozlou, Fast circle detection using gradient pair vectors, *Proc. of the 7th International Conference on Digital Image Computing: Techniques and Applications*, Sydney, Australia, pp.879-887, 2003.

[10] F. Wang, Z. Yuan, N. Zheng and Y. Liu, A fast, memory-efficient and parallelizable arc/circle segmentation algorithm, *Applied Mathematics and Computation*, vol.205, no.2, pp.841-848, 2008.

[11] G. Chen, J. Ji and L. Sun, A novel circle detector based on sub-image point pairs, *Proc. of the 2008 IEEE International Conference on Information and Automation*, Zhangjiajie, China, pp.661-666, 2008.

[12] H. S. Kim and J. H. Kim, A two-step circle detection algorithm from the intersecting chords, *Pattern Recognition Letters*, vol.22, no.6-7, pp.787-798, 2001.

[13] J. R. Bergen and H. Shvaytser, A probabilistic algorithm for computing Hough transforms, *Journal of Algorithms*, vol.12, no.4, pp.639-656, 1991.

[14] N. Kiryati, Y. Eldar and A. M. Bruckstein, A probabilistic Hough transform, *Pattern Recognition*, vol.24, no.4, pp.303-316, 1991.

[15] L. Xu, E. Oja and P. Kultanen, A new curve detection method: Randomized Hough transform (RHT), *Pattern Recognition Letters*, vol.11, no.5, pp.331-338, 1990.

[16] L. Xu and E. Oja, Randomized Hough transform (RHT): Basic mechanisms, algorithms, and computational complexities, *CVGIP: Image Understanding*, vol.57, no.2, pp.131-154, 1993.

[17] S. Y. Guo, X. F. Zhang and F. Zhang, Adaptive randomized Hough transform for circle detection using moving window, *Proc. of the 2006 International Conference on Machine Learning and Cybernetics*, Dalian, China, pp.3880-3885, 2006.

[18] T. C. Chen and K. L. Chung, An efficient randomized algorithm for detecting circles, *Computer Vision and Image Understanding*, vol.83, no.2, pp.172-191, 2001.

[19] F. Shang, J. Liu, X. Zhang and D. Tian, An improved circle detection method based on right triangles inscribed in a circle, *2009 World Congress on Computer Science and Information Engineering*, Los Angeles, CA, USA, pp.382-387, 2009.

[20] L. Y. Jiang, Fast detection of multi-circle with randomized Hough transform, *Optoelectronics Letters*, vol.5, no.1, pp.0397-0400, 2009.

[21] S. H. Chiu and J. J. Liaw, An effective voting method for circle detection, *Pattern Recognition Letters*, vol.26, no.2, pp.121-133, 2005.

[22] S. H. Chiu, J. J. Liaw and K. H. Lin, A fast randomized Hough transform for circle/circular arc recognition, *International Journal of Pattern Recognition and Artificial Intelligence*, vol.24, no.3, pp.457-474, 2010.

[23] M. Zhang and H. Cao, A new method of circle's center and radius detection in image processing, *Proc. of the IEEE International Conference on Automation and Logistics*, Qingdao, China, pp.2239-2242, 2008.

[24] R. E. Walpole, R. H. Myers, S. L. Myers and K. Ye, *Probability and Statistics for Engineers and Scientists*, 8th Edition, Pearson Prentice Hall, 2007.