CIRCLE VALIDATION USING RANDOMIZED ALGORITHM FOR DETECTING CIRCLES

Preprocessing steps:

- Read an image and convert into gray image [*fig. 1*]
- Now apply canny edge detection algorithm on gray image [*fig. 2*]
- Now calculate the image pixels' density if it is beyond given density threshold then return the null image else follow the preprocessing steps
- Remove the unwanted pixels which are not part of circle outline by nulling the pixels from center of image of given percentage of square area which will increase the speed of execution of randomized circle detection [*fig. 3*]

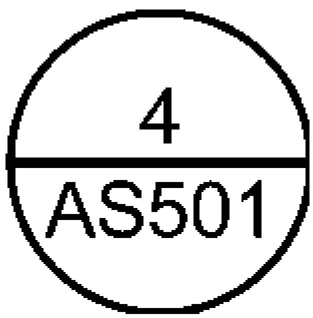


Fig. 1 gray image

Fig. 2 canny edge image

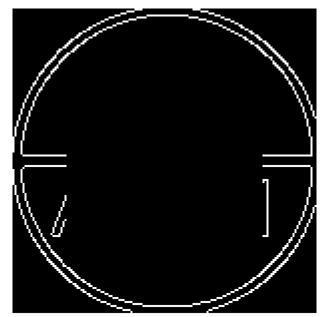


Fig. 3 image after nulling pixels from center of image

Randomized circle detection theory:

The equation of circle is defined as

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

Where a, b are center coordinates of circle

r is radius of circle

let us consider $(x_1,\ y_1)$, $(x_2,\ y_2)$, $(x_3,\ y_3)$ are three non-collinear points on the circle then their circle equations can be written as follows

$$2\mathrm{a}x_1 + 2\mathrm{b}y_1 + d = x_1^2 + y_1^2$$

$$2\mathrm{a}x_2 + 2\mathrm{b}y_2 + d = x_2^2 + y_2^2$$

$$2\mathrm{a}x_3 + 2\mathrm{b}y_3 + d = x_3^2 + y_3^2$$

Where d can written be as $(r^2 - a^2 - b^2)$

A representation of the above three equations in terms of matrix form yields

$$\begin{pmatrix} 2x_1 & 2y_1 & 1 \\ 2y_2 & 2y_2 & 1 \\ 2x_3 & 2y_3 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ d \end{pmatrix} = \begin{pmatrix} x_1^2 + y_1^2 \\ x_2^2 + y_2^2 \\ x_3^2 + y_3^2 \end{pmatrix}$$

Applying Gaussian elimination, we have,

$$\begin{pmatrix} 2x_1 & 2y_1 & 1 \\ 2(x_2 - x_1) & 2(y_2 - y_1) & 0 \\ 2(x_3 - x_1) & 2(y_3 - y_1) & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ d \end{pmatrix} = \begin{pmatrix} x_1^2 + y_1^2 \\ x_2^2 + y_2^2 - (x_1^2 + y_1^2) \\ x_3^2 + y_3^2 - (x_1^2 + y_1^2) \end{pmatrix}$$

By Cramer's rule, the center can be obtained by

$$a = \frac{\begin{vmatrix} x_2^2 + y_2^2 - (x_1^2 + y_1^2) & 2(y_2 - y_1) \\ x_3^2 + y_3^2 - (x_1^2 + y_1^2) & 2(y_3 - y_1) \end{vmatrix}}{4((x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1))}$$

$$b = \frac{\begin{vmatrix} 2(x_2 - x_1) & x_2^2 + y_2^2 - (x_1^2 + y_1^2) \\ 2(x_3 - x_1) & x_3^2 + y_3^2 - (x_1^2 + y_1^2) \end{vmatrix}}{4((x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1))}$$

$$r = \sqrt{(x_i - a)^2 + (y_i - b)^2}$$

for any i = 1,2,3

Randomized circle detection algorithm:

Step-1:

Store all the edge pixels $\boldsymbol{v_i = (x_i, y_i)}$ to the list ***V*** and divide the ***V*** into four lists $\boldsymbol{v_1, v_2, v_3, v_4}$ depending on their quadrants so that picking a pixel for each quadrant increases the probability of edge pixels of the image are on the circle and non-collinearity. Initialize the failure count ***f*** to be 0. Let $\boldsymbol{T_f}$, $\boldsymbol{T_{min}}$, $\boldsymbol{T_a}$, $\boldsymbol{T_d}$, and $\boldsymbol{T_r}$ be the five given thresholds. Here, $\boldsymbol{T_f}$ denotes the number of failures that we can tolerate. If there are less than $\boldsymbol{T_{min}}$ pixels in ***V***, we stop the task of circle detection. The distance between any two agent pixels of the possible circle should be larger than $\boldsymbol{T_a}$, $\boldsymbol{T_d}$ and $\boldsymbol{T_r}$ are the distance threshold and ratio threshold, respectively.

Step-2:

If $\boldsymbol{f} = \boldsymbol{T_f}$ then stop; otherwise randomly pick four pixels one from each $\boldsymbol{v_1, v_2, v_3, v_4}$. when $\boldsymbol{p_j}$ has been choosen, list $\boldsymbol{v_i} = \boldsymbol{v_i} - \{\boldsymbol{p_j}\}$ **,** i = 1,2,3,4 & j<$|\boldsymbol{v_i}|$

Step-3:

From the four edge pixels, find out the possible circle such that the distance between any two of the three agent pixels is larger than $\boldsymbol{T_a}$ and the distance between the fourth pixel and the boundary of the possible circle is larger than $\boldsymbol{T_d}$**;** go to Step-4. Otherwise put $\boldsymbol{p_j}$ back to the corresponding quadrant edge pixels list; perform ***f = f + 1;*** go to Step-2.

Step-4:

Assume $\boldsymbol{C_{ijk}}$ is the possible circle. Set the counter ***C*** to be 0. For each $\boldsymbol{p_i}$ in ***V***, we check whether $\boldsymbol{d_{l \to ijk}}$ is not larger than the given distance threshold $\boldsymbol{T_d}$. If yes, ***C = C + 1***. After examining all the edge pixels in ***V***, assume C = n_p, i.e., there are n_p edge pixels satisfying $\boldsymbol{d_{l \to ijk}} \boldsymbol{<=} \boldsymbol{T_d}$.

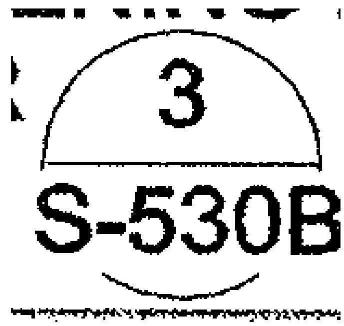
Step-5:
If n_p >= $2\pi r_{ijk} T_r$ go to Step 6. Otherwise, regard the possible circle as a false circle, perform ***f = f+1***; and go to Step-2.

Step-6:
The possible circle C_{ijk} has been detected as a true circle. So return the given is a valid circle.

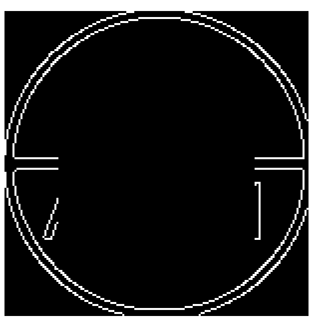
Experimental results:

Many experiments are carried out to validate high speed and resolution of proposed algorithm. All experiments are performed on an intel i5 2.60GHZ computer using C# language. In this experiment thresholds T_f, T_{min}, T_a, and T_r are set to be 2000, 30, 15 and 1. Minimum image density and maximum image density thresholds are 0.01 and 0.08. n_p ratio is 0.47. The experimental results are as shown below

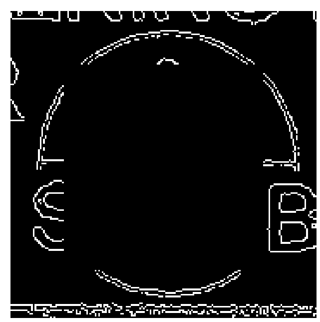
(a) original image - 1



(b) original image - 2



(c) preprocessed image - 1



(d) preprocessed image - 2

(e) detected circle image - 1

(f) detected circle image - 2

References:

[1] Teh-Chuan Chen, Kuo-Liang Chung, "An Efficient Randomized Algorithm for Detecting Circles" *Computer Vision and Image Understanding* 83, 172–191 (2001).

[2] Lianyuan Jiang, Peihe Tang, Haohao Yuan, Chungui Li, Yalan Zhang, "Fast Randomized Algorithm for Circle Detection by Efficient Sampling" *Journal of Theoretical and Applied Information Technology* 20th February 2013. Vol. 48 No.2.

[3] Li-Qin Jia, Hong-Min Liu, Zhi-Heng Wang, Hong Chen, "An Effective Non-HT Circle Detection for Centers and Radii" 2011 International Conference on Machine Learning and Cybernetics, Guilin, 10-13 July, 2011.

[4] Wei Chen, Lichun Sui, Zhengchao Xu,Yu Lang, "Improved Zhang-Suen Thinning Algorithm in Binary Line Drawing Applications" 2012 International Conference on Systems and Informatics (ICSAI 2012).

[5] Lianyuan Jiang, "Efficient randomized Hough transform for circle detection using novel probability sampling and feature points" L. Jiang / Optik 123 (2012) 1834–1840.

[6] Z. Guo and R. W. Hall, "*Parallel thinning with two-subiteration algorithms*" Comm. ACM, vol. 32, no. 3, pp. 359-373, 1989.

[7] Lam, L., Seong-Whan Lee, and Ching Y. Suen, "*Thinning Methodologies-A Comprehensive Survey*" IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 14, No. 9, September 1992, p. 879

[8] Rafael C. Gonzalez, "*Digital Image Processing*" 3rd edition