Introduction

Object : To detect pool balls over the input image

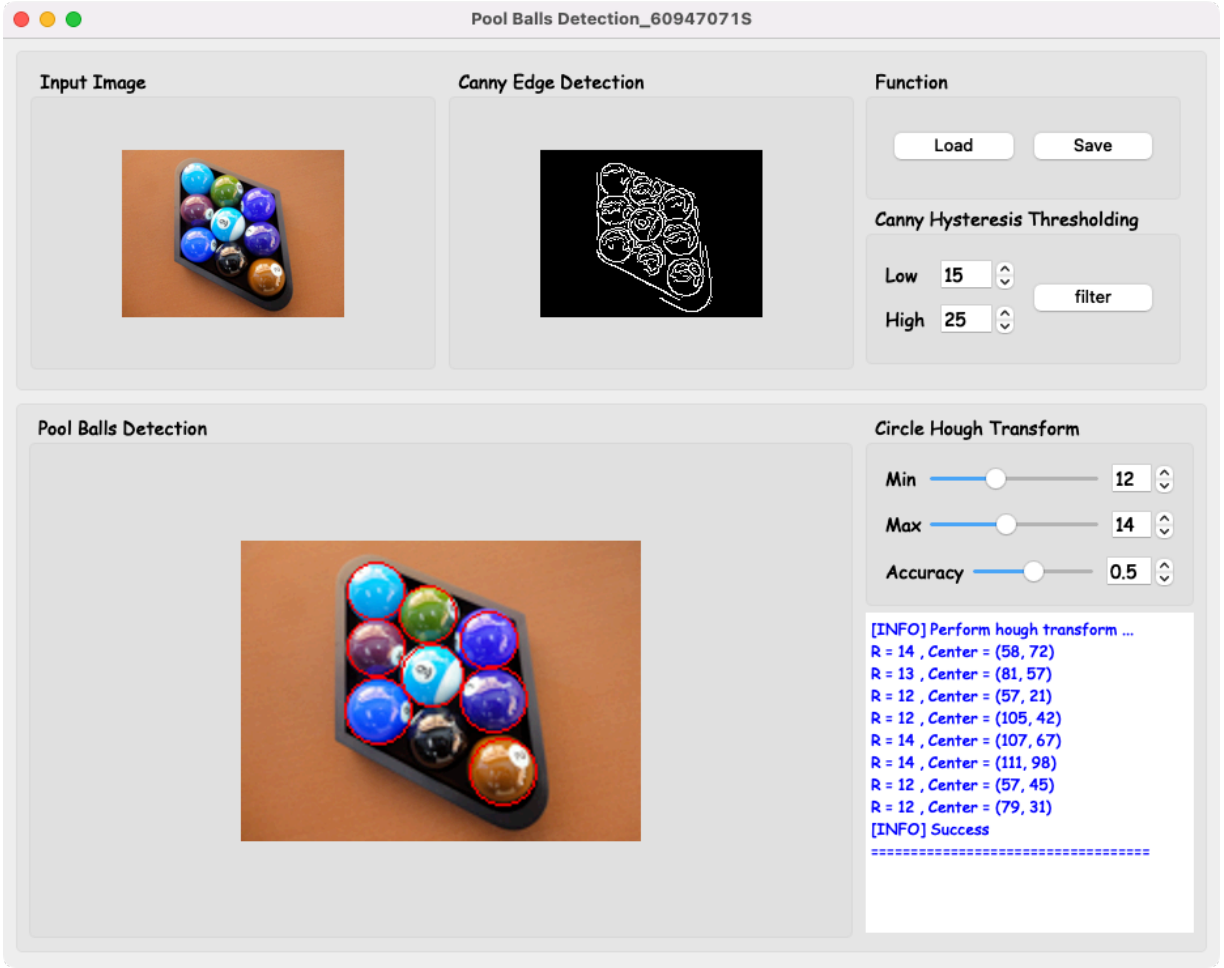
Program language : Python

GUI Interface : PyQt5

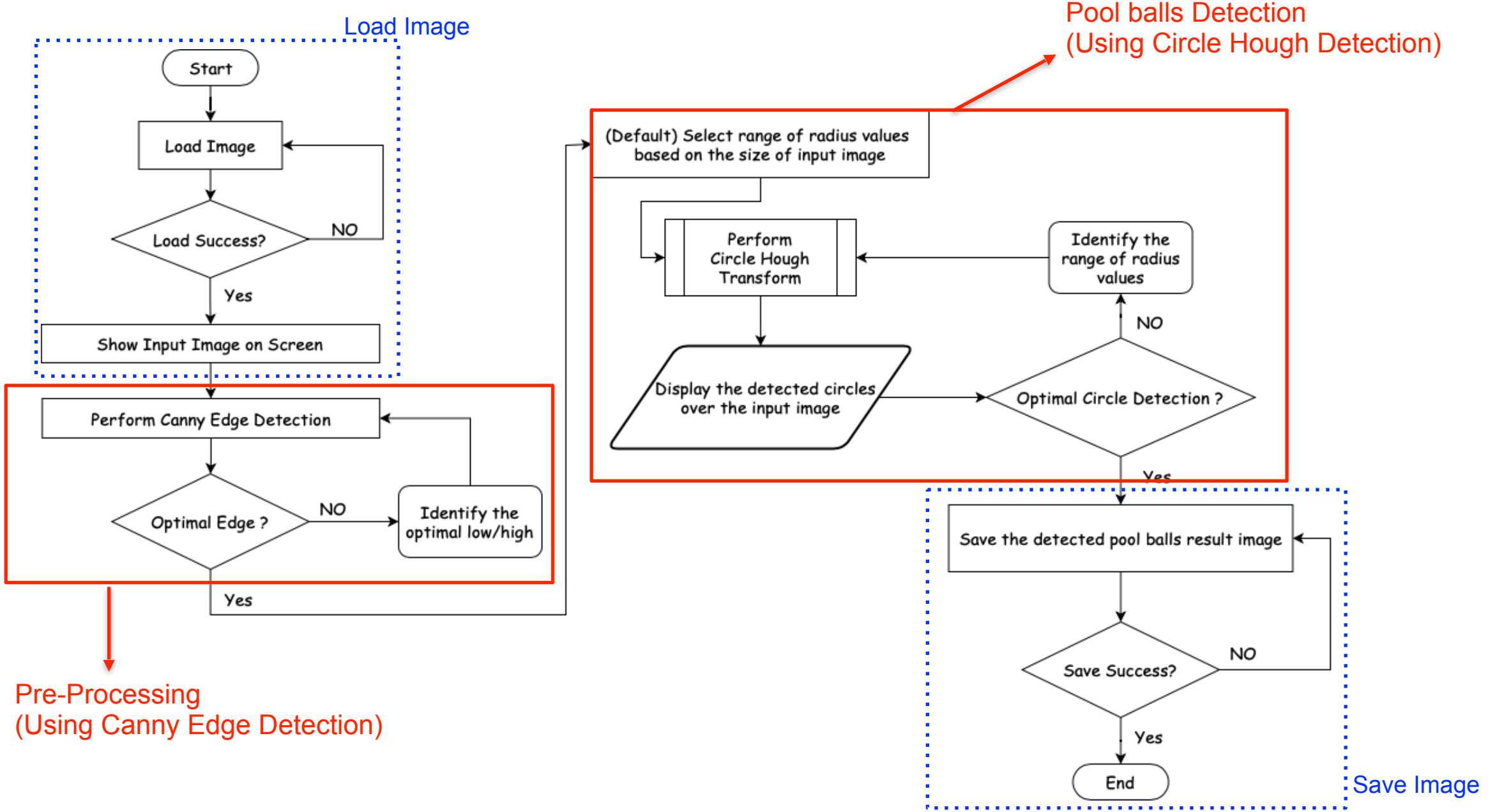
Main Function :

Canny_Edge_Detection()

Circle_Hough_Detection()

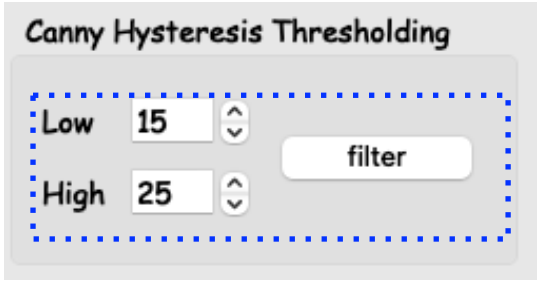


Flow Chart


Load Image
Start
Load Image
Load Success?
NO
Yes
Show Input Image on Screen
Perform Canny Edge Detection
Optimal Edge ?
NO
Identify the optimal low/high
Yes
Pre-Processing
(Using Canny Edge Detection)
Pool balls Detection
(Using Circle Hough Detection)
(Default) Select range of radius values based on the size of input image
Perform Circle Hough Transform
Identify the range of radius values
NO
Display the detected circles over the input image
Optimal Circle Detection ?
Yes
Save the detected pool balls result image
Save Success?
NO
Yes
End
Save Image

Pre-Proceesing

```
def canny_edge_detection(input_image, low_value, high_value):
    input_pixels = input_image.load()
    width = input_image.width
    height = input_image.height

    # Transform the image to grayscale
    grayscaled = compute_grayscale(input_pixels, width, height)

    # Blur it to remove noise
    blurred = compute_blur(grayscaled, width, height)

    # Compute the gradient
    gradient, direction = compute_gradient(blurred, width, height)

    # Non-maximum suppression
    filter_out_non_maximum(gradient, direction, width, height)

    # Filter out some edges
    keep = filter_strong_edges(gradient, width, height, low_value, high_value)

    return keep
```



Gaussian Filter

$$K = \frac{1}{256}\begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$

Sobel Filter

-1	0	+1
-2	0	+2
-1	0	+1

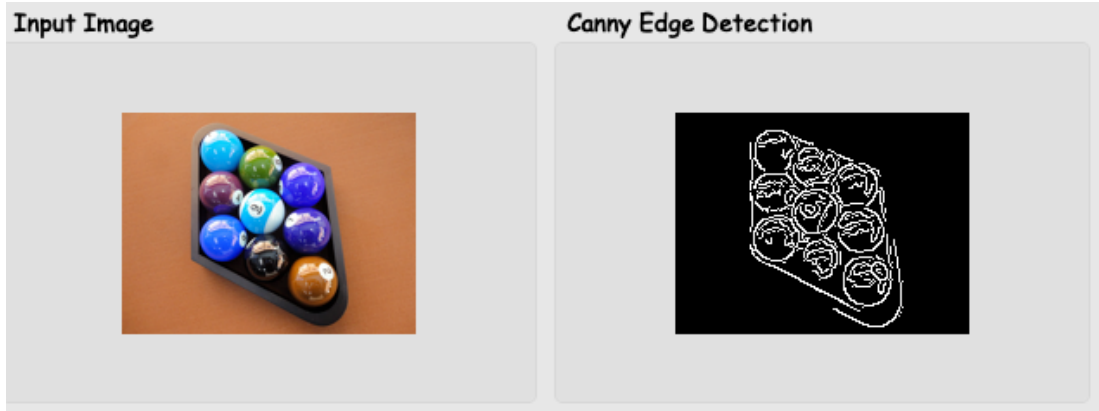
Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

$$G = \sqrt{G_x^2 + G_y^2}$$

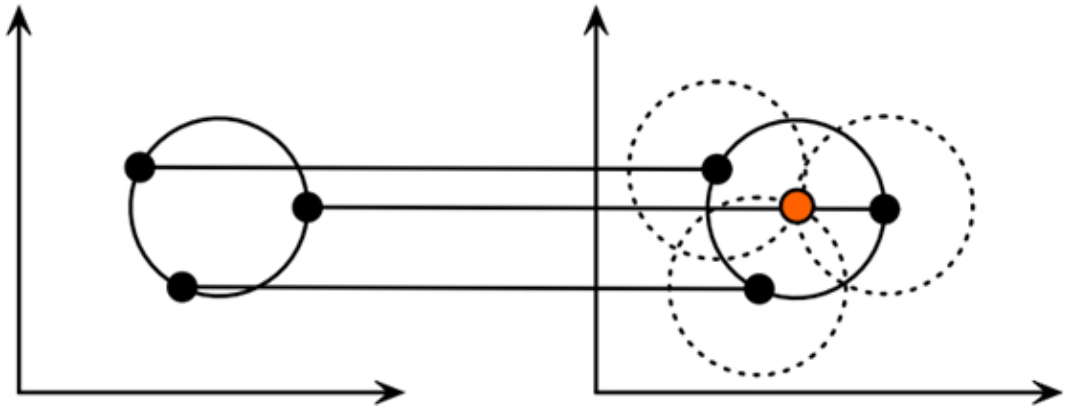
$$\theta = \arctan\left(\frac{G_y}{G_x}\right)$$



Pool Balls Detected

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

$$x = a + R\cos(\theta)$$
$$y = b + R\sin(\theta)$$



Predefine min-radius & max-radius