# BBM418 Computer Vision Lab.
# Assignment 1 - Circle by Using Hough Transform

**Fatma Nur Demirbaş**
21727116
Department of Computer Engineering
Hacettepe University
Ankara, Turkey
b21727116@cs.hacettepe.edu.tr

## Overview

The aim of this assignment is to detect closed or open circles from the given data set using Python3 with the Hough Transform method we wrote.

## 1 Introduction

The Hough transform is a specific feature extraction technique used in digital image processing. The purpose of this technique is to find instances of objects of a particular shape class through a voting procedure. This voting procedure is performed using accumulation arrays. Hough transform can be applied to detect many random shapes in an image.

**Circle detection with Hough Transform:**

Equation of circle with radius r with center M(a,b):

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

In the Hough Transform algorithm, the equation of the circle in the polar coordinate system is used.

$$a = x - r.cos(\theta)$$
$$b = y - r.sin(\theta)$$

**Hough Transform Detection Algorithm**

- All elements of the Accumulator A[a,b,r] array that we created for the Hough Transform space should be initially set to 0. Every A[a,b,r ] = 0
- First of all, filtering operators should be run on the captured scene (image, frame).
- Gaussian Bluring , cvtGrayScale ( graying ) , Canny Edge Detector operators should be applied on the received frame, respectively.
- For each pixel, the center and radius of the circles that may occur should be calculated using the polar coordinate equations mentioned above.
- Each pixel should be voted on the circles it can create with the voting algorithm specified below.
- For the voted pixels, the Hough Transform space can be specified with the pixels with the most votes in Accumulator A and those with more votes than the specified limit.

- The accumulator with the most votes contains the center point, coordinates and radius of the circle to be determined.
- The circle with the most votes is drawn.

## 2 Implementation

Hough transform is a feature extraction technique used in digital image processing. The purpose of this technique is to find desired instances of objects of a particular shape class through a **voting** procedure. This voting procedure is carried out in this project using backlogs.

I used the Python programming language and the OpenCV library to implement this assignment. OpenCV is a library of programming functions mainly aimed at real-time computer vision[1]. OpenCV has given us many more possibilities such as blurring of images (Gaussian Blur), edge detection (Canny).

In this assignment, I chose the **Canny** application for determining the edges. Because the Canny edge detector is an operator that successfully detects edges using a multistage algorithm to detect a wide variety of edges in images[2]. Canny edge detection algorithm consists of 5 steps:

- Noise reduction;
- Gradient calculation;
- Non-maximum suppression;
- Double threshold;
- Edge Tracking by Hysteresis.

Thanks to all these steps, clearly identifying the edges will make it very easy for me to process the next circle search in my project.

Generally in this assignment,the image is softened using a Gaussian Filter for ease in removing shapes where unwanted noise is desired to be removed. The visual effect of this blurring technique is the shadow of an object under normal lighting or the image from a translucent screen that is markedly different from the bokeh effect produced by an out-of-focus lens. Edges are then detected in the image using Canny Edge. This provides the basic outline in the resulting image. At each point on the edge, all possible circles in the Hough space are voted on. The local maximum in the Hough space gives the circles. A threshold is used to determine the qualifier local maximums.. The basic idea in the algorithm is that the center of gravity of that particular radius circle should be where the most hits occur.

The algorithm used to implement Hough Circles in the assignment is as follows:

- Step 1: Threshold assignments and read the image
  **imread()** is in OpenCV is used to read an image from the filesystem.I had to normalize the accumulator array after I had it to get a standard threshold value between 0 and 1. This threshold determines the occlusion ration and, as a result, the circle detection sensitivity. Threshold is 0.55 .

- Step 2: Gaussian Blur the image to reduce noise
  **GaussianBlur()** is in OpenCV is used to blur the image to a specified kernel size. I used Gaussian Blur mainly to blur the image and remove sharp noise in images. This will make the Canny edge detector work correctly.

- Step 3:Edge detection using Canny Edge Detector
  **Canny()** is used to detect edges in the image. Canny edge detector is a very good algorithm

for detecting edges in an image. It has two thresholds assigned to it, in optimization I decided 25 and 150 are ideal thresholds for both.

- Step 4: Find circle candicates
  Different r and theta combinations are kept in the **circle_candidates** array in the given radius range and theta range.

- Step 5:Find and vote the circle from candidate circles passing through a found edge pixel.
  . For every pixel in image
  . For thetas and r in circle_candicates
  . x = x0 – R*cos (theta)
  . y = y0 – R*sin(theta)
  . Accumulator[R][x][y]+=1

- Step 6: Construct circles for every r,x,y according to votes
  Sort the accumulator based on the votes for the candidate circles .First of all, the vote percentages of the candicate circles are calculated and the values that are equal to or greater than the bin_threshold value are determined. Then values close to pixel_threshold are deleted. The remaining values tell us the circle to be drawn.

- Step 7: Draw Circle
  **circle()** is in OpenCV is used to draw circles onto an image at a particular pixel x,y with a radius r.

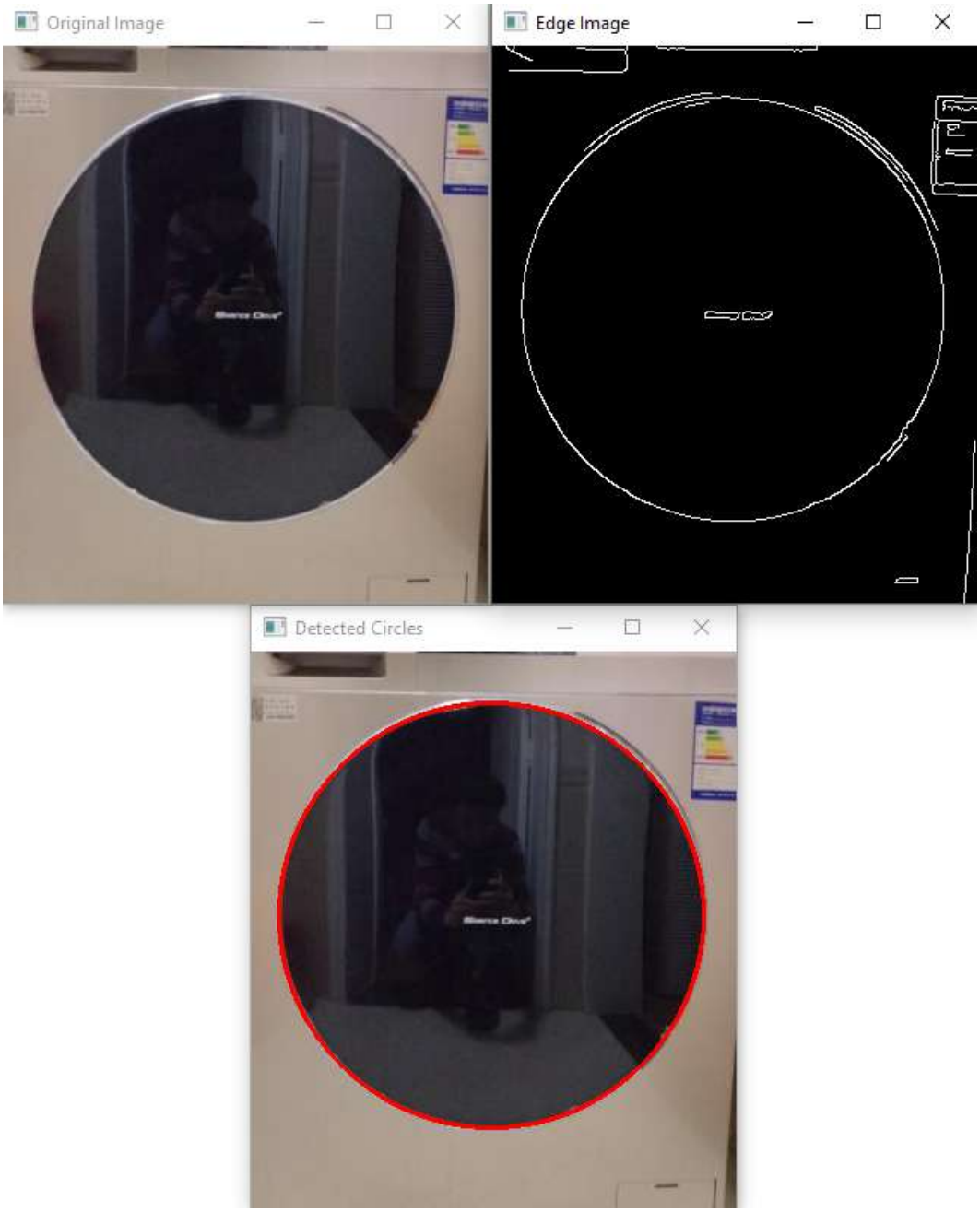
- Step 8: Calculate IOU scores for each circles and average IOU scores for images
  For each circle selected from the Ground Thurut file, and for each circle found, the center distances of the circle areas and the unions of the circles are found. Then, the intersections of the circles are found according to the relationship between the Radius and the distance(d), the IOU Score is calculated. Finally, the average IOU is calculated based on the specific picture.

## 3 Successful Examples

### 3.1 3.jpg

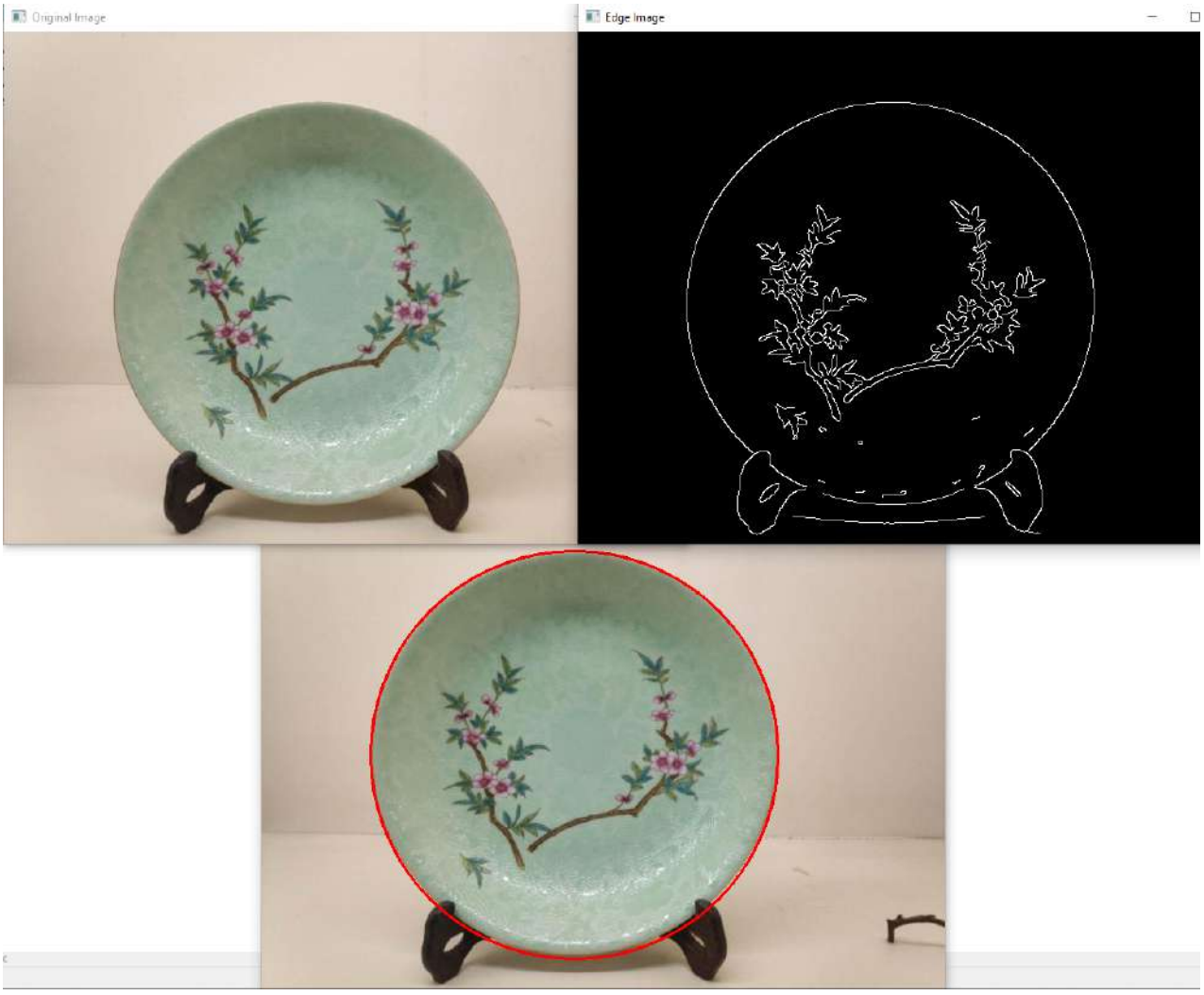

**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.983
Max IOU Score of 2. Circle: 0.821
**Average IOU score for this image: 0.902**

### 3.2 88.jpg




**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.98
**Average IOU score for this image: 0.98**

### 3.3 28.jpg



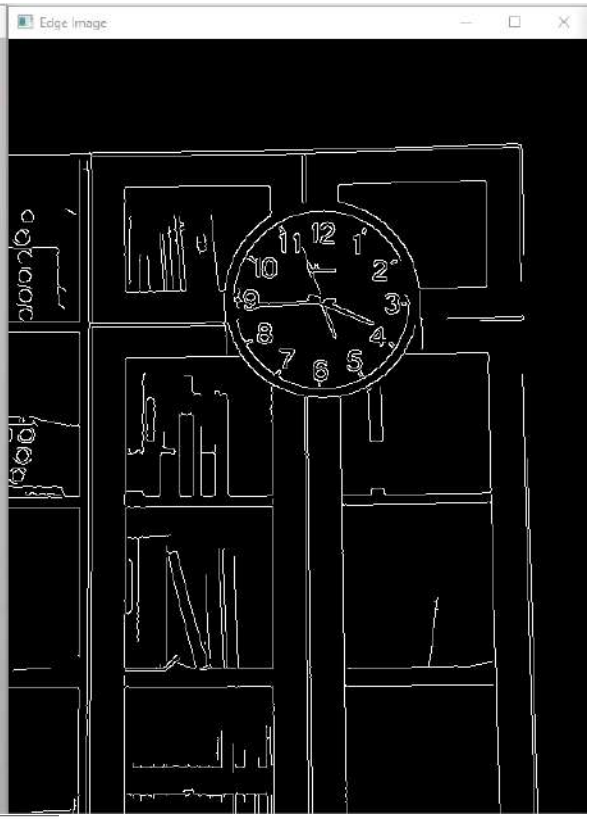

**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.969
**Average IOU score for this image: 0.969**

### 3.4 27.jpg


Original Image
关注安全 有你有我
禁止扒门
禁止踹门
禁止倚靠
禁止使用肢体或物体挡门
禁带危险品
禁止抢上抢下
看好儿童
超载勿上
面向轿厢门进出 注意轿厢位置
如遇火警
请勿乘坐电梯
轻按选层按钮
Edge Image
Detected Circles

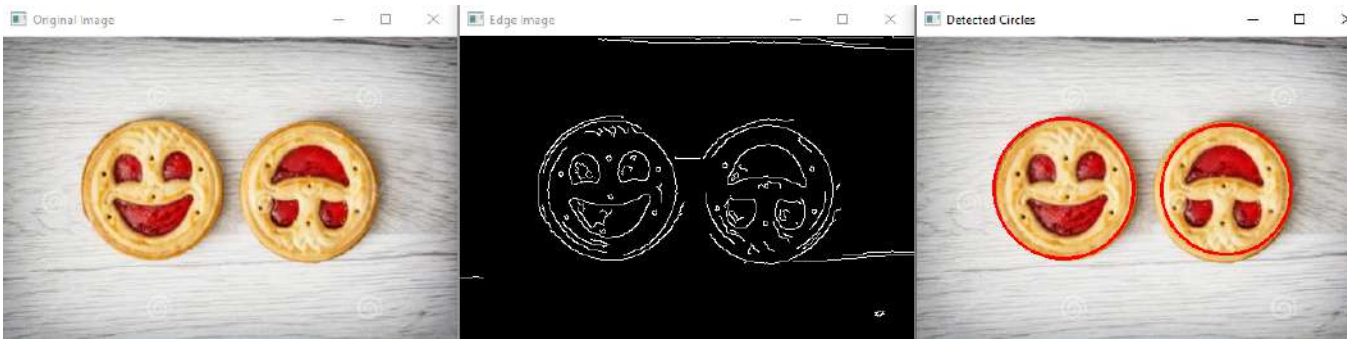
**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.66
Max IOU Score of 2. Circle: 0.886
Max IOU Score of 3. Circle: 0.627
Max IOU Score of 4. Circle: 0.918
Max IOU Score of 5. Circle: 0.654
Max IOU Score of 6. Circle: 0.94
Max IOU Score of 7. Circle: 0.974
Max IOU Score of 8. Circle: 0.694
Max IOU Score of 9. Circle: 0.689
Max IOU Score of 10. Circle: 0.952
Max IOU Score of 11. Circle: 0.962
Max IOU Score of 12. Circle: 0.654
Max IOU Score of 13. Circle: 0.958
Max IOU Score of 14. Circle: 0.654
Max IOU Score of 15. Circle: 0.724
Max IOU Score of 16. Circle: 0.924
Max IOU Score of 17. Circle: 0.724
Max IOU Score of 18. Circle: 0.909
Max IOU Score of 19. Circle: 0.968
Max IOU Score of 20. Circle: 0.633
**Average IOU score for this image: 0.805**

### 3.5 49.jpg

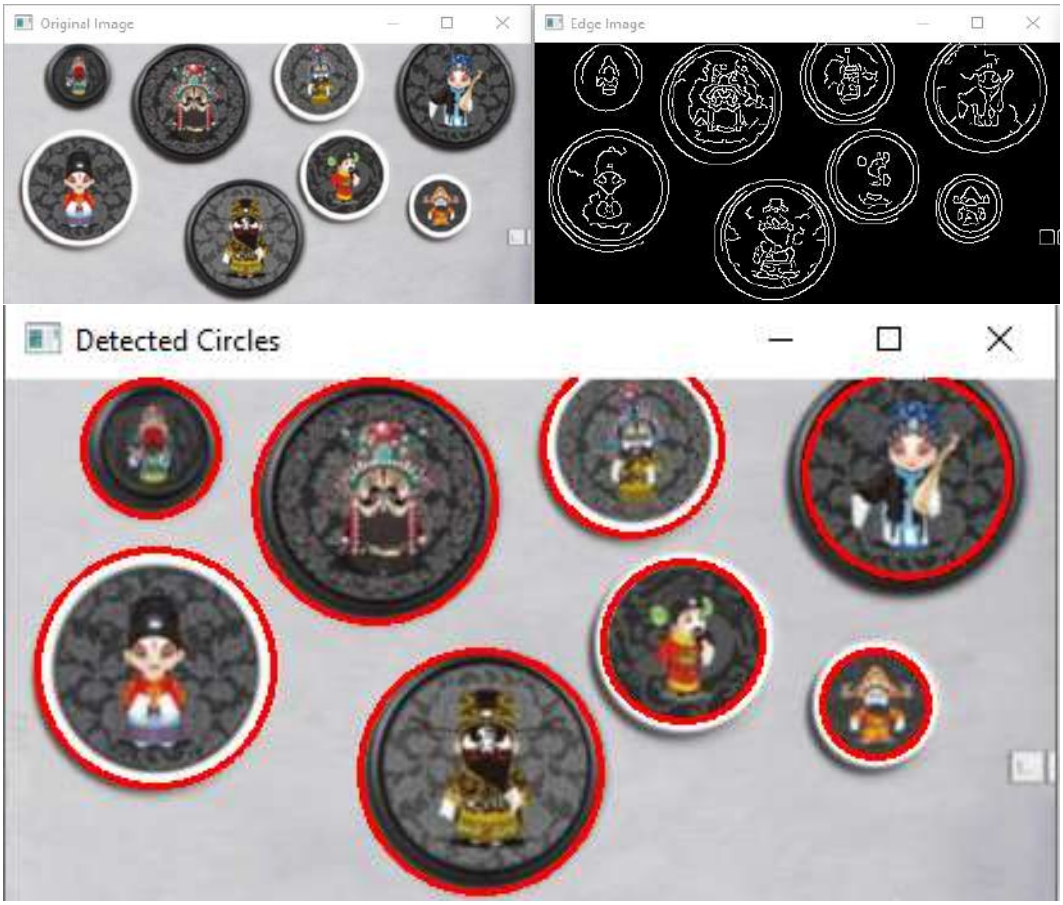

**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.85
Max IOU Score of 2. Circle: 0.952
**Average IOU score for this image: 0.901**

### 3.6 68.jpg




**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.973
Max IOU Score of 2. Circle: 0.855
**Average IOU score for this image: 0.914**

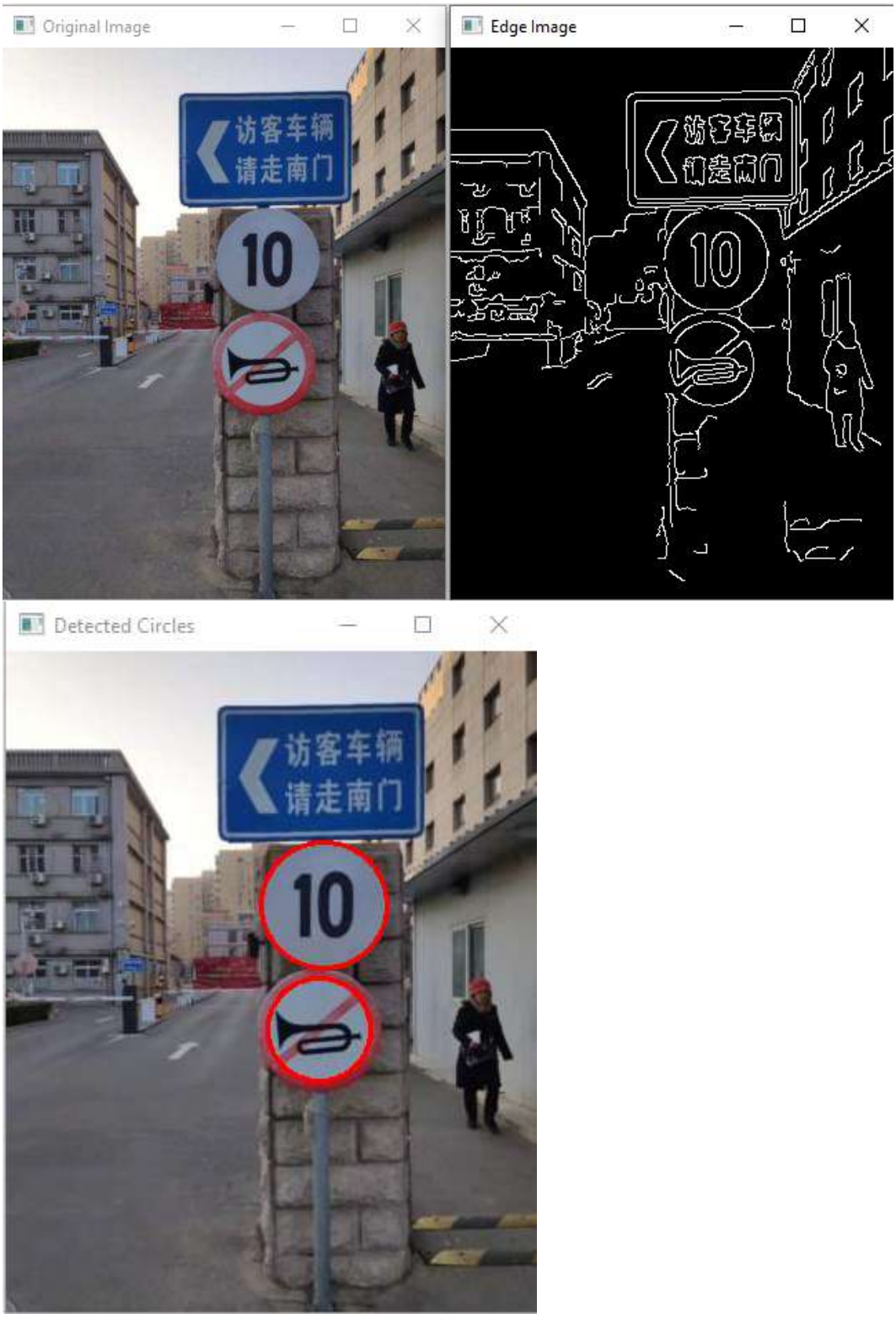
### 3.7 69.jpg




**IOU scores of this file :**

Max IOU Score of 1. Circle: 0.944
Max IOU Score of 2. Circle: 0.706
Max IOU Score of 3. Circle: 0.92
Max IOU Score of 4. Circle: 0.74
Max IOU Score of 5. Circle: 0.952
Max IOU Score of 6. Circle: 0.702
Max IOU Score of 7. Circle: 0.766
Max IOU Score of 8. Circle: 0.941
Max IOU Score of 9. Circle: 0.664
Max IOU Score of 10. Circle: 0.911
Max IOU Score of 11. Circle: 0.795
Max IOU Score of 12. Circle: 0.882
Max IOU Score of 13. Circle: 0.96
Max IOU Score of 14. Circle: 0.735
Max IOU Score of 15. Circle: 0.901
Max IOU Score of 16. Circle: 0.562
**Average IOU score for this image: 0.922**

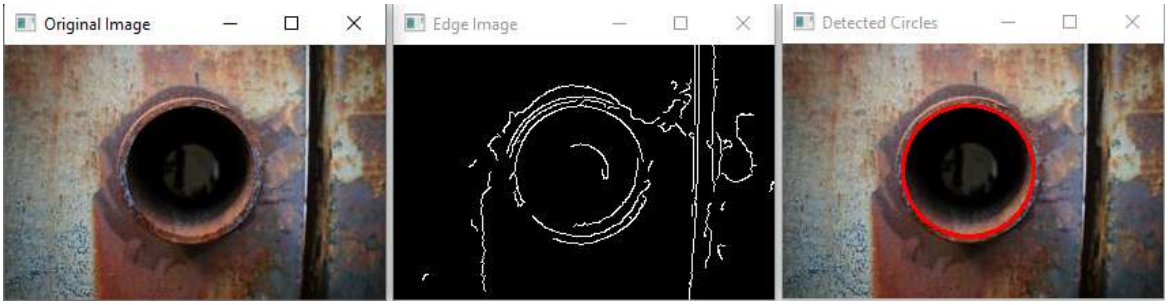
### 3.8 85.jpg




**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.898

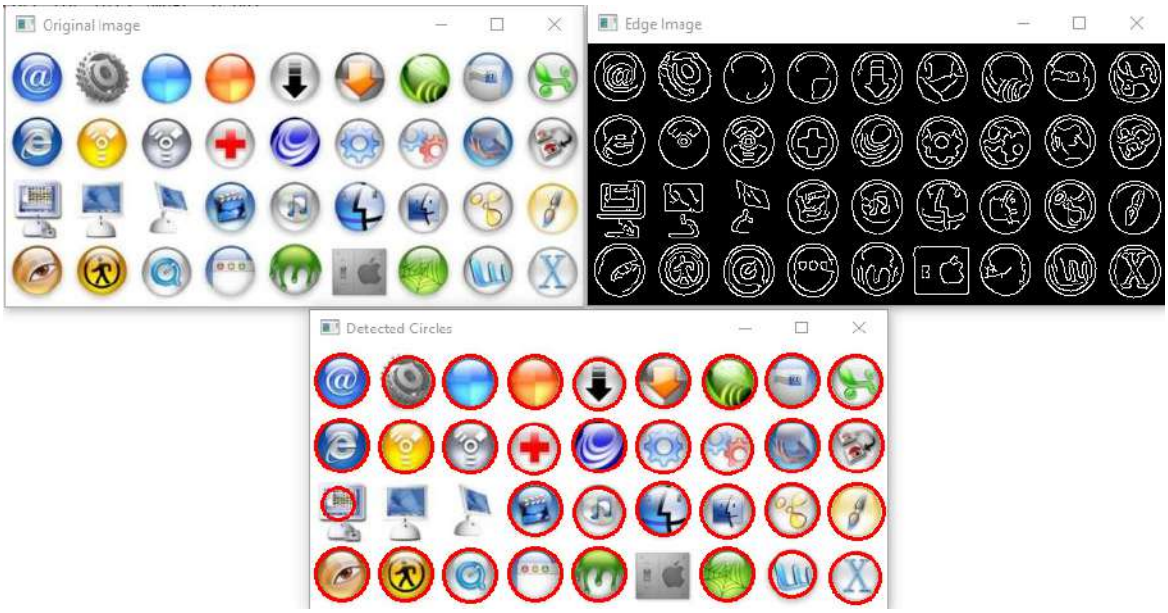
Max IOU Score of 2. Circle: 0.742
**Average IOU score for this image: 0.82**

### 3.9 100.jpg




**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.79
Max IOU Score of 2. Circle: 0.908
**Average IOU score for this image: 0.849**

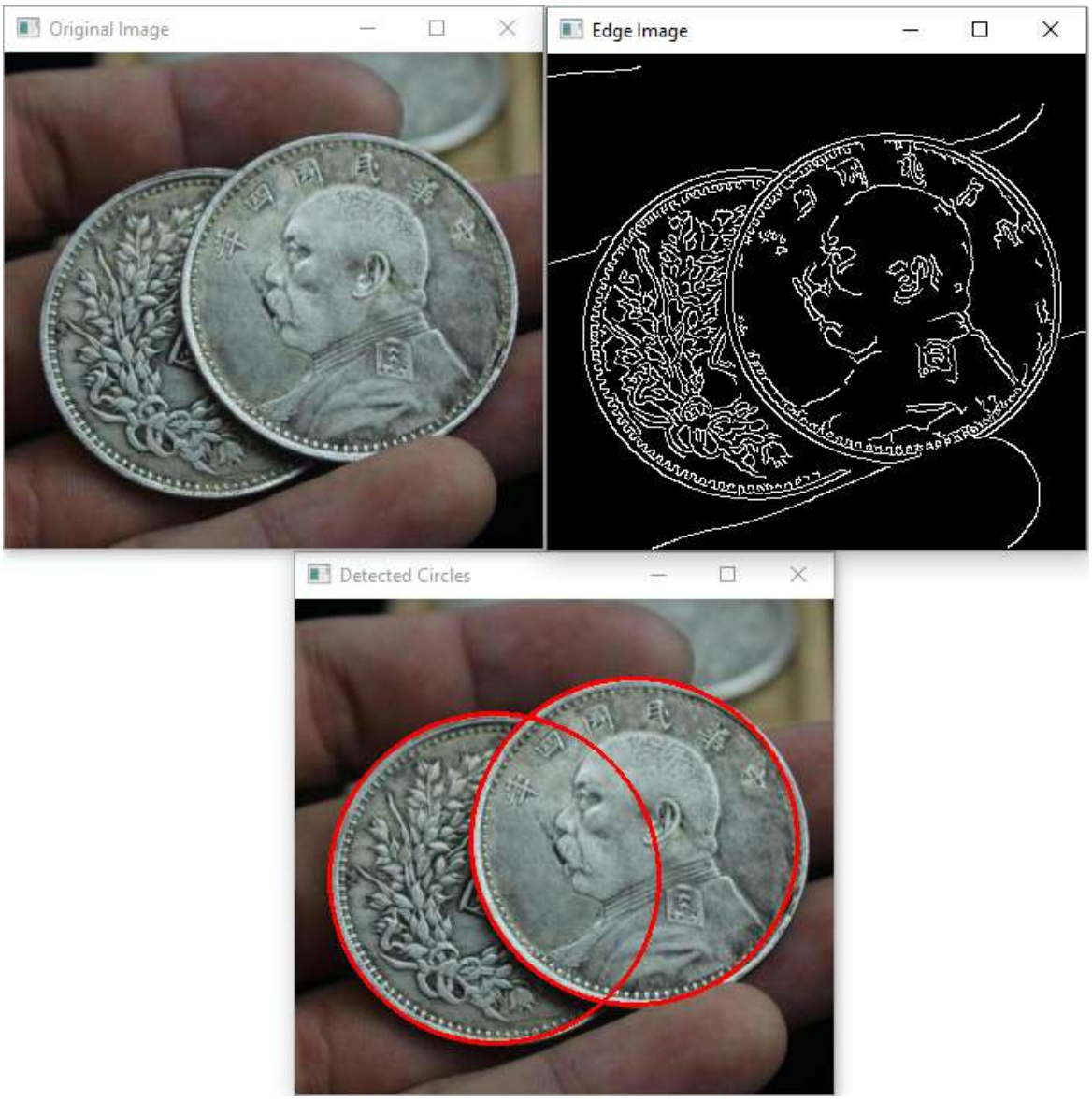
### 3.10 102.jpg




**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.819
Max IOU Score of 2. Circle: 0.735
Max IOU Score of 3. Circle: 0.819
Max IOU Score of 4. Circle: 0.817
Max IOU Score of 5. Circle: 0.819
Max IOU Score of 6. Circle: 0.819
Max IOU Score of 7. Circle: 0.819
Max IOU Score of 8. Circle: 0.819
Max IOU Score of 9. Circle: 0.815
Max IOU Score of 10. Circle: 0.819
Max IOU Score of 11. Circle: 0.914
Max IOU Score of 12. Circle: 0.819
Max IOU Score of 13. Circle: 0.895

Max IOU Score of 14. Circle: 0.813
Max IOU Score of 15. Circle: 0.819
Max IOU Score of 16. Circle: 0.819
Max IOU Score of 17. Circle: 0.819
Max IOU Score of 18. Circle: 0.819
Max IOU Score of 19. Circle: 0.813
Max IOU Score of 20. Circle: 0.819
Max IOU Score of 21. Circle: 0.819
Max IOU Score of 22. Circle: 0.735
Max IOU Score of 23. Circle: 0.826
Max IOU Score of 24. Circle: 0.851
Max IOU Score of 25. Circle: 0.898
Max IOU Score of 26. Circle: 0.819
Max IOU Score of 27. Circle: 0.918
Max IOU Score of 28. Circle: 0.907
Max IOU Score of 29. Circle: 0.819
Max IOU Score of 30. Circle: 0.819
Max IOU Score of 31. Circle: 0.819
**Average IOU score for this image: 0.828**

### 3.11 106.jpg



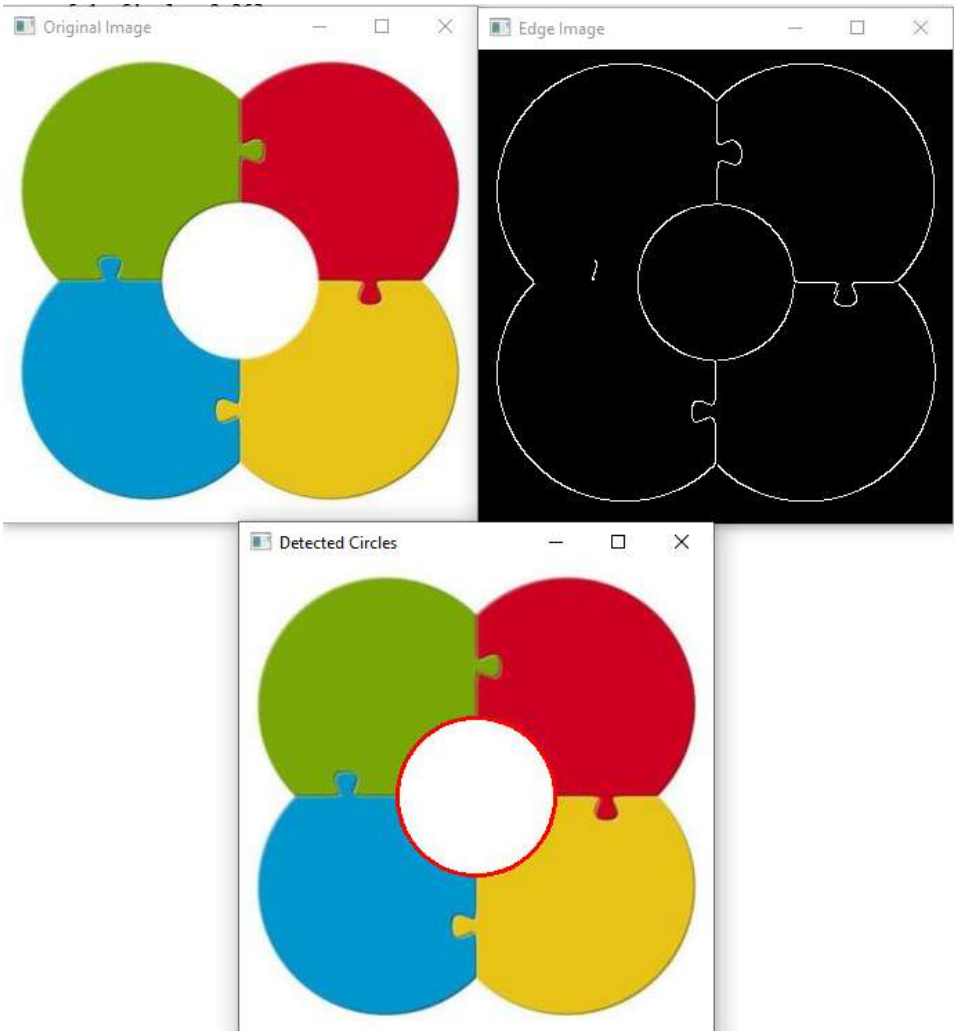

**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.961
Max IOU Score of 2. Circle: 0.974
**Average IOU score for this image: 0.968**

# 4 Failed Examples

## 4.1 8.jpg

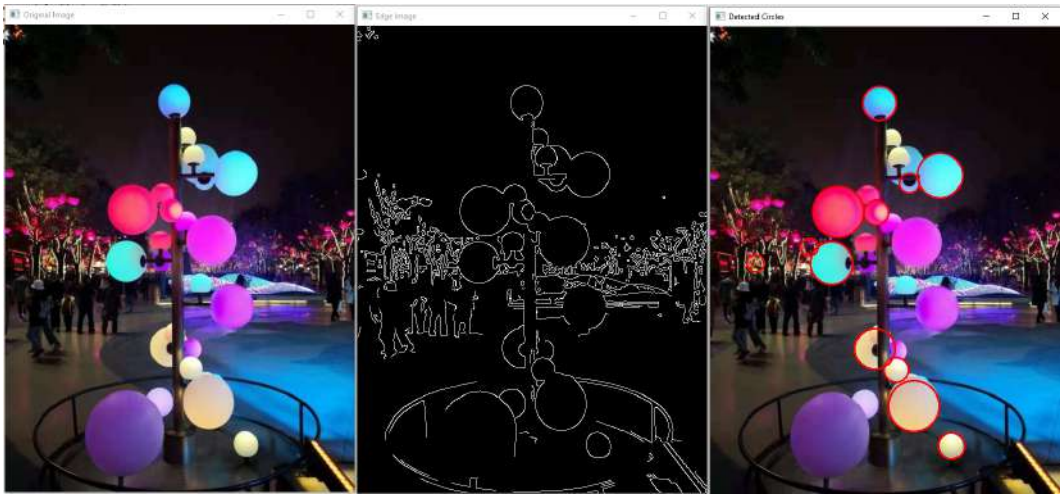

**IOU scores of this file :**
Max IOU Score of 1. Circle: 0
Max IOU Score of 2. Circle: 0
Max IOU Score of 3. Circle: 0
Max IOU Score of 4. Circle: 0.000519
Max IOU Score of 5. Circle: 0.771
Max IOU Score of 6. Circle: 0.972
Max IOU Score of 7. Circle: 0
Max IOU Score of 8. Circle: 0
Max IOU Score of 9. Circle: 0
Max IOU Score of 10. Circle: 0.178
Max IOU Score of 11. Circle: 0
Max IOU Score of 12. Circle: 0
**Average IOU score for this image: 0.16**

### 4.2 71.jpg



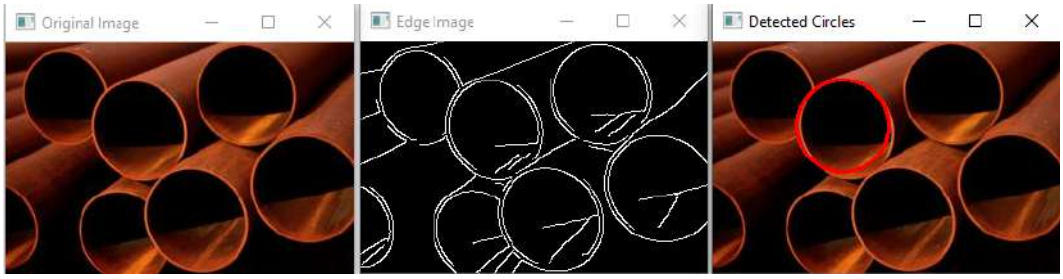

**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.134
Max IOU Score of 2. Circle: 0.139
Max IOU Score of 3. Circle: 0.147
Max IOU Score of 4. Circle: 0.137
Max IOU Score of 5. Circle: 0.965
**Average IOU score for this image: 0.304**

### 4.3 45.jpg



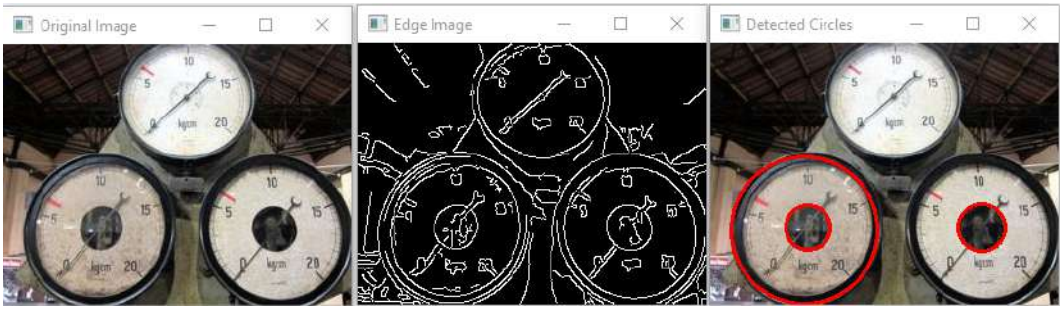
**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.944
Max IOU Score of 2. Circle: 0
Max IOU Score of 3. Circle: 0
Max IOU Score of 4. Circle: 0.862
Max IOU Score of 5. Circle: 0
Max IOU Score of 6. Circle: 0.882
Max IOU Score of 7. Circle: 0.922
Max IOU Score of 8. Circle: 0.915
Max IOU Score of 9. Circle: 0.81
Max IOU Score of 10. Circle: 0
Max IOU Score of 11. Circle: 0.859
**Average IOU score for this image: 0.563**

### 4.4 96.jpg



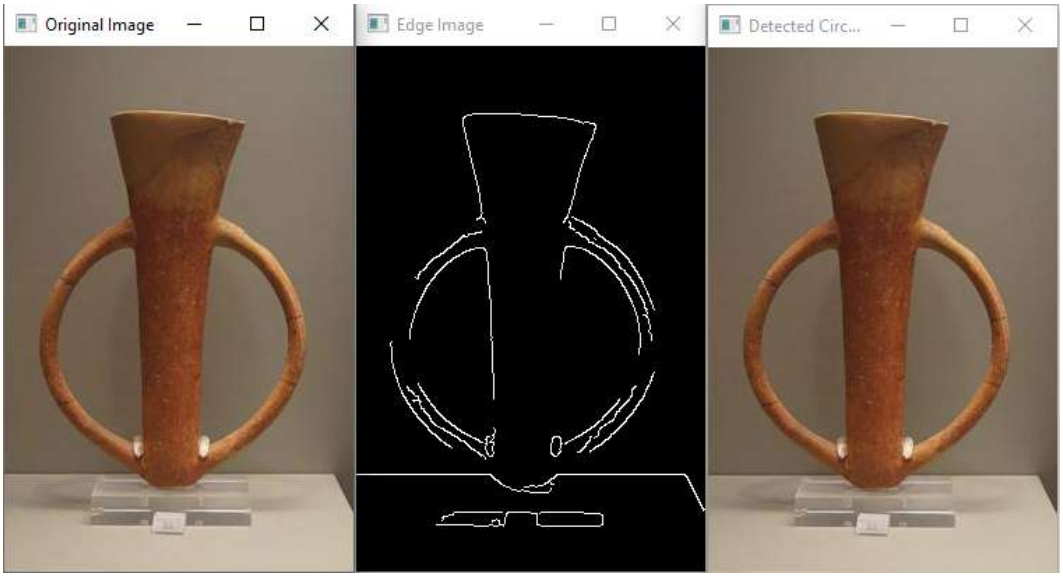

**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.0203
Max IOU Score of 2. Circle: 0.92
Max IOU Score of 3. Circle: 0
Max IOU Score of 4. Circle: 0
Max IOU Score of 5. Circle: 0
Max IOU Score of 6. Circle: 0
Max IOU Score of 7. Circle: 0
**Average IOU score for this image: 0.134**

### 4.5 98.jpg




**IOU scores of this file :**
Max IOU Score of 1. Circle: 0.0939
Max IOU Score of 2. Circle: 0.124
Max IOU Score of 3. Circle: 0.898
Max IOU Score of 4. Circle: 0.934
Max IOU Score of 5. Circle: 0.807
Max IOU Score of 6. Circle: 0.905
Max IOU Score of 7. Circle: 0
**Average IOU score for this image: 0.537**

### 4.6 111.jpg




**IOU scores of this file :**
There is no circle !
Max IOU Score of 1. Circle: 0
**Average IOU score for this image: 0**

## 5 Conclusions

| Successful Circle Detection Examples | |
|---|---|
| File Name | IOU Score |
| 3.jpg | 0.902 |
| 88.jpg | 0.98 |
| 27.jpg | 0.805 |
| 28.jpg | 0.969 |
| 49.jpg | 0.901 |
| 68.jpg | 0.914 |
| 69.jpg | 0.922 |
| 85.jpg | 0.82 |
| 100.jpg | 0.849 |
| 102.jpg | 0.828 |
| 106.jpg | 0.968 |

| | Failed Circle Detection Examples |
|---|---|
| File Name | IOU Score |
| 8.jpg | 0.16 |
| 71.jpg | 0.304 |
| 45.jpg | 0.563 |
| 96.jpg | 0.0134 |
| 98.jpg | 0.537 |
| 111.jpg | 0.0 |

1. The Final Result largely depends on the type of noise in the image.

2. A noisier image will have more false positives than a less noisy image.

3. The results in the above successful images are usually successful, but sometimes missing or extra-circling as in the unsuccessful images, mainly due to edge noise.

4. The extra circle detected can be eliminated with stricter thresholds, or the flats that cannot be found can be found by lowering the threshold value.

5. Only OpenCV's library functions are used to eliminate noise and detect edges.

6. Thresholds found are set for the image provided as input. Because each image is different in content, noise, and edges, there is no guarantee that these will work for all images.

## References

Your references here.

[1] https://www.geeksforgeeks.org/python-opencv-bgr-color-palette-with-trackbars/

[2]https://en.wikipedia.org/wiki/Canny_edge_detector https://en.wikipedia.org/wiki/Hough_transform

https://en.wikipedia.org/wiki/Generalised_Hough_transform

https://stackoverflow.com/questions/4247889/area-of-intersection-between-two-circles

https://opencv.org