CIRCLE DETECTION BY ARC-SUPPORT LINE SEGMENTS

Changsheng Lu[1], *Siyu Xia*[1], *Wanming Huang*[2], *Ming Shao*[3] *and Yun Fu*[4]

[1] Key Laboratory of Measurement and Control of CSE, Southeast University, Nanjing 210096, China
[2] Joint Stars Technology CO., LTD, Nanjing 211100, China
[3] Computer and Information Science, University of Massachusetts, Dartmouth, MA 02747, USA
[4] Department of ECE, Northeastern University, Boston, MA 02115, USA

ABSTRACT

Circle detection is fundamental in both object detection and high accuracy localization in visual control systems. We propose a novel method for circle detection by analysing and refining arc-support line segments. The key idea is to use line segment detector to extract the arc-support line segments which are likely to make up the circle, instead of all line segments. Each couple of line segments is analyzed to form a valid pair and followed by generating initial circle set. Through the mean shift clustering, the circle candidates are generated and verified based on the geometric attributes of circle edge. Finally, twice circle fitting is applied to increase the accuracy for circle locating and radius measuring. The experimental results demonstrate that the proposed method performs better than other well known approaches on circles that are incomplete, occluded, blurry and over-illumination. Moreover, our method shows significant improvement in accuracy, robustness and efficiency on the industrial Printed Circuit Board (PCB) images as well as the synthesized, natural and complicated images.

Index Terms— Circle detection, arc-support, polarity analysis, line segment, circle fitting

1. INTRODUCTION

Circle detection plays a key role in computer vision and has been widely applied in different practical problems. In PCB industrial automation, defect detection machine needs to recognize circular pads and find their central positions accurately and efficiently. Additionally, circle detection is an indispensable part in iris localization, traffic sign detection and robot vision. Currently, the methods of circle detection are mainly divided into three categories: Circle Hough Transform (CHT) [1–4], Random Sample Consensus (RANSAC) [5, 6], and line segments approximating [7].

CHT is the earliest theory for detecting circular shapes. The basic idea is mapping an arbitrary edge point into a parameter space on the right circular cone surface, which, in effect, equals voting for a 3D accumulator. If there is a circle in the image, the local peak of the parameter space will occur because its corresponding bin of accumulator has acquired the enough votes. However, CHT consumes excessive memory to construct the 3D accumulator and requires a lot of time for voting and local peaks search. Besides, the votes from noisy edge points may be added up and lead to the false peaks. Finally, the parameter, e.g. bin size, significantly affects final accuracy, which is not easy to tune. Even though considerable efforts [8–12] have been proposed to improve the performance, they still hardly reach the industry requirement.

Another type of methods are based on RANSAC, out of which the most representative one is Randomized Circle Detection (RCD). In brief, RCD picks four edge pixels iteratively and randomly. It then uses three of them for generating a hypothetic circle and the rest one for testing. Comparing to CHT, RCD just requires some variables instead of an accumulator, and validates the hypothetic circle rather than accumulating it. As a result, it saves substantial time and memory. However, it also introduces many unnecessary calculations and false detections because of the early decision and randomized edge pixels.

The third type of methods are based on the line segments approximating and circle edge geometric attributes. This new technology gets development by the appearance of line segment (LS for short) detector with false detection control (LSD) [13], where nonstraight curve is being approximated by LSs. On the strength of LSD anti-noise ability and linear-time complexity, we naturally employ LSs to circle detection. Recently, Truc Le el al. use pairs of LSs to calculate the circle centers and radii, then cluster the circles to generate candidates, and finally implement verification by the edge pixels [7]. While this method has achieved excellent performance, it still has three problems. The first one lies in redundant calculation caused by useless LSs such as those actually derived from straight edges, since only LSs with arc-support edges are possible to generate the circle candidates. Supposing that n LSs exist in an image, the number of pairs could be up to

This work is supported by the Fundamental Research Funds for the Central Universities award under Grant 2242017K40120 and the National Natural Science Foundation of China under Grant 61671151.

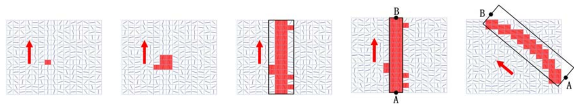
C_n^2, which implies that the efficiency will increase about 4 times when reducing 50% of useless LSs. Secondly, except coordinates of endpoints, a LS also contains information like gradients and possible direction of circle center, which the algorithm fails to utilize. Lastly, the accuracy of LSD still cannot meet the requirement of industry.

The aim of this paper is to present a robust, efficient and accurate circle detection algorithm while avoiding those identified issues from existing methods. We first compute the gradient of an image to acquire the edge image and extract the arc-support LSs. Next, for each pair of two LSs, we analyze to determine whether the pair could generate an initial circle or not, through the polarity analysis, region restriction and criteria of radii and inliers. Then, we cluster the initial circle set to produce the circle candidates followed by validations. Finally, we refine the results by a twice circle fitting for each circle.

2. CIRCLE DETECTION ALGORITHM

2.1. Arc-support line segments

LS mainly derives from two situations. First, LS may come from the support region where points share roughly the same level-line angle (gradient angle rotates clockwise $90°$) and distribute highly straight (Fig. 1(d)). Another situation comes from the arc-support region, whose points' level-line angles and distributions change like a curve (Fig. 1(e)). The second type LS is called as arc-support LS. Extracting the arc-support LS is our main task.

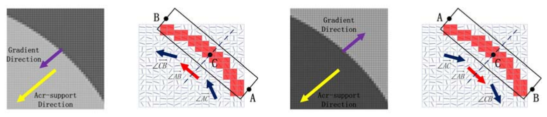


(a) (b) (c) (d) (e)

Fig. 1. LSD and arc-support LS. (a)$\sim$(d) show the process of LSD algorithm [13]. (e) shows the arc-support LS.

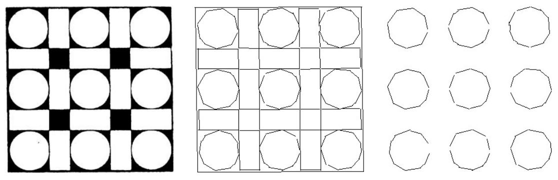
LSD [13] firstly computes the gradient at each pixel and rejects the point whose gradient magnitude is too small. Then, it applies region growing algorithm starting from the seed points. Each aligned point's level-line angle should be up to a certain tolerance with the main orientation of the support region. Each region is a candidate for LS and will be approximated by a rectangle. Next, it conducts refinement and validation steps by computing the number of false alarms (NFA) of each rectangle based on a contrario approach and Helmholtz principle [14, 15], as shown in Fig. 1(a)$\sim$(d). The smaller NFA value of a rectangle means a higher possibility of being a LS.

Unlike LSD that detects all the LSs, our methods only detects the arc-support LSs. Assume the two terminals of the regional approximated rectangle are A and B, and the main angle of the region is denoted as $\angle\overrightarrow{AB}$. $\angle\overrightarrow{AB}$ can be set to the principal inertial axis of the support region. We divide the region into two parts by a dash line passing center point C and perpendicular to $\angle\overrightarrow{AB}$. Repeat running the local PCA, the main angles of two parts $\{\angle\overrightarrow{AC}, \angle\overrightarrow{CB}\}$ can be obtained, as shown in Fig. 2(b) and Fig. 2(d). If $\{\angle\overrightarrow{AC}, \angle\overrightarrow{AB}, \angle\overrightarrow{CB}\}$ change in the same direction and have an angle interval at least T_{ai} in $\{\angle\overrightarrow{AC}, \angle\overrightarrow{AB}\}$ and $\{\angle\overrightarrow{AB}, \angle\overrightarrow{CB}\}$, arc-support LS may be produced from the corresponding support region. Moreover, the states of angle variation of them can be anticlockwise and clockwise. Fig. 2(a) is the original grayscale image of level-line Fig. 2(b), whose overall gradient direction is consistent with its arc-support direction (where the circle may occur). We call that the LS's polarity is positive for this situation. Similarly, if $\{\angle\overrightarrow{AC}, \angle\overrightarrow{AB}, \angle\overrightarrow{CB}\}$ change clockwise as shown in Fig. 2(d), the polarity of LS is negative, as shown in Fig. 2(c).



(a) (b) (c) (d)

Fig. 2. Arc-support LSs and associated polarities. In (a), the gradient direction is same as arc-support direction. (b) shows the three main angles changing anticlockwise associated with (a). (c) and (d) are the counter-examples.



(a) (b) (c)

Fig. 3. Results of LSs extraction. (a) the origin image. (b) 146 LSs are extracted by the method of LSD [13]. (c) 92 arc-support LSs are extracted with our method.

Following our approach, the detected LSs are all generated from the decomposition of curves, as shown in Fig. 3. With this procedure, the arc-support LSs are purer, which significantly reduces the pairs of LSs that generate false candidates.

2.2. Paired line segments analysis

Around the edge of a circumference, it can be found that inside of the circle is always either brighter or darker than the outside where brighter means that polarities of arc-support LSs are positive and darker means the polarities are negative. If arc-support LSs are all generated from a circle, their polarities should be the same. This accords with our observation, especially for the industrial circle objects. If the probability of arc-support LS with positive polarity is equal to the negative, we will reduce the invalid pairs of LSs by half. And in certain cases, we just need to detect the round objects which are brighter (or darker) than background. Our method can also detect them successfully because we only retain the corresponding polarity arc-support LSs for circle detection.

In the set of arc-support LSs, most of pairs will not contribute to calculating the intersection of their midperpendiculars because there is high probability that two arc-support LSs come from different circles or curves. Therefore, an early decision before calculating the circle parameters is essential. In Fig. 4, intuitively $\{$L1, L3$\}$ and $\{$L1, L4$\}$ should not be

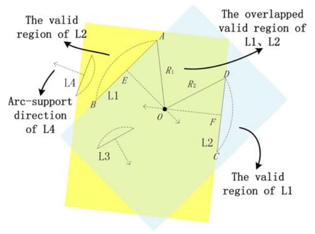


Fig. 4. Region restriction for a pair of two arc-support LSs.

paired. The reason is that L4 is not in the valid region corresponding to the arc-support direction of L1. Though L3 is in the L1's valid region, L1 is out of the valid region of L3. As a result, only L1 and L2 are both in the valid regions, where the regions are overlapping. Therefore, we have:

$$\begin{cases} \overrightarrow{AC} \cdot \overrightarrow{ARC_{L1}} > \rho_d \\ \overrightarrow{AD} \cdot \overrightarrow{ARC_{L1}} > \rho_d \\ \overrightarrow{CA} \cdot \overrightarrow{ARC_{L2}} > \rho_d \\ \overrightarrow{CB} \cdot \overrightarrow{ARC_{L2}} > \rho_d \end{cases}$$

The $\overrightarrow{ARC_{Li}}$ is the normalized arc-support direction vector of Li ($i = 1$ or 2). ρ_d is distance threshold.

The algorithm continues to compute the intersection O of midperpendiculars of each pair $\{$L1, L2$\}$. R_1 and R_2 are the distances from O to the endpoints of L1 and L2. In addition, we add radial criterion that R_1 and R_2 should be within a radial distance tolerance ϵ_{rd} [16]. If it is true, the radius R will be averaged by R_1 and R_2. A circle is verified against the inliers that make up L1 and L2 based on another distance tolerance ϵ_{id} and normal tolerance α (the angle deviation between gradient angle of edge pixel P_j and its normal angle $\overrightarrow{OP_j}$) [7]. Moreover, the percentage of valid inliers should be more than γ.

2.3. Validation &Twice circle fitting

In fact, the initial circles generated from the valid pairs of arc-support LSs, will have many duplicates. We therefore apply non-maximum suppression via mean-shift clustering [17] to remove duplicates for the generation of candidates. Since the circle candidates are coarse, they will be refined individually using the fast least-squares fitting by the circular inliers and verified against the number of support edge pixels and the completeness of circle. The number of valid inliers on circle should have a positive correlation with the radius in the general case [18]. As a result, we use a ratio threshold T_{ni} and expect there are $2\pi RT_{ni}$ edge pixels on the circle. In addition, circle completeness is also an important criterion. So we measure the angular coverage of the circular connected component of the edge pixels and accept only the circle whose completeness is at least T_{ac} degrees [7].

Recall that the circle candidates need to be fitted before validation because the candidate from the associated pair of arc-support LSs only implies a circle appearing in the neighborhood. As a result, the more sufficient support inliers there are, the more accurate the final circle will be. If a candidate after the first circle fitting generates the final circle, the new inliers are more sufficient than the old. This observation motivates us for a twice circle fitting, which improves the accuracy for circle detection in the experiments. Moreover, the time complexity of twice circle fitting only relates to number of the final circles, meaning that the running time will only slightly change.

3. EXPERIMENTS

There are eight parameters involved in our algorithm, as shown in Table 1. T_{ai} is used in the arc-support LSs extraction; ρ_d can be minus in some extreme cases in region restriction; ϵ_{rd}, ϵ_{id}, α and γ are applied in radii and inliers criteria respectively; T_{ni} and T_{ac} are used in circle validation. We have already tuned the parameters and the algorithm works well in plenty of experiments.

Table 1. The involved parameters in our method

Para.	Illustration	Value
T_{ai}	angle interval	$2.25°$
ρ_d	distance threshold	$-1.5\epsilon_{rd}$
ϵ_{rd}	radial distance tolerance	1% of image minimum size
ϵ_{id}	inliers' distance tolerance	$0.75\epsilon_{rd}$
α	normal tolerance	$22.5°$
γ	ratio of support inliers making up the arc-support LSs	0.6
T_{ni}	ratio of support edge pixels on a circle	0.5
T_{ac}	angle of circular completeness	$165°$

We compare our approach[1] with the current three competitive methods: the method in [7] (based on LSs), CHT and R-CD. Particularly, CHT is the optimized and built-in *imfindcircles* from MATLAB with sensitivity as 0.9. For an objective

[1]https://github.com/AlanLuSun/Circle-detection

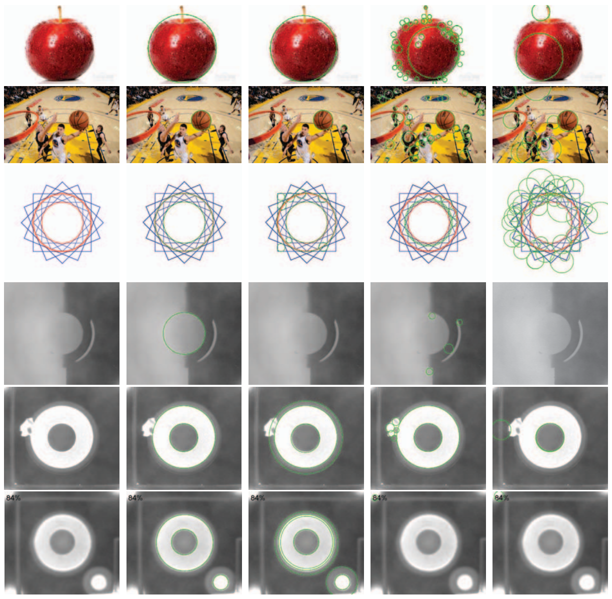
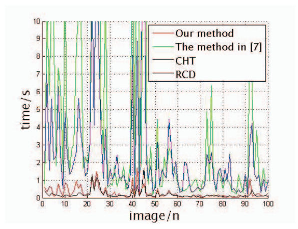


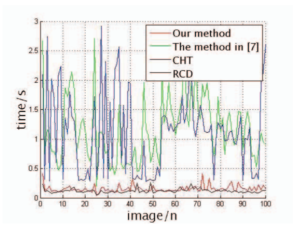
Fig. 5. Circle detection results by different methods. The first column is the original image. From the second column to the fifth column: our method, the method in [7], CHT, RCD.

evaluation, we elaborately prepare two datasets[1]: natural image dataset (NID) and industrial PCB image dataset (PCBID). Each dataset contains 100 images with various disturbances and each image contains at least one circular shape. We label all the images manually. All the experiments are implemented by MATLAB on a computer with the Intel Core i7-5500U CPU 2.4GHz and 8 GB memory.

Fig. 5 illustrates some examples from two datasets. These images are subject to disturbances such as complicated background, brightness and shadow, discontinuous circle, blurry edge, occlusions, etc. These factors greatly challenge the circle detection algorithm. As we can see, CHT and RCD can detect some ground true circles (TPs) in the image, but also cause false positives (FPs) and fail to detect the ground true circles, namely false negatives (FNs). In complicated backgrounds, tremendous noisy edge points fool CHT to accumulate the false peaks, and also lead RCD to make a lot of useless attempts for pixel selections. Moreover, CHT and RCD are hard to detect the circles when they are blurry. In contrast to CHT and RCD, the method in [7] is more robust and reduces a lot of FPs due to the false control ability of LSD [13]. However, when the occluded objects and complicated backgrounds occur, it also generate some FPs. On the contrary, our method is immune to those factors. Thanks to the good features of arc-support LSs, our method shows strong anti-noisy ability. In addition, the proposed twice circle fitting also help us further improve the performance of accuracy. The detected circle by our method can fit more accurately and globally while avoiding fitting locally, as shown in the first row of Fig. 5.



(a)



(b)

Fig. 6. Dynamic running time curves of four circle detection algorithms in (a) the natural image dataset and (b) the PCB image dataset.

Table 2. Results in the natural image dataset

Method type	Precision	Recall	Average time
Our method	97.26%	81.45%	284.6 ms
The method in [7]	86.40%	82.60%	4467.8 ms
CHT	26.36%	61.95%	2457.7 ms
RCD	31.06%	34.99%	190.2 ms

Table 3. Results in the industrial PCB image dataset

Method type	Precision	Recall	Average time
Our method	100.00%	94.24%	155.3 ms
The method in [7]	89.06%	97.12%	1160.0 ms
CHT	35.53%	55.56%	1106.9 ms
RCD	52.27%	18.93%	118.3 ms

For convincing comparisons, we follow the performance measurements proposed in [19, 20] to evaluate the four circle detection algorithms. The precision = $\frac{TPs}{TPs+FPs}$ and recall = $\frac{TPs}{TPs+FNs}$. In the dynamic running time curves in Fig. 6, our method and RCD spend much less time and are more stable than CHT and the method in [7]. In Table 2 and Table 3, our algorithm and the method in [7] shine over the CHT and RCD in the precision and recall. Although the method in [7] is comparable with ours in recall, our method has a much better precision and much less running time. This is mainly because our method generates lower FPs. Surprisingly, our algorithm dose not generate any FPs in the PCBID experiment, which might be the result of careful LS pair selection and strict analysis in purifying the candidates.

4. CONCLUSIONS

We proposed a robust, precise and efficient circle detection method based on arc-support LSs. The arc-support LSs carry rich information such as the whole gradient direction and polarity, which allow us to analyse each pair of arc-support LSs rigorously and implement region restriction to reduce the pairing time significantly. The twice circle fitting improves the accuracy to industry level that meets the requirement of PCB, which casts a light of applying our method in real-world industry problems.

5. REFERENCES

[1] Paul C. Hough, V, "Method and means for recognizing complex patterns," 1962.

[2] Richard O. Duda, "Use of the hough transformation to detect lines and curves in pictures," in *Commun. ACM*, 1972, pp. 11–15.

[3] Carolyn Kimme, "Finding circles by an array of accumulators," *Communications of the Acm*, vol. 18, no. 2, pp. 120–122, 1975.

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

[5] Teh-Chuan Chen and Kuo-Liang Chung, "An efficient randomized algorithm for detecting circles," *Computer Vision and Image Understanding*, vol. 83, no. 2, pp. 172–191, 2001.

[6] Li-qin Jia, Cheng-zhang Peng, Hong-Min Liu, and Zhi-Heng Wang, "A fast randomized circle detection algorithm," in *Image and Signal Processing (CISP), 2011 4th International Congress on*. IEEE, 2011, vol. 2, pp. 820–823.

[7] Truc Le and Ye Duan, "Circle detection on images by line segment and circle completeness," in *IEEE International Conference on Image Processing*, 2016, pp. 3648–3652.

[8] HK Yuen, John Princen, John Illingworth, and Josef Kittler, "Comparative study of hough transform methods for circle finding," *Image and vision computing*, vol. 8, no. 1, pp. 71–77, 1990.

[9] Lei Xu, Erkki Oja, and Pekka Kultanen, "A new curve detection method: randomized hough transform (rht)," *Pattern recognition letters*, vol. 11, no. 5, pp. 331–338, 1990.

[10] Nahum Kiryati, Yuval Eldar, and Alfred M Bruckstein, "A probabilistic hough transform," *Pattern recognition*, vol. 24, no. 4, pp. 303–316, 1991.

[11] Raymond KK Yip, Peter KS Tam, and Dennis NK Leung, "Modification of hough transform for circles and ellipses detection using a 2-dimensional array," *Pattern recognition*, vol. 25, no. 9, pp. 1007–1022, 1992.

[12] Huashan Ye, Guocan Shang, Lina Wang, and Min Zheng, "A new method based on hough transform for quick line and circle detection," in *Biomedical Engineering and Informatics (BMEI), 2015 8th International Conference on*. IEEE, 2015, pp. 52–56.

[13] von Gioi R Grompone, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall, "Lsd: a fast line segment detector with a false detection control.," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 4, pp. 722–732, 2010.

[14] Agnès Desolneux, Lionel Moisan, and Jean-Michel Morel, "Meaningful alignments," *International Journal of Computer Vision*, vol. 40, no. 1, pp. 7–23, 2000.

[15] Agnès Desolneux, Lionel Moisan, and Jean-Michel Morel, *From Gestalt theory to image analysis. A probabilistic approach*, Springer New York, 2008.

[16] Heung-Soo Kim and Jong-Hwan Kim, "A two-step circle detection algorithm from the intersecting chords," *Pattern recognition letters*, vol. 22, no. 6, pp. 787–798, 2001.

[17] Yizong Cheng, "Mean shift, mode seeking, and clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[18] Zenon Kulpa, "On the properties of discrete circles, rings, and disks," *Computer graphics and image processing*, vol. 10, no. 4, pp. 348–365, 1979.

[19] D Powers, "Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation," *Sch. Informatics Eng. Flinders*, 2007.

[20] David L Olson and Dursun Delen, *Advanced data mining techniques*, Springer Science & Business Media, 2008.