Pattern Recognition 43 (2010) 2659–2665

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Fast randomized algorithm for center-detection

Kuo-Liang Chung [a,1], Yong-Huai Huang [b,*,2], Jyun-Pin Wang [a], Ting-Chin Chang [a], Hong-Yuan Mark Liao [c,3]

[a] Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei, Taiwan 10672, Republic of China

[b] Institute of Computer and Communication Engineering and Department of Electronic Engineering, Jinwen University of Science and Technology, No. 99, An-Chung Road, Hsin-Tien, Taipei 23154, Taiwan, Republic of China

[c] Institute of Information Science, Academia Sinica, No. 128, Academia Road, Section 2, Taipei, Taiwan 115, Republic of China

ARTICLE INFO

Article history:
Received 6 March 2009
Received in revised form
17 February 2010
Accepted 10 March 2010

Keywords:
Center-detection
Hough transform
Gradient line
Multiple centers
Randomized algorithm

ABSTRACT

Recently, Cauchie et al. presented an adaptive Hough transform-based algorithm to successfully solve the center-detection problem which is an important issue in many real-world problems. This paper presents a fast randomized algorithm to solve the same problem. With similar memory requirement and accuracy, the computational complexity analysis and comparison show that our proposed algorithm performs much better in terms of efficiency. We have tested our algorithm on 13 real images. Experimental results indicated that our algorithm has 38% execution-time improvement over Cauchie et al.'s algorithm. The extension of the proposed algorithm to detect multiple centers is also addressed.

1. Introduction

In the field of pattern recognition, line-detection, circle-detection, and ellipse-detection [1–4] have been extensively studied in the past decades. These issues are still important because they are all fundamental issues in pattern recognition. Besides the detection of the above three geometric features, center-detection is also an important research issue. It can be applied to many real-world problems, such as detecting centers from digital images of Debye–scherrer rings, a spiral galaxy, a solar eclipse, a car wheel, a concentric circle, an X-ray diffraction pattern, and a cell. The detected centers can provide important information to astronomers, car designers, medical doctors, and so on.

Based on the Hough transform technique proposed in [5,6], Dammer et al. [7] presented an efficient algorithm to determine the center from digitized X-ray diffraction patterns. Recently, a number of algorithms have been proposed to detect the center of pupils from facial images [8–10]. In [11], Wong and Yip presented a motion information-based algorithm for detecting the center of circulating and spiraling objects from video sequences. Note that the above mentioned algorithms work well in some special cases, such as X-ray images, facial images, and video sequences. To perform center-detection from general still images, Cauchie et al. [12] presented an efficient algorithm which uses the adaptive Hough transform and the coarse-to-fine approach together.

In this paper, we propose a fast randomized center-detection algorithm. With similar memory requirement and detected centers as in [12], the computational complexity analysis indicates that our algorithm significantly outperforms Cauchie et al.'s algorithm in terms of efficiency. Using the same set of test images as in [12] and several other test images, our algorithm significantly saves 38% computation on time on average when compared with Cauchie et al.'s algorithm. Precisely speaking, the improvement ratio is always in the approximative range 20–50%. To show the powerfulness of our algorithm, we also extend the proposed algorithm to detect multiple centers.

The rest of this paper is organized as follows. Cauchie et al.'s algorithm and its computational complexity analysis will be introduced in Section 2. The proposed algorithm and its computational complexity analysis will be presented in Section 3. Experimental results for detecting single and multiple centers will be shown in Section 4. Concluding remarks will be drawn in Section 5.

* Corresponding author. Tel.: +886 2 82122000x6818.
E-mail address: yonghuai@ms28.hinet.net (Y.-H. Huang).

[1] Supported by the National Science Council of R.O.C. under the contract NSC98-2923-E-011-001-MY3.

[2] Supported by the National Science Council of R.O.C. under the contract NSC99-2218-E-228-001.

[3] This work was completed when visiting Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology from March 2009 to August 2009.

0031-3203/$ - see front matter
doi:10.1016/j.patcog.2010.03.010

2. The past work by Cauchie et al. and its complexity analysis

In Cauchie et al.'s center-detection algorithm [12], for the input image I_0, the edge pixels are first detected by using Canny's edge detector [13]. At the same time, a 2-D array A whose size is much smaller than that of the input image is used as the accumulator array. For each edge pixel, its gradient direction is used to draw a digital gradient line L on I_0 which is realized by A where each entry of A, say $A(i,j)$, represents a subimage, i.e. a block. If the digital gradient line L crosses the block covered by $A(i,j)$, we add the gradient magnitude of the edge pixel to the accumulated gradient magnitude (AGM) of $A(i,j)$.

After performing the above AGM-accumulation process for gradient lines of all edge pixels, we select the set $A(i',j')$'s when each AGM value of $A(i',j')$'s is larger than a pre-defined threshold T. Further, we determine the connected components for these $A(i',j')$'s. For each connected component, we find a smallest rectangle to cover it and compute its mean AGM. The rectangle that has the maximum mean AGM is considered as the area with the highest probability containing the center. New center-detection task changes from finding the center (X_c,Y_c) in the original input image I_0 to finding the center (X_c,Y_c) in the subimage I_1 covered by the rectangle with the maximum mean AGM. The array A is used as the accumulator array again for finding the center in I_1. Generally speaking, in the k th iteration, the cardinality of voting set considered in the accumulator array A is not greater than that in the $(k-1)$th iteration. On the other hand, the precision of the detected center considered in the k th iteration is higher than that in the $(k-1)$th iteration. We continue the above coarse-to-fine center-finding process until a satisfactory precision is achieved. Cauchie et al.'s iterative algorithm consists of the following six steps:

Step 1. Given an input image I_0 with size $W_{I_0} \times H_{I_0}$, run Canny's edge detector [13] on I to obtain the edge map with m edge pixels; insert these m edge pixels into the linked-list data structure L_E. The variable k is set to 0 and it is used to indicate the iteration round.

Step 2. The $N \times N$ array A is used as the accumulator array to realize the center-finding task in the subimage I_k where $N \ll W_{I_k},H_{I_k}$. All entries in A are initialized to 0's. Corresponding to A, all entries in B with size $N \times N$ are initialized to 0's, and B will be used in Step 4.

Step 3. For each edge pixel E_i in L_E, its corresponding gradient line is generated. If the generated gradient line of E_i does not cross the image I_k, remove E_i from the list L_E; otherwise, using Bresenham's line-drawing algorithm [14], the gradient line is drawn along the digital line on I_k. For each $A(i,j)$ crossed by the gradient line, we add the gradient magnitude of the relevant gradient line to the AGM of $A(i,j)$.

Step 4. If the AGM of $A(i,j)$ is larger than T, $1 \le i,j \le N$, where T is a specified threshold, we set $B(i,j)$ to 1. Further, we apply the connected component operator [15] to B. For each connected component C_i with 1's, $1 \le i \le c$, where c is the number of the connected components, we determine the smallest rectangle R_i containing C_i. For each R_i, $1 \le i \le c$, calculate the mean AGM of $A(i^*,j^*)$'s included in R_i. Select the rectangle R from all R_i's such that R has the maximal mean AGM.

Step 5. Perform $k=k+1$. If k is larger than the allowable iteration round T_k, go to Step 6. Otherwise, remove p edge pixels from L_E if these p edge pixels are also in L_D. Let I_k denote the subimage which is bounded by the rectangle R. Go to Step 2.

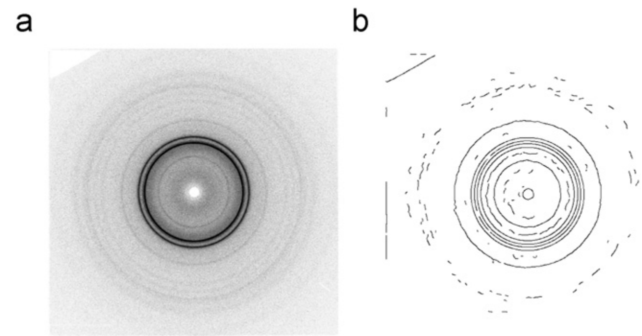

Fig. 1. (a) Debye-scherrer rings and (b) Edge map of Debye-scherrer rings.

Step 6. The final detected center (X_c,Y_c) for the input image I_0 is determined by calculating the center of the rectangle R.

We now provide the time and the storage complexity analysis for Cauchie et al.'s algorithm. Since Canny's edge detector in Step 1 is a preprocessing step, we do not consider it in the complexity analysis. In fact, besides the Canny's edge detector, any edge detectors can be adopted, for example, both the Sobel edge detector and the Prewitt edge detector are good for executing edge detection. The definition of the complexity notation O is suggested to cite the book [16]. The memory complexity of Cauchie et al.'s algorithm is bounded by $M_{Cauchie} = O(m)+O(N^2)$ where $O(m)$ denotes the memory size required in the link-list L_E; $O(N^2)$ denotes the memory size required in the accumulator array A and the binary array B. Because of $m \gg N^2$, the total memory requirement $M_{Cauchie}$ can therefore be simplified to $M_{Cauchie}=O(m)$. For example, the number of edge pixels in the edge map of Fig. 1(a) is $m=6808$. According to the source code of Cauchie et al.'s algorithm [17], we have $N^2=25$, so the condition $m \gg N^2$ does hold. It is easy to verify that Step 2 takes $O(N^2)$ time to initialize the accumulator array A; Step 3 takes $O(m \times N)$ time since for each edge pixel, it needs $O(N)$ time to perform Bresenham's line-drawing algorithm on A and accumulate the gradient magnitudes to the AGM of each touched $A(i,j)$. For Step 4, it takes $O(N^2)$ time to determine whether $B(i,j)$'s are set to 1 or not and it needs $O(N^2)$ time to perform the connected component operation. Step 5 only takes $O(1)$ time to determine a bounding box to construct the subimage I_k. Since Steps 2–6 are performed T_k times, the time complexity of Cauchie et al.'s algorithm is bounded by $T_{Cauchie} = O(T_k) \times [O(N^2)+O(m \times N)] = O(T_k) \times O(m \times N)$.

Proposition 1. *Cauchie et al.'s center-detection algorithm takes $O(m)$ memory and $O(T_k \times m \times N)$ time, where T_k denotes the allowable iteration round; m denotes the number of edge pixels; N denotes the width of the used accumulator array.*

3. The proposed efficient center-detection algorithm

Based on the randomized approach, a new two-stage center-detection algorithm is presented in this section. With similar memory requirement and accuracy of detected centers, the proposed algorithm can achieve a much better computation-saving performance. The proposed algorithm consists of three stages: (1) determining the candidate center first, (2) determining the true center, and then (3) refining the true center.

Based on the concept of randomization, we can randomly select a few edge pixels to determine a candidate center. Previously, McLaughlin proposed a randomized hough transform-based approach [18] for detecting ellipses. In his algorithm, three edge pixels are randomly selected to determine the center and radius of a possible ellipse. Among the selected edge pixels E_1, E_2, and E_3, the midpoint of E_1 and E_2, say m_1, and the intersection point of tangent lines of E_1 and E_2, say t_1, are calculated to determine a line $\overleftrightarrow{t_1 m_1}$. Furthermore, E_2 and E_3 are used to determine the second line $\overleftrightarrow{t_2 m_2}$ by the same way, and then the center of a possible ellipse can be determined by the intersection point of $\overleftrightarrow{t_1 m_1}$ and $\overleftrightarrow{t_2 m_2}$. However, the computation of $\overleftrightarrow{t_1 m_1}$ and $\overleftrightarrow{t_2 m_2}$, as well as their intersection point is computationally expensive. To speed up the process, we propose a new candidate center determination strategy in the following.

The first stage of our strategy is to determine the candidate center from an edge map. First, Canny edge detector is applied to the input gray image and the set of all obtained m edge pixels is denoted by E. Let $E_i \in E$ denote the i th edge pixel in E for $1 \le i \le m$. For each $E_i = (x_i, y_i)$, the gradient direction g_i and the gradient magnitude M_i are calculated by

$$g_i = \frac{G_i^y}{G_i^x} \quad \text{and} \quad M_i = \sqrt{(G_i^x)^2 + (G_i^y)^2}$$

where G_i^x and G_i^y denote the gradients in x-direction and y-direction, respectively.

In the edge set E, we randomly select three edge pixels with different gradient directions, E_i, E_j, and E_k, for $1 \le i,j,k \le m$, to construct three gradient lines L_i, L_j, and L_k by

$$L_i : y - g_i x - y_i + g_i x_i = 0, \tag{1a}$$

$$L_j : y - g_j x - y_j + g_j x_j = 0, \quad \text{and} \tag{1b}$$

$$L_k : y - g_k x - y_k + g_k x_k = 0. \tag{1c}$$

Considering the two gradient lines L_i and L_j, their intersection is calculated by

$$P_{i,j} = \left(\frac{y_i - y_j - g_i x_i + g_j x_j}{g_j - g_i}, \frac{g_j y_i - g_i y_j - g_i g_j x_i + g_i g_j x_j}{g_j - g_i} \right). \tag{2}$$

The distance between L_k and $P_{i,j} = (x^*, y^*)$ is calculated by

$$D(P_{i,j}, L_k) = \frac{|y^* - g_k x^* - y_k + g_k x_k|}{\sqrt{g_k^2 + 1}}. \tag{3}$$

If $D(P_{i,j}, L_k) < \varepsilon_1$, where ε_1 is a specified threshold and is set to 1 empirically, then point $P_{i,j}$ and line L_k should be very close to each other. For this case, $P_{i,j}$ is selected as the candidate center from the randomization sense. Otherwise, if $D(P_{i,j}, L_k) > \varepsilon_1$, we discard $P_{i,j}$. We continue the above randomized candidate-center determination process until ℓ candidate centers are found.

After finishing the first stage, ℓ candidate centers i.e., $P_1, P_2, \ldots,$ and P_ℓ, are determined. In the second stage, we define a 1-D accumulator array A_{near} such that $A_{near}[i]$ records the number of gradient lines which are close to P_i for $1 \le i \le \ell$. Similar to Eq. (3), the distance between the candidate center $P_i = (x_i^*, y_i^*)$, $1 \le i \le \ell$, and the gradient line L_j, $1 \le j \le m$, is calculated by

$$D(P_i, L_j) = \frac{|y_i^* - g_j x_i^* - y_j + g_j x_j|}{\sqrt{g_j^2 + 1}}. \tag{4}$$

If $D(P_i, L_j) < \varepsilon_2$, where ε_2 is a specified threshold and is set to 10 empirically, then point P_i and line L_j should be very close to each other and $A_{near}[i]$ is incremented by one. That is, the number of votes in the cell $A_{near}[i]$ is incremented by one.

After performing the above voting process for all P_i's and L_j's, $1 \le i \le \ell$ and $1 \le j \le m$, a threshold T_c used to eliminate those candidate centers with lower probability is defined as follows:

$$T_c = \frac{\alpha \times N_V}{\ell}, \tag{5}$$

where N_V denotes the number of gradient lines that are close to at least one candidate center and α is set to 1.5 empirically. If $A_{near}[i] < T_c$, we discard P_i from the set of candidate centers since it is a candidate center that has very low probability and it hinders itself to become a true center; otherwise, we put P_i to the set P' whose members have high probability to be a true center. From P', a true center P_T is determined by

$$P_T = \frac{\sum_{P_i \in P'} P_i \times W_{P_i}}{\sum_{P_i \in P'} W_{P_i}}, \tag{6}$$

where

$$W_{P_i} = \frac{A_{near}[P_i]}{\sum_{P_j \in P'} A_{near}[P_j]}, \quad P_i \in P',$$

Because of adopting the randomized concept, the true center P_T may be biased. We further present a refinement scheme to enhance the detected center accuracy. Given a specified bandwidth Δ, we build up a $(\Delta+1) \times (\Delta+1)$ small squared subimage I' centered with P_T, and then we run Cauchie et al.'s center-detection method on I' to refine the detected center. In order to eliminate unnecessary edge pixels E_j's whose gradient lines L_j's are out of the subimage I', by Eq. (4), when $D(P^T, L_j) \le \sqrt{2 \times \Delta^2}$ is held, we add E_j into the edge set E'; otherwise, the edge pixel E_j and the gradient line L_j are not considered in the refinement process. Let the cardinality of the final E' be $|E'| = m'$, $m' \le m$. From the subimage I' and the set E', Cauchie et al.'s center-detection algorithm is adopted to refine the center P^T to obtain a better center P^R. Since the subimage I' is much smaller than the original image I_0, the allowable iteration round T_k' used here is smaller than that originally used in Cauchie et al.'s center-detection algorithm.

Based on the above three-stage strategy, our proposed center-detection algorithm consists of the following seven steps:

Step 1. Given an input image I, apply Canny's edge detector to I to obtain the edge map with m edge pixels and insert these m edge pixels into the set E. The set P will be used to store theses ℓ candidate centers to be determined and it is initialized to an empty set. Here ℓ is set to 10 empirically. Initialize the variable K to be 0 and it is used to indicate the number of the detected candidate centers.

Step 2. Randomly select three edge pixels with different gradient directions, E_i, E_j, and E_k, for $1 \le i,j,k \le m$, from E to construct three gradient lines L_i, L_j, and L_k by Eqs. (1a)–(1c). By Eqs. (2) and (3), calculate the intersection point $P_{i,j}$ between L_i and L_j, and the distance $D(P_{i,j}, L_k)$ between L_k and $P_{i,j}$. If $D(P_{i,j}, L_k) < \varepsilon_1$, insert the intersection point $P_{i,j}$ into P and perform $K = K+1$; otherwise, discard $P_{i,j}$.

Step 3. If $K = \ell$, initialize the array A_{near} by setting $A_{near}[i] = 0$ for $1 \le i \le \ell$, and then go to Step 4; otherwise, go to Step 2.

Step 4. For each edge pixel E_j, $1 \le j \le m$, Eq. (4) is applied to calculate the distance $D(P_i, L_j)$ between each candidate center P_i in P, $1 \le i \le \ell$, and the gradient line L_j of the edge pixel E_j. If $D(P_i, L_j) < \varepsilon_2$, perform $A_{near}[i] = A_{near}[i] + 1$.

Step 5. Initialize the set P' to be an empty set. Apply Eq. (5) to calculate T_c. For $1 \le i \le \ell$, if $A_{near}[i] < T_c$, we discard P_i since it is the candidate center with lower probability to be a true center; otherwise, P_i is added to the set P' to be considered to determine the true center.

Step 6. Based on the candidate centers in the set P', we determine the true center P_T by Eq. (6). Based on P_T and a specified bandwidth Δ, we construct a $(\Delta+1) \times (\Delta+1)$ squared subimage I' in order to refine the accuracy of the detected center. Calculate the distance $D(P^T, L_j)$ between the gradient line L_j of each edge pixel E_j in E and P_T. If the condition $D(P^T, L_j) \le \sqrt{2 \times \Delta^2}$ is held, we add E_j into the set E'; otherwise the unnecessary edge pixel E_j is discarded.

Step 7. Based on I' and E', perform Cauchie et al.'s center-detection algorithm with smaller allowable iteration round T_k' to determine the refined center P^R.

We now analyze the time and the memory complexities of our proposed two-stage algorithm. The memory complexity of the proposed algorithm is denoted by $M_{ours} = O(m) + O(m') + O(\ell)$ where $O(m)$ and $O(m')$ is the memory required to save the sets E and E'; $O(\ell)$ is the memory required for realizing A_{near}, P, and P'. Due to the fact that $m \gg \ell$ and $m' \le m$, the memory complexity is simplified to $M_{ours} = O(m)$. Steps 2 and 3 take $O(\ell \times N_c)$ time to determine ℓ candidate centers where each candidate center can be determined after randomly selecting $3 \times N_c$ pixels. Step 4 takes $O(\ell \times m)$ time to calculate all $D(P_i, L_j)$'s. Steps 5–6 take $O(\ell)$ time to determine the true center and $O(m)$ time to build up the set E'. Step 7 takes $O(T_k' \times m' \times N)$ time for performing Cauchie et al.'s center-detection algorithm. Thus, the total time complexity of our proposed algorithm is denoted by $T_{ours} = O(\ell \times N_c) + O(\ell \times m) + O(\ell) + O(T_k' \times m' \times N) = O(\ell \times m + T_k' \times m' \times N)$ due to $m \gg N_c$.

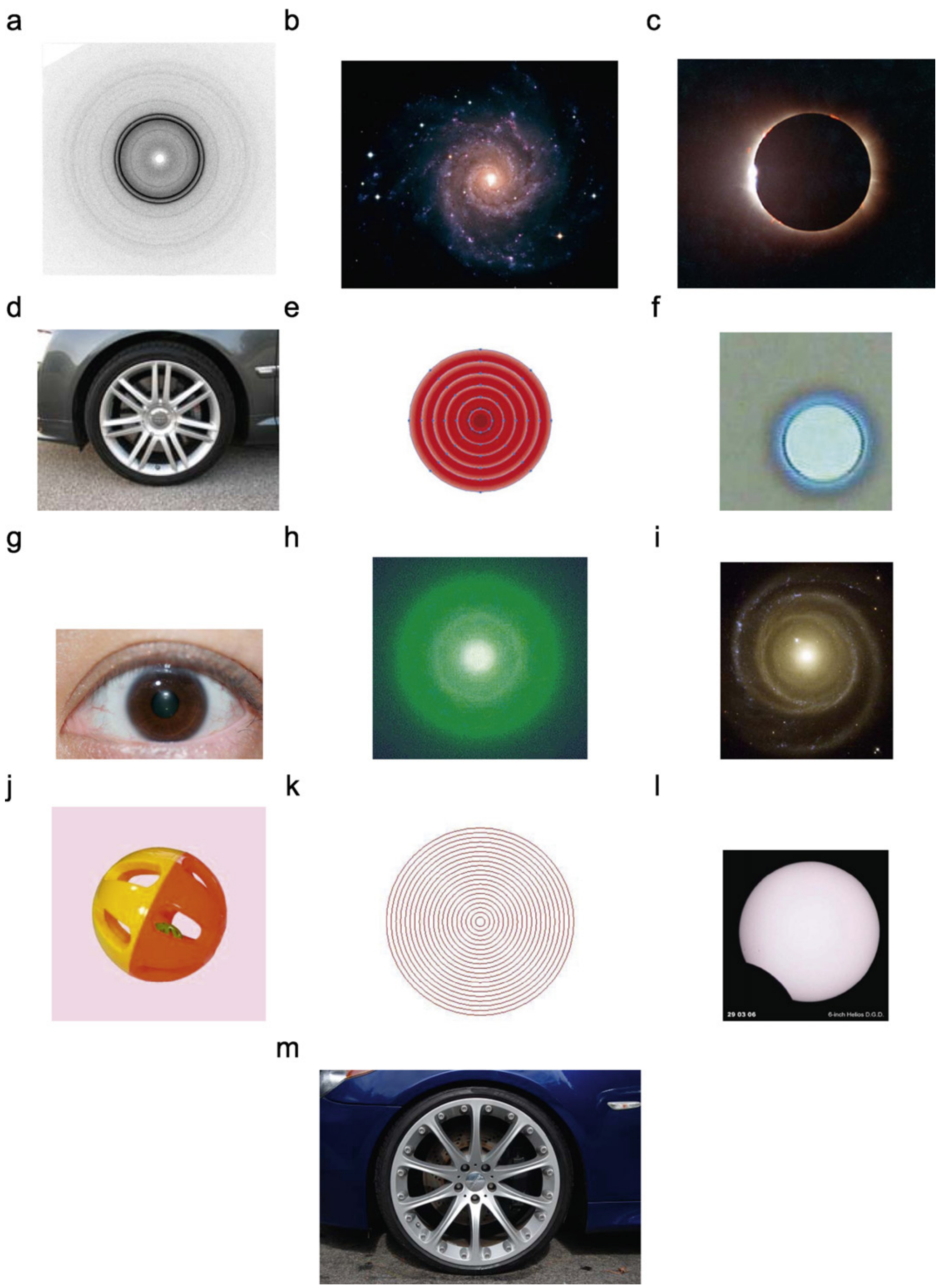

Fig. 2. Thirteen test images.

Proposition 2. *Our proposed algorithm takes* $O(m)$ *memory and* $O(\ell \times m + T_k' \times m' \times N)$ *time, where* $m \gg \ell$.

We further discuss how the proposed algorithm can be extended to handle multiple centers. For the set P' determined by Step 5 in our proposed algorithm, the clustering process [19] is adopted to separate P' into two subsets P_1' and P_2' with the center P_{T1} and the center P_{T2}, respectively. Further, we calculate the distance between P_{T1} and P_{T2}. If the distance is smaller than a specific threshold T_m, it indicates that the input image has only one center; otherwise, the input image contains at least two centers. To determine the maximal number of centers existed in the image, the above clustering process is performed again to separate P' into three subsets P_1', P_2', and P_3' with centers P_{T1}, P_{T2}, and P_{T3}, respectively. Computing the distance between two centers, if one or more distances are smaller than T_m, it indicates that the input image contains two centers and the two centers determined in the previous clustering process are used as the final detected centers. Otherwise, we try to examine the possibility of four centers until the maximal number of centers has been determined.

4. Experimental results

In this section, some experimental results are demonstrated to show the performance comparison between Cauchie et al.'s

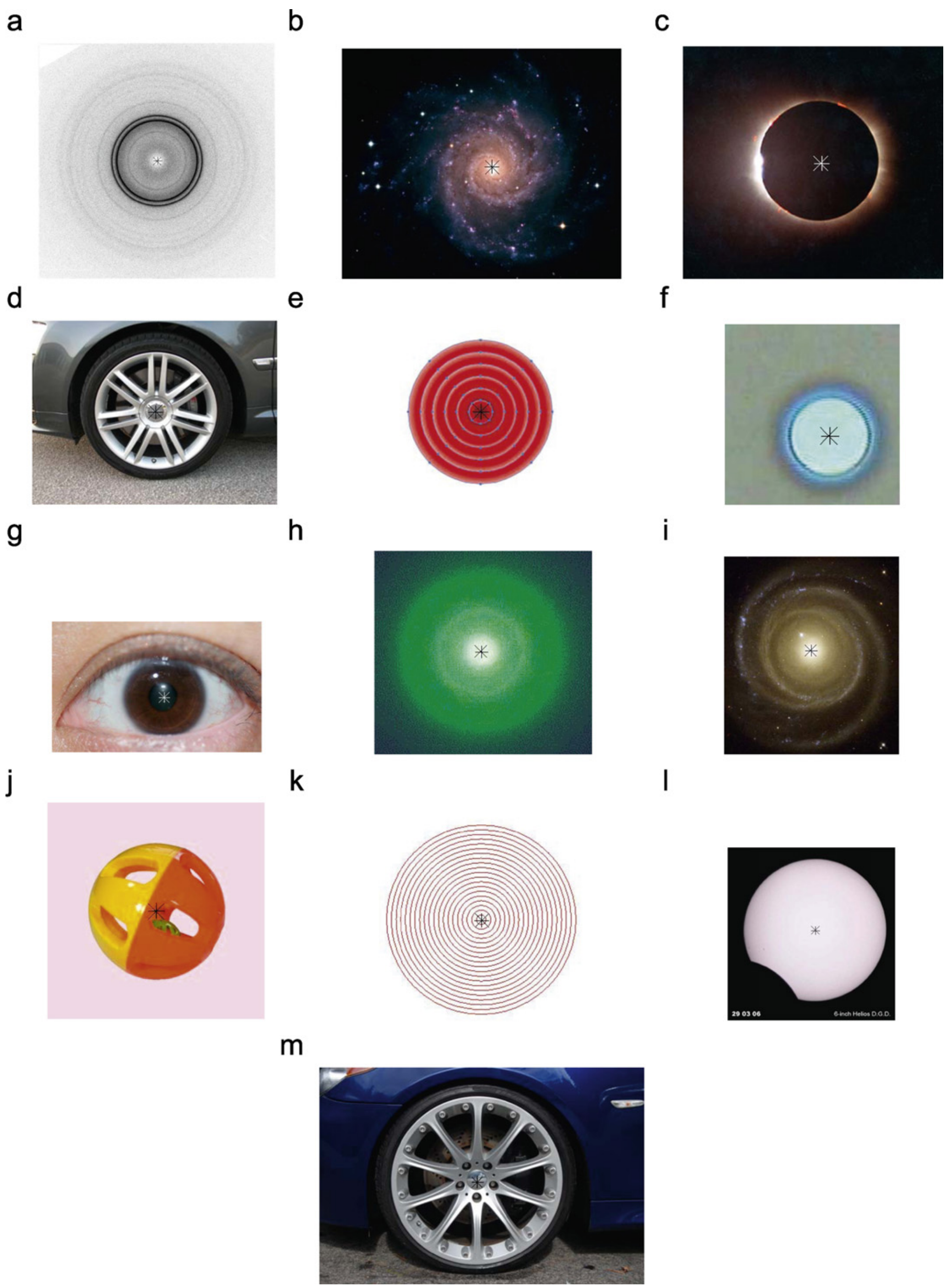

Fig. 3. Detected centers for Fig. 3. The marks '+' and '×' denote centers detected by Cauchie et al.'s algorithm and the proposed algorithm, respectively.

Table 1
Execution-time comparison in terms of millisecond.

	Cauchie et al.'s algorithm	Our proposed algorithm	Execution-time improvement ratio
Fig. 2(a)	11.8	6.4	46% $\left(=\frac{11.8-6.4}{11.8}\times 100\%\right)$
Fig. 2(b)	2.7	2.1	22% $\left(=\frac{2.7-2.1}{2.7}\times 100\%\right)$
Fig. 2(c)	1.9	1.4	26% $\left(=\frac{1.9-1.4}{1.9}\times 100\%\right)$
Fig. 2(d)	3.7	2.7	27% $\left(=\frac{3.7-2.7}{3.7}\times 100\%\right)$
Fig. 2(e)	4.8	3.5	29% $\left(=\frac{4.8-3.5}{4.8}\times 100\%\right)$
Fig. 2(f)	1.3	1.0	23% $\left(=\frac{1.3-1.0}{1.3}\times 100\%\right)$
Fig. 2(g)	1.5	1.1	27% $\left(=\frac{1.5-1.1}{1.5}\times 100\%\right)$
Fig. 2(h)	2.3	1.3	43% $\left(=\frac{2.3-1.3}{2.3}\times 100\%\right)$
Fig. 2(i)	4.4	3.3	25% $\left(=\frac{4.4-3.3}{4.4}\times 100\%\right)$
Fig. 2(j)	3.0	2.1	30% $\left(=\frac{3.0-2.1}{3.0}\times 100\%\right)$
Fig. 2(k)	19.1	10.5	45% $\left(=\frac{19.1-10.5}{19.1}\times 100\%\right)$
Fig. 2(l)	2.9	1.6	45% $\left(=\frac{2.9-1.6}{2.9}\times 100\%\right)$
Fig. 2(m)	20.3	12.5	38% $\left(=\frac{20.3-12.5}{20.3}\times 100\%\right)$
Average	6.1	3.8	38% $\left(=\frac{6.1-3.8}{6.1}\times 100\%\right)$

Table 2
Average distance between centers detected by Cauchie et al.'s algorithm and the proposed algorithm for Figs. 2(a)–(m) in terms of the number of pixels.

Figure	(a)	(b)	(c)	(d)	(e)	(f)	(g)
Distance between centers	1.7	2.6	1.9	1.4	1.0	0.9	1.0
Figure	(h)	(i)	(j)	(k)	(l)	(m)	Average
Difference between centers	1.2	0.8	0.6	1.3	1.7	2.5	1.4

algorithm and our proposed randomized algorithm. All the concerned experiments are performed on the Intel Core 2 Quad Q6600 Processor with 2.4 GHz and 2 GB RAM. The operating system adopted is MS-Windows XP and the programming environment is Borland C++ Builder 6.0.

From Cauchie et al.'s source code [17], it is know that $T_k=10$ and $N=5$. In our implementation, ℓ and T_k' are set to 10 and 2, respectively, and we thus have $\ell < T_k \times N$ and $T_k' < T_k$. In our experiments, the above parameter settings can achieve satisfactory precision. Proposition 1 and 2 clearly indicate the computation-saving effect of our proposed algorithm with $O(\ell \times m + T_k' \times m' \times N)$ time which is much faster than that in Cauchie et al.'s algorithm whose time bound is $O(T_k \times m \times N)$.

As shown in Figs. 2(a)–(m), 13 test images with sizes 590×588, 400×330, 350×312, 350×260, 228×228, 141×146, 400×250, 350×328, 300×341, 270×270, 388×288, 400×413, and 1549×1037, respectively, are used to compare the computation time requirement of the two algorithms. After running the two concerned algorithms on thirteen test images, the detected centers are shown in Fig. 3. It is observed that the detected centers by the two algorithms are quite similar. The execution-time requirement needed in the two algorithms is shown in Table 1 and it indicates that the execution-time improvement ratio of our proposed algorithm over the Cauchie et al.'s algorithm is 38% on average. The improvement ratio always ranges from 20% to 50% depending on the images taken. Further, to demonstrate the accuracy of the proposed algorithm,

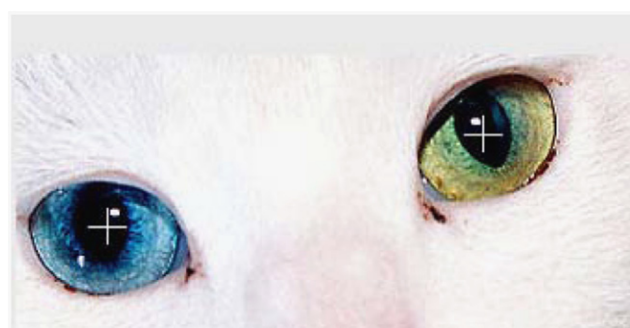

Fig. 4. Detected centers for the image with two centers.

Fig. 5. Detected centers for the image with three centers.

Table 2 shows the average distance between the center detected by Cauchie et al.'s algorithm and the center detected by our proposed algorithm for Figs. 2(a)–(m) in terms of the number of pixels. It is observed that the accuracy of the proposed algorithm is very close to that of Cauchie et al.'s algorithm.

In order to demonstrate the applicability of the proposed multiple-center algorithm mentioned at the end of Section 3, two test images, one with size 337×154 and two centers; the other with size 720×450 and three centers, are used. The detected results are illustrated in Figs. 4 and 5; the resultant centers are satisfactory.

5. Conclusions

We have proposed a two-stage randomized algorithm to speed up the center-detection task significantly while keeping the similar detection accuracy and memory requirement when compared with Cauchie et al.'s algorithm. Under 13 test images, experimental results demonstrated that the average execution-time improvement ratio of the proposed algorithm over the Cauchie et al.'s algorithm is 38%. We further extend the proposed algorithm to detect multiple centers and experimental results illustrated that it also works well for images with multiple centers.

References

[1] C.T. Ho, L.H. Chen, A fast ellipse/circle detector using geometric symmetry, Pattern Recognition 28 (1) (1995) 117–124.
[2] C.T. Ho, L.H. Chen, A high-speed algorithm for elliptical object detection, IEEE Transactions on Image Processing 5 (3) (1996) 547–550.
[3] T.C. Chen, K.L. Chung, An efficient randomized algorithm for detecting circles, Computer Vision and Image Understanding 83 (2) (2001) 172–191.
[4] R.C. Lo, W.H. Tsai, Gray-scale Hough transform for thick line detection in gray-scale images, Pattern Recognition 28 (5) (1995) 647–661.
[5] J. Illingworth, J. Kittler, A survey of the Hough transform, Computer Vision Graphics and Image Understanding: Image Processing 44 (1) (1988) 87–116.
[6] V. Leavers, Which Hough transform, Computer Vision Graphics and Image Understanding: Image Processing 58 (2) (1993) 50–64.
[7] C. Dammer, P. Leleux, D. Villers, M. Dosiere, Use of the Hough transform to determine the center of digitized X-ray diffraction patterns, Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms 132 (1) (1997) 214–220.
[8] H.W.S.W.T. Wang, C. Xu, Eye localization based on hue image processing, in: International Symposium on Intelligent Signal Processing and Communication Systems, Xiamen, 2007, pp. 730–733.
[9] N.T.H.B.N. Huan, H. Kim, Location of iris based on circular and linear filters, in: International Conference on Control, Automation, Robotics and Vision, Hanoi, 2008, pp. 412–416.
[10] T.R.B.S.E.-R.V. Kith, M. El-Sharkawy, S. Elnoubi, A feature and appearance based method for eye detection on gray intensity face images, in: International Conference on Computer Engineering and Systems, Hahaii, 2008, pp. 41–47.
[11] K.Y. Wong, C.L. Yip, Identifying centers of circulating and spiraling vector field patterns and its applications, Pattern Recognition 42 (7) (2009) 1371–1387.
[12] J. Cauchie, V. Fioletand, D. Villers, Optimization of an Hough transform algorithm for the search of a center, Pattern Recognition 41 (2) (2008) 567–574.
[13] J. Canny, A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (6) (1986) 679–698.
[14] J.E. Bresenham, Algorithm for computer control of a digital plotter, IBM Systems Journal 4 (1) (1965) 25–30.
[15] R.C. Gonzalez, R.E. Wood, Digital Image Processing, second ed., Prentice Hall, New Jersey, 2002 (Chapter 9).
[16] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, second ed., The MIT Press, Cambridge, Mass, 2001.
[17] ⟨http://staff.umh.ac.be/Villers.Didier/Hough/Hough_Center.html⟩.
[18] R.A. McLaughlin, Randomized hough transform: improved ellipse detection with comparison, Pattern Recognition Letters 19 (3–4) (1998) 299–305.
[19] C. Zhu, L.M. Po, Minimax partial distortion competitive learning for optimal codebook design, IEEE Transactions on Image Processing 7 (10) (1998) 1400–1409.

About the Author—KUO-LIANG CHUNG received his B.S., M.S., and Ph.D. degrees in Computer Science and Information Engineering from the National Taiwan University in 1982, 1984, and 1990, respectively. He was a visiting scholar at the University of Washington in the summer of 1999. During 2003–2006, he was the head of the Department of Computer Science and Information Engineering at the National Taiwan University of Science and Technology. He was promoted to Professor and University Chair Professor in 1995 and 2009, respectively. During 1996–1998, he was the executive editor of Journal of the Chinese Institute of Engineers. Dr. Chung received the Distinguished Engineering Professor Award from Chinese Institute of Engineers in 2001; received the Distinguished Research Award from the National Science Council in 2004; received the Distinguished Scholar Research Award from the National Science Council of Taiwan in 2009, and received the best paper award from the Society of Computer Vision, Graphics, and Image Processing, Taiwan, in 2007. He is now a senior member of IEEE and a fellow of IET. He has authored and coauthored over 130 publications in books and international well-known journals. His research interests include image/video compression, image/video processing, pattern recognition, artificial intelligence, algorithms, and multimedia applications.

About the Author—YONG-HUAI HUANG received the B.S. degree in Information Management from Aletheia University, Tamsui, Taipei, Taiwan, and the M.S. and Ph.D. degrees in Computer Science and Information Engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan. He is now an assistant professor in the Institute of Computer and Communication Engineering at Jinwen University of Science and Technology, Hsin-Tieny, Taipei, Taiwan. His research interests include image processing and compression, and algorithms.

About the Author—JYUN-PIN WANG received the B.S. degree in Applied Mathematics from the National Dong Hwa University, Hualien, Taiwan and the M.S. degrees in Computer Science and Information Engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan. She is now an engineer in the AU Optronics Corporation, Longtan, Taoyuan, Taiwan. Her research interests include image processing and multimedia applications.

About the Author—TING-CHIN CHANG received the B.S. degree in Computer Science and Information Engineering from Yuan Ze University, Jhongli, Taoyuan, Taiwan and the M.S. degrees in Computer Science and Information Engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan. He is now an engineer in the AVerMedia Technologies Inc., Taipei, Taiwan. His research interests include image processing and multimedia applications.

About the Author—HONG-YUAN MARK LIAO received the B.S. degree in physics from National Tsing Hua University, Hsinchu, Taiwan, in 1981 and the M.S. and Ph.D. degrees in electrical engineering from Northwestern University, Evanston, IL, in 1985 and 1990, respectively. He was a Research Associate with the Computer Vision and Image Processing Laboratory, Northwestern University, during 1990–1991. In July 1991, he was with the Institute of Information Science, Academia Sinica, Taipei, Taiwan, as an Assistant Research Fellow. He was promoted to Associate Research Fellow and then Research Fellow in 1995 and 1998, respectively. From August 1997 to July 2000, he was the Deputy Director of the institute. From February 2001 to January 2004, he was the Acting Director of the Institute of Applied Science and Engineering Research. He is jointly appointed as a Professor with the Department of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu. He is the Editor-in-Chief of the Journal of Information Science and Engineering. He is also an Editorial Board Member of the International Journal of Visual Communication and Image Representation, the EURASIP Journal on Advances in Signal Processing, and the Research Letters in Signal Processing. His current research interests include multimedia signal processing, video-based surveillance systems, content-based multimedia retrieval, and multimedia protection. Dr. Liao was the recipient of the Young Investigators' Award from Academia Sinica in 1998, the Excellent Paper Award from the Image Processing and Pattern Recognition Society of Taiwan in 1998 and 2000, the Distinguished Research Award from the National Science Council of Taiwan in 2003, the National Invention Award of Taiwan in 2004, and the Distinguished Scholar Research Award from the National Science Council of Taiwan in 2008. He was an Associate Editor of the IEEE Transactions on Multimedia during 1998–2001. He served as the Program Cochair of the Second IEEE Pacific Rim Conference on Multimedia. He was the Conference Cochair of the Fifth International Conference on Multimedia and Exposition (ICME 2004) in June 2004, the Technical Cochair of ICME 2007, a committee member of the 2005, 2006, and 2007 ACM Multimedia Conferences and the 2007 World Wide Web Conference.