# Computer Vision

Assignment 3 - Hough Transform
Summer 2023

## Introduction

In this assignment you will demonstrate your ability to implement utilize the *Hough Transform* for finidng parameterizable objects from edge pixels. In particulater, you will demonstrate your ability to:

- Generate "fake" edge data.
- Apply hough transform to edge data for various parameterizable shapes.
- Find local maxima in hough transform.
- Generate shapes based on parameterized shapes.

In this assignment, as with all of our assignments, you shouldn't be using built-in functions that violate the "spirit" of the assignment. For instance, any sort of functions to generate circles or lines, or to compute hough transforms, are forbidden. *However*, since we have already implemented edge detection in a prior assignment, you **may** use a function like *edge* to obtain edge pixels in an image. In general, use your intuition, and when in doubt ask the instructor or TA.

## Grading

| | |
|---|---|
| Theory Questions | 10pts |
| Generating lines and circles | 10pts |
| Hough transform for a line | 20pts |
| Hough transform for a circle | 20pts |
| Single circle detection on real image | 20pts |
| Multiple circle detection on real image | 10pts |
| Performance on another test image (done by instructor) | 10pts |
| **TOTAL** | 100pts |

Table 1: Grading Rubric

# Dataset

For this assignment we're going to use two pieces of data:

1. Synthetically generated data
2. A provided grayscale image, *circles1.gif*

# 1 (10pts) Theory Questions

For the following questions, show the computations to support your answers.

1. (5pts) Given an image with a width of 200 pixels, and a height of 100 pixels, what is the size of the hough transform accumulator for lines if the step side for each parameter is 1.0 (assume that the angle is in degrees)?

$$Size = ceil(\sqrt{200^2 + 100^2}) \cdot (180 + 1) = \mathbf{40{,}321}$$

2. (5pts) If the probability of an edge pixel is actually on the object we are attempting to detecting is 0.2, and the desired accuracy of our model is 0.99, how many independent tests must we run $RANSAC$ on, if we are attempting to detect a line?

$$N_{tests} = \frac{log(1 - 0.99)}{log(1 - 0.2^2)} = \mathbf{112}$$

## 2 Generate Fake Data

To start off with, let's generate some fake edge data so that we know what the "solution" is.

Create a $400 \times 400$ binary image that has two objects on it:

- A line with slope $m = 1$ and y-intersept $b = -100$.
- A circle with center $x = 100, y = 200$ and radius $r = 50$.

You can generate the line by starting varying $x$ from its minimium to maximum value, generating $y$ values along the way according to the formula $y = mx + b$.

You can generate the circle by varying $\theta = [0, 2\pi]$ and generating the $(x, y)$ coordinates according to:

$$x = x_0 + rcos\left(\theta\right)$$

$$y = y_0 + rsin\left(\theta\right)$$

where $(x_0, y_0)$ is the center of the circle and $r$ is its radius.

*Note* that since the origin of an image's coordinate system is the top-left, its positive y-axis is *down*, and therefore your image is a reflection about the x-axis from what you might imagine it to be.

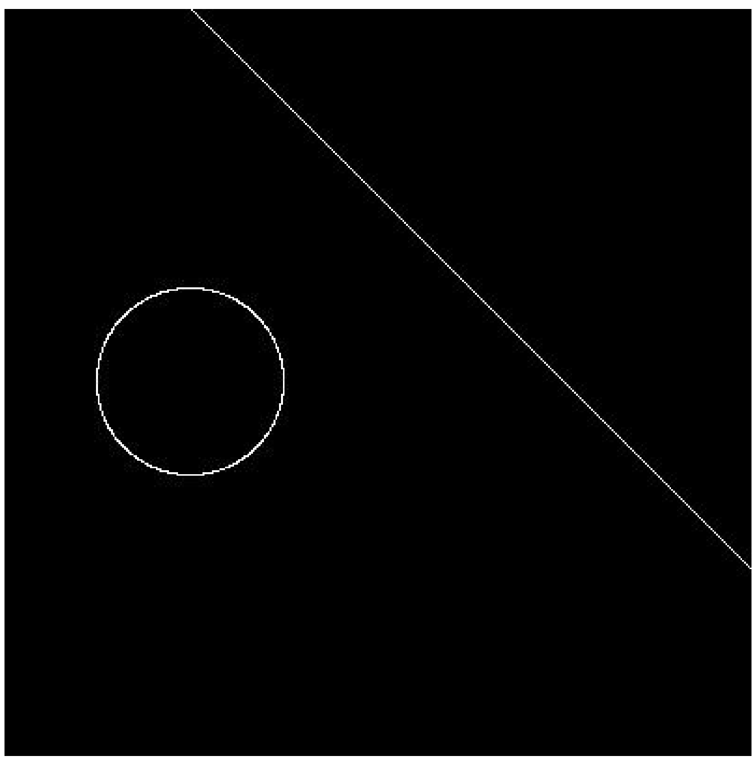


Figure 1: My binary image

# 3 Hough Transform for a Line

Next let's try to find the parameters of that line! Apply the *Hough Transform for a Line* to your binary image. Use the *polar* form, varying the of the parameters $\theta, r$ according to the slides, and incrementing them by one for each bin.

**NOTE**: You CANNOT use the Matlab function *hough* or anything related to do this for you. However, you MAY want to use a function like *ind2sub* to help you get subscript indices from a linear index.

In your report, provide:

- The value of $(\theta, r)$ that cooresonds to the maximum value in the Hough Transform.
- The corresponding values for $(m, b)$ where $m$ is the slope and $b$ is the y-intercept. Show the equations/formulas that allow you to compute those from $(\theta, r)$.
- A plot of the Hough Transform as an image.

Figure 2: $\theta = -45, \rho = 69, m = 1, b = -100$

# 4 Hough Transform for Circle

Next let's try to find that circle! To find all the parameterize your circle, you'll need a 3D Hough Transform ($x_0, y_0, r$, where $x_0$ and $y_0$ are the $x$ and $y$ coordinates of the center of the circle, respectively, and $r$ is its radius).

You may make the following assumptions:

- The circle's center is within the bounds of the image.
- The circle's radius is less than the diagonal length of the image.

Much like with the line, in your report, provide:

- The value of $(x_0, y_0, r)$ that cooresonds to the maximum value in the Hough Transform.
- A plot of the Hough Transform as an image. **However**, since this is a 3D histogram just plot $x$ vs $y$ for the *slice* where $r = rmax$ where $rmax$ is the value of $r$ find in the max bin.

An example can be found in Figure 3.

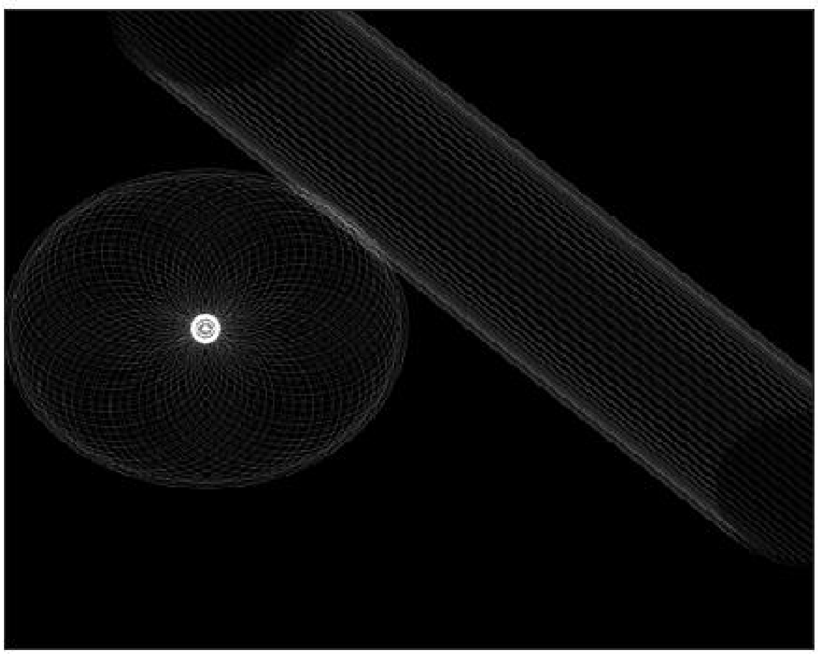


Figure 3: $(x, y, r) = (100, 200, 48)$

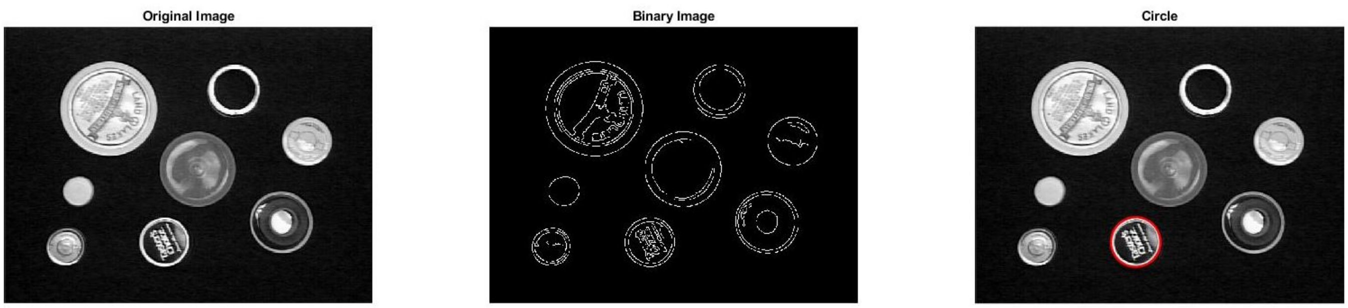
# 5 Apply to a Real Image

Now let's apply this stuff to a real image!

For the problem we'll use the provide image *circles.gif*. However, this is a grayscale image, not a binary image. In your previous assigment you implemented elements of a Canny Edge Detector. For this one you'll just use Matlab's *edge* function to do this for you. Feel free to play with any parameters of that function, but if you deviate from the defaults, put in your report what parameters you changed.

Once you have your binary image, apply your Hough Circle detection to it. Display as subplots:

- Original image
- Binary image
- Original image with dominant circle superimposed in *red*.




$$(x, y, r) = (279, 374, 43)$$

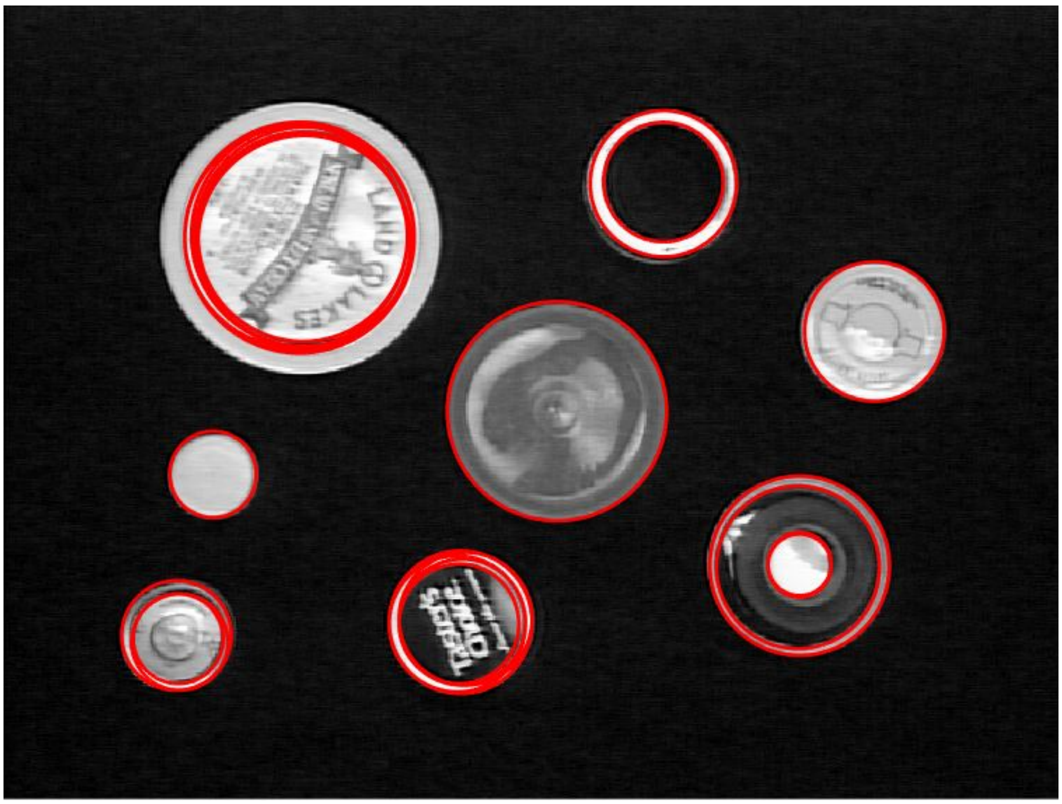
# 6 Additional Circles

Of course there was more than one circle in that image! For the last part, attempt to identify all the "major" circles in the image. Of course "major" is a bit subjective, but in general no circles should be either partial or fully enclosed in one another. If a set are, the largest should only be kept.

Other ideas include:

- Apply some threshold
- Then find local maximas. If you do this, you CANNOT use a function like *findlocalmaxima* or related. You must do so yourself.

In your report describe how you implemented this. In addition, provide a subplot like you did in the previous part as well as the parameters of the circles you detected. We will also test your approach on an additional image for robustness.



This method filters the Hough Accumulator based on the fraction of the previously established maximum, and then finds the regional maxima of the binary image, which is then used to iterate over the centers of the major circles.

# Submission

For your submission, upload to Blackboard a single zip file containing:

1. PDF writeup that includes:
   (a) Your answer to the theory question(s).
   (b) For Part 2, your generated binary image.
   (c) For Part 3, the values for $\theta$ and $r$ that provide the maximum in your Hough transform, their corresponding values for the slope and y-intercept of that line, and an image of the Hough transform.
   (d) For Part 4, the values for the center and radius of the circle that provides the maximum in your Hough transform as well as an image of a *slice* of the Hough transform where $r$ has its maximum value.
   (e) For Part 5, three images: original image, binary edge image, and original image with detected dominent circle superimposed on it. In addition, provide the parameters of that dominant circle.
   (f) For Part 6, a description of your algorithm, a list of the parameters of the detected circles, the subplot image, similar to in the previous part.
2. A README text file (**not** Word or PDF) that explains:
   (a) Any unique features of your program (if applicable).
   (b) Any instructions on how to run your script to reproduce your results.
3. Your source file(s).