Lab Report 9 : Hough Transform for Straight Line Detection and Circle Detection

Course Title: Digital Image Processing Lab

Course Code: CSE-406

Date of Performance: October 27, 2024

Date of Submission: November 03, 2024

Submitted to:

Dr. Morium Akter
Professor

Dr. Md. Golam Moazzam
Professor

Department of Computer Science and Engineering
Jahangirnagar University

Submitted by:

Class Roll	Exam Roll	Name
370	202182	Rubayed All Islam

Experiment Name

a **Hough Transform for Straight Line Detection in Images**

b **Hough Transform for Circle Detection in Images**

Objectives

The objectives of the experiments performed in this lab are as follows:

Straight Line Detection:

1. To apply the Hough Transform for detecting straight lines in a grayscale image.
2. To preprocess the image using edge detection, enhancing line features for more effective detection.
3. To visualize the Hough Transform matrix, identify peaks, and use these to detect significant lines.
4. To overlay detected lines on the original image for visual verification.

Circle Detection:

1. To detect circular shapes within an image using a Circular Hough Transform.
2. To determine appropriate radius ranges for detecting circles of different sizes.
3. To overlay detected circles on the original image for verification.

Experiment 1: Hough Transform for Straight Line Detection in Images

MATLAB Code:

Listing 1: Hough_Transform_StraightLine.m

```
% Read the image
I = imread('image.jpeg');  % Replace with your image filename

% Convert the image to grayscale if it is in RGB
grayImage = rgb2gray(I);

% Apply edge detection (using Canny edge detector)
edges = edge(grayImage, 'canny');

% Perform the Hough transform
[H, theta, rho] = hough(edges);

% Find peaks in the Hough transform matrix
```

```
P = houghpeaks(H, 5, 'threshold', ceil(0.3 * max(H(:))));

% Find and draw lines on the original image
lines = houghlines(edges, theta, rho, P, 'FillGap', 5,
    'MinLength', 7);

% Create a figure with multiple subplots
figure;

% Display the original image
subplot(2, 2, 1);
imshow(I);
title('Original Image');

% Display the edge-detected image
subplot(2, 2, 2);
imshow(edges);
title('Edge Detected Image');

% Display the Hough transform matrix
subplot(2, 2, 3);
imshow(imadjust(rescale(H)), 'XData', theta, 'YData', rho, ...
        'InitialMagnification', 'fit');
xlabel('\theta (degrees)');
ylabel('\rho');
axis on;
axis normal;
title('Hough Transform of Edge Image');
colormap(gca, hot);

% Plot the peaks on the Hough transform matrix
hold on;
plot(theta(P(:,2)), rho(P(:,1)), 's', 'color', 'blue');
hold off;

% Display the original image with detected lines
subplot(2, 2, 4);
imshow(I);
title('Detected Lines on Original Image');
hold on;

% Plot each detected line segment
for k = 1:length(lines)
    xy = [lines(k).point1; lines(k).point2];
    plot(xy(:,1), xy(:,2), 'LineWidth', 2, 'Color', 'green');

    % Plot beginnings and ends of lines
    plot(xy(1,1), xy(1,2), 'x', 'LineWidth', 2, 'Color',
        'yellow');
    plot(xy(2,1), xy(2,2), 'x', 'LineWidth', 2, 'Color', 'red');
end
```

```
63
64 hold off;
```

Output:

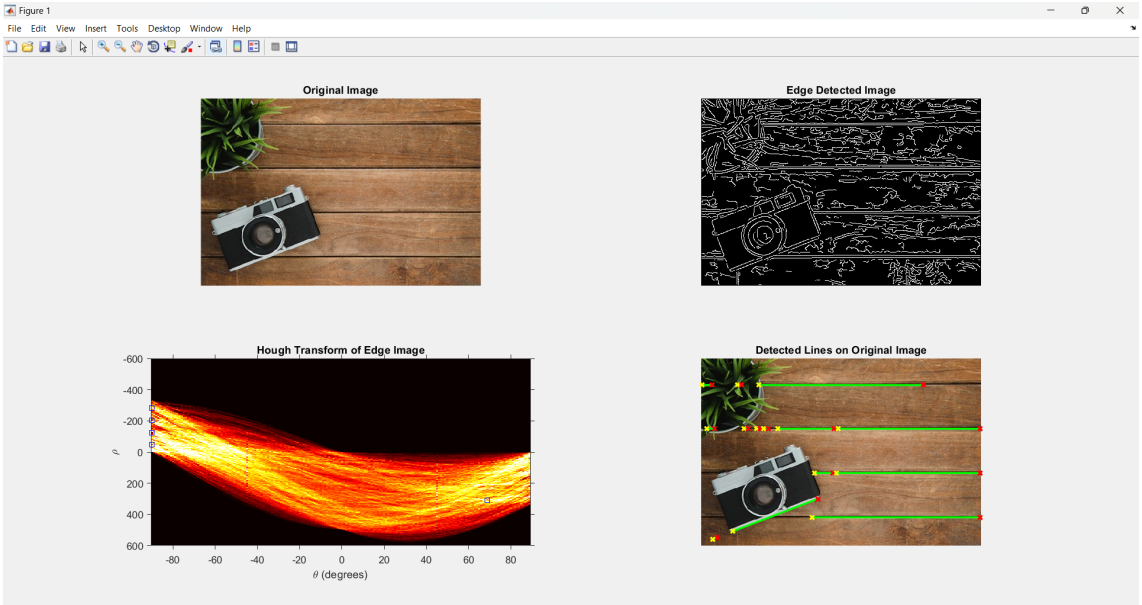

Figure 1: Corrupting Image by Gaussian, Rayleigh, and Erlang Noise

Experiment 2: Hough Transform for Circle Detection in Images

MATLAB Code:

Listing 2: Hough_Transform_Circle.m

```
% Read the image
img = imread('Circle_Image.jpg');  % Replace 'Circle_Image.jpg' with your image file

% Check if the image is grayscale or RGB, and convert if necessary
if size(img, 3) == 3
    grayImg = rgb2gray(img);       % Convert to grayscale if the image is RGB
else
    grayImg = img;                 % Image is already grayscale
end

% Set radius range based on expected circle sizes
minRadius = 20;
maxRadius = 150;

% Detect circles using Circular Hough Transform
[centers, radii, metric] = imfindcircles(grayImg, [minRadius maxRadius], ...
                                         'Sensitivity', 0.95, 'EdgeThreshold', 0.1);

% Plot both the original image and the detected circles side-by-side
figure;

% Display the original image
subplot(1, 2, 1);
imshow(img);
title('Original Image');

% Display the image with detected circles
subplot(1, 2, 2);
imshow(img);
title('Detected Circles');
hold on;

% Check if any circles are detected
if ~isempty(centers)
    % Draw circles and mark centers
    viscircles(centers, radii, 'EdgeColor', 'b', 'LineWidth', 1.5);
```

```
    plot(centers(:,1), centers(:,2), 'r+', 'MarkerSize', 10,
        'LineWidth', 2);
else
    disp('No circles were detected.');
end

hold off;
```

Output:

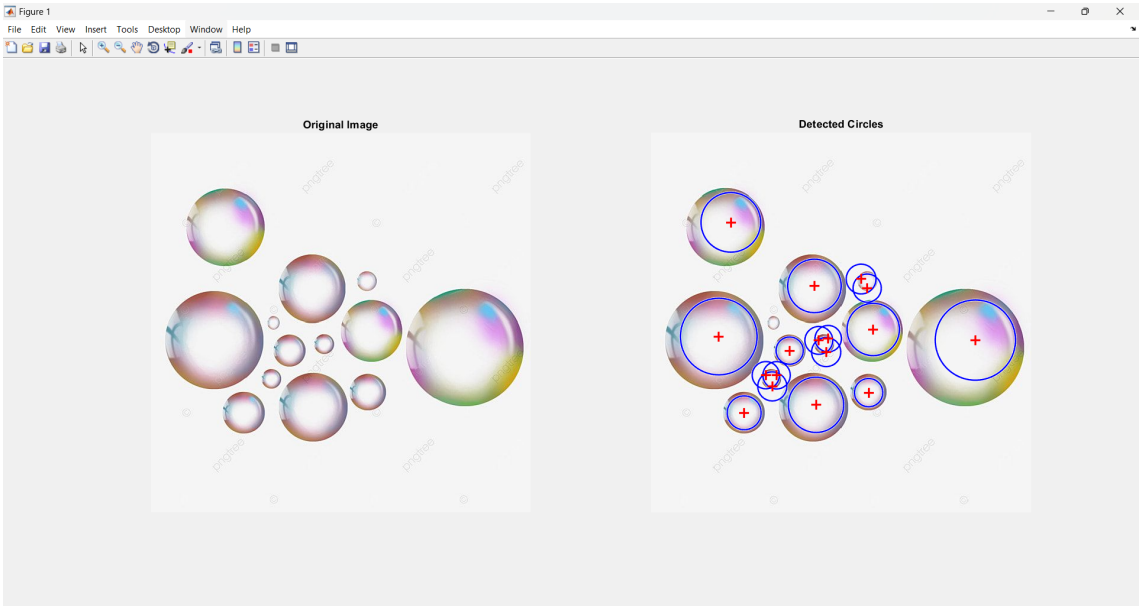

Figure 2: Noise Removal by Arithmetic Mean Filter and Geometric Mean Filter

Results

1. Hough Transform for Straight Line Detection

The process for detecting straight lines in the input image involved the following steps:

- **Original Image**: Displayed as a baseline for line detection analysis.
- **Edge Detection**: Applied Canny edge detection on the grayscale version of the image, highlighting edges and reducing noise.
- **Hough Transform**: The edge-detected image was converted to Hough space. The resulting Hough Transform matrix displayed intensity peaks that correspond to potential lines in the image.
- **Peaks Identification**: Identified peaks in the Hough Transform matrix, each peak representing a significant line in the image.

- **Line Overlay**: The lines were plotted on the original image using coordinates derived from the identified peaks. Each detected line segment is highlighted in green, with line endpoints marked in yellow and red.

Figure Summary:

1. Original Image.
2. Edge-Detected Image.
3. Hough Transform Matrix with peaks marked in blue.
4. Original Image with Detected Lines Overlaid.

The results show that the Hough Transform effectively detected prominent straight lines within the image, indicating its accuracy and reliability in applications requiring line detection.

2. Hough Transform for Circle Detection

The steps for detecting circular shapes in the input image were as follows:

- **Original Image**: Displayed for reference.
- **Grayscale Conversion**: Converted the RGB image to grayscale, simplifying processing and focusing on luminance.
- **Edge Detection**: Applied Canny edge detection to highlight circular features in the image.
- **Circular Hough Transform**: Specified a radius range for circles to detect circles of different sizes. The Circular Hough Transform detected the centers and radii of circles within this range.
- **Circle Overlay**: Detected circles were overlaid on the original image. Circle boundaries are marked in blue, with circle centers marked in red for better visibility.

Figure Summary:

1. Original Image.
2. Edge-Detected Image.
3. Original Image with Detected Circles Overlaid.

The Circular Hough Transform accurately detected circular shapes within the specified radius range. This method demonstrates its utility in applications requiring the detection of circular features, such as bubble detection, object tracking, and medical imaging.

Conclusion

The Hough Transform successfully detected straight lines, confirming its reliability for line detection tasks. The edge detection preprocessing effectively reduced noise and complexity. This experiment highlights the Hough Transform's usefulness in applications needing straight-line recognition, such as road detection and structural inspection.

The Circular Hough Transform effectively identified circular shapes within the specified radius range, verifying its application for circle detection tasks. This method is useful for identifying objects such as wheels, bubbles, and other circular features in various image processing applications. Improvements could involve adjusting sensitivity parameters or exploring multi-scale Hough Transforms to detect circles of varying sizes simultaneously.