Computer Vision Project 2 Report

Pratik Mane
Graduate Student, Computer Science
Indiana University, Indianapolis

October 22, 2024

Overview

This report summarizes the implementation and results of key computer vision tasks, including Gradient-Based Edge Detection, Boundary Tracing, Harris Corner Detection, and Hough Transform for shape detection. Each task's objective, solution, observations, and known failure cases are outlined. Overall, the report reflects on the successes and limitations of these classical algorithms, highlighting areas for potential improvement in robustness and handling of complex image scenarios.

---

1 Gradient-Based Edge Detection

Objective

To write a gradient-based edge detector that computes the gradient magnitude and orientation of an image after smoothing it with a Gaussian filter.

Solution

Loaded a grayscale image and converted it to a double array.
Smoothed the image using a Gaussian filter, with the amount of smoothing determined by the parameter sigma.
Computed horizontal and vertical derivatives using a convolution-based approach.
Calculated the gradient magnitude and orientation, and displayed the gradient magnitude as an image and the orientation using quiver plots.

Observation

The edge detector was effective at identifying strong edges in images with clear contrast and sharp boundaries.
The gradient magnitude visualization provided clear edge strength, while the orientation plot correctly captured the direction of the gradient at each point.

Known Issues

Performance degraded with noisy or low-contrast images, as small variations could affect the gradient calculations.
Excessive smoothing (high sigma) resulted in blurring of important edges, while insufficient smoothing led to noisy gradients.

Output

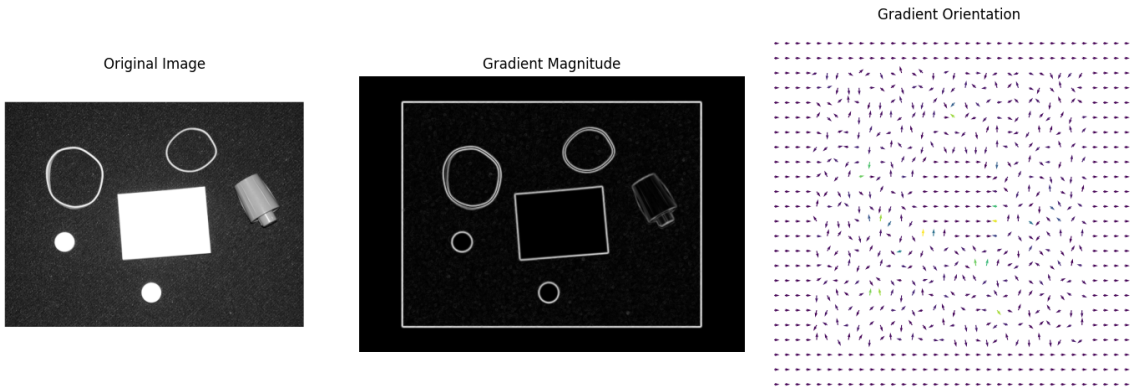

Figure 1: Gradient-Based Edge Detection

2 Boundary Detection

Objective

To extract and trace the outer boundaries of objects in an image, starting from a manually located boundary pixel.

Solution

Applied the edge detection algorithm to generate an edge map.
Used manual selection to locate a pixel on the outer boundary of each object.
Implemented a custom algorithm to trace the boundary pixels by following edge contours and plotting the traced boundary.

Observation

The boundary tracing algorithm performed well for objects with clearly defined edges and minimal noise.
Traced boundaries were visually close to the actual object boundaries, especially in images with simple shapes and minimal complexity.

Known Issues

Incomplete boundaries or gaps in the edge map caused discontinuities in the traced boundaries.
Complex shapes with many curves or fine details were more difficult to trace accurately.
Noise in the image often resulted in the tracing algorithm veering off from the true object boundary, especially in areas where the boundary was weak.

Output

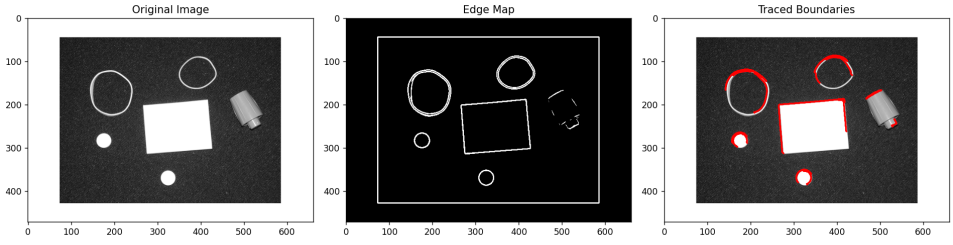

Figure 2: Boundary Detection

3 Harris Corner Detection

Objective

To implement a Harris corner detector that identifies corner points in an image by calculating the corner response at each pixel.

Solution

Computed the corner response at each pixel by evaluating the gradients and second-order derivatives of the image.
Applied non-maximum suppression to extract local maxima within a predefined radius, identifying these points as corners.

Observation

The Harris corner detector effectively identified corners in geometric objects such as rectangles and triangles.
It performed well in distinguishing corner points in images with regular shapes and clearly defined junctions.

Known Issues

The algorithm was sensitive to noise, leading to many false positives in high-texture regions.
In some images, it generated too many corner points, especially when dealing with patterns or textures where no clear structural corners were present.
Fine-tuning the radius for local maxima detection was critical to reduce false positives while still capturing significant corners.

Output

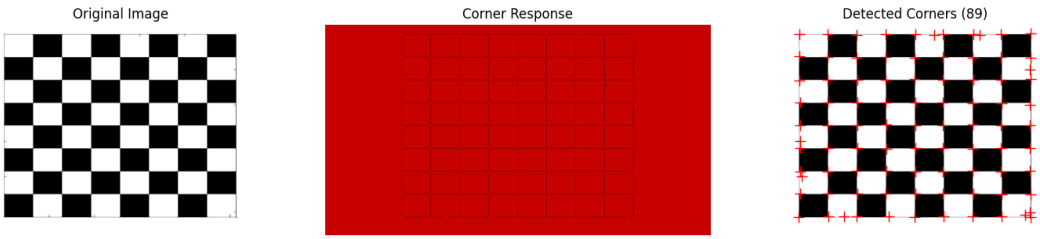

Figure 3: Corner Detection Checkboard

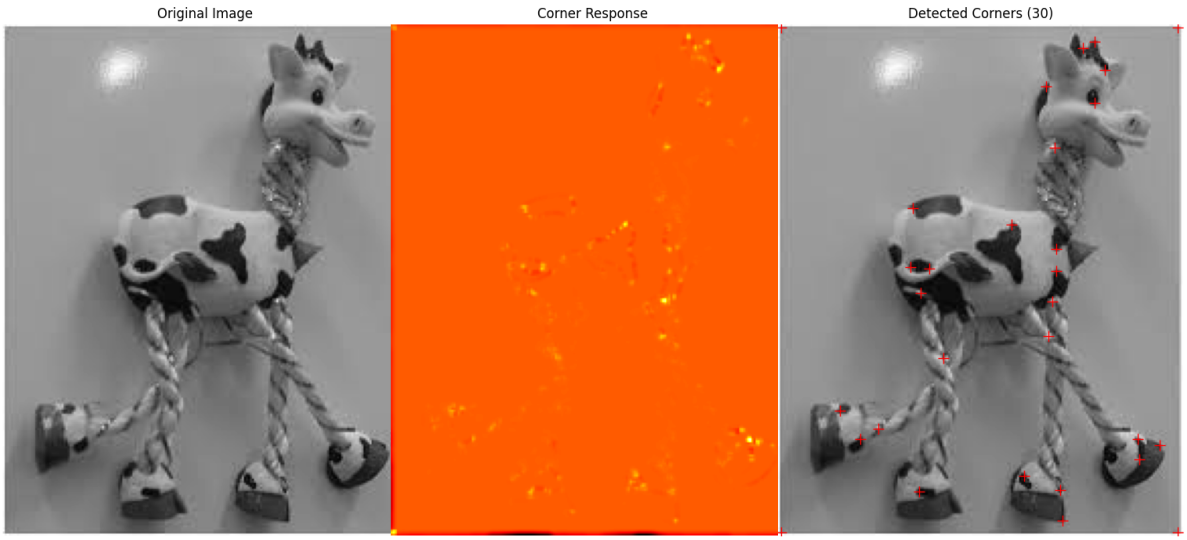

Figure 4: Upsampled Twice Median Filtered

4 Hough Transform for Line Detection

Objective

To implement the Hough transform to detect straight lines in a binary image.

Solution

Converted the image to a binary form and used an accumulator-based Hough transform to detect straight lines.
Implemented a voting system in Hough space, where points voted for all possible lines passing through them.
Identified the top n lines based on the voting results and plotted them on the image.

Observation

The line detection algorithm accurately identified straight lines in structured images, particularly when lines were well-separated and clearly defined.
. The voting mechanism in Hough space effectively ranked lines by prominence, allowing for the detection of the most important lines in the image.

Known Issues

In cases where lines were closely spaced or overlapping, the algorithm sometimes confused them, leading to inaccurate line detection.
Noise in the image led to extraneous line detections that did not correspond to actual features in the image.

Output

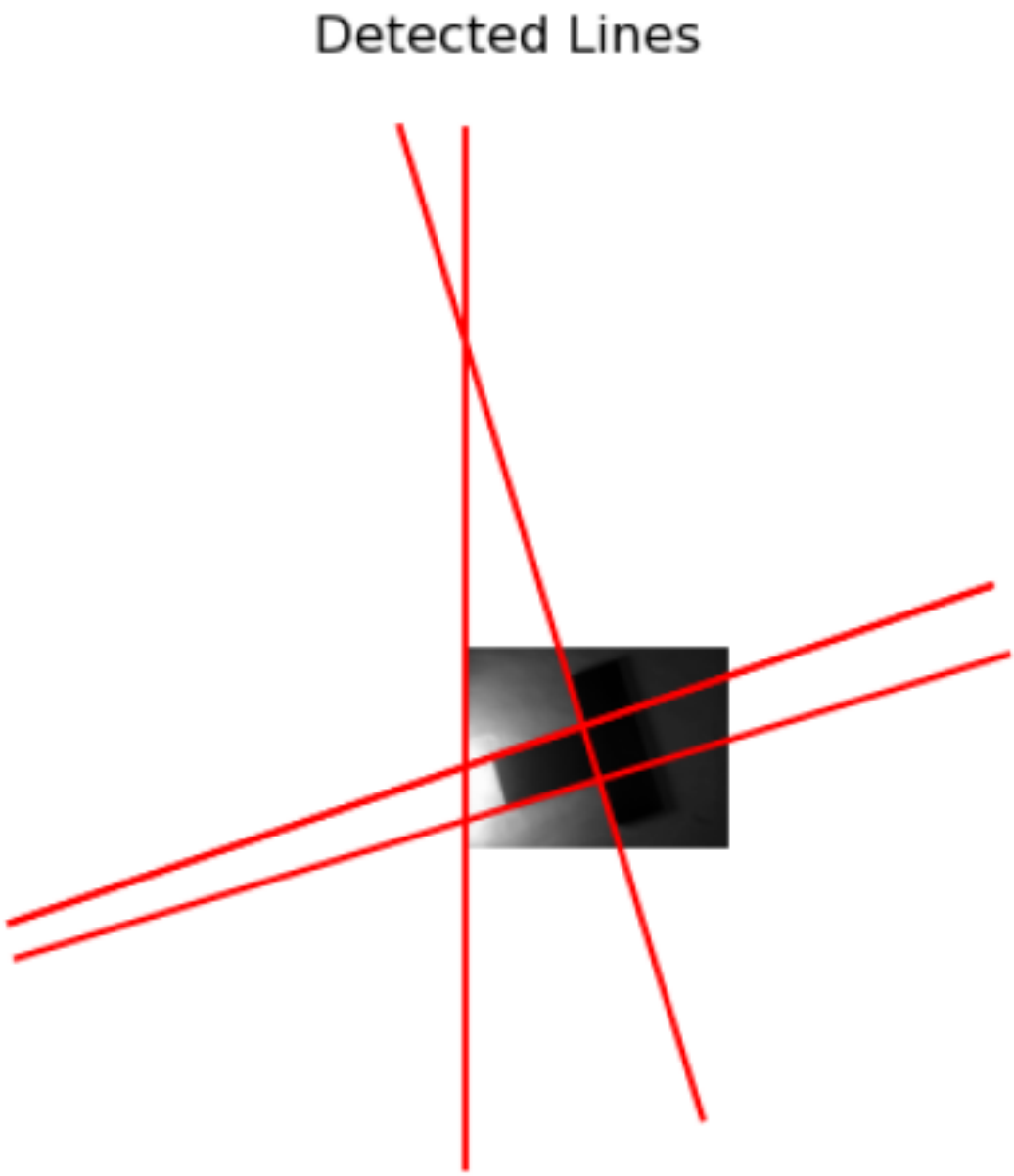

Figure 5: Line Detection

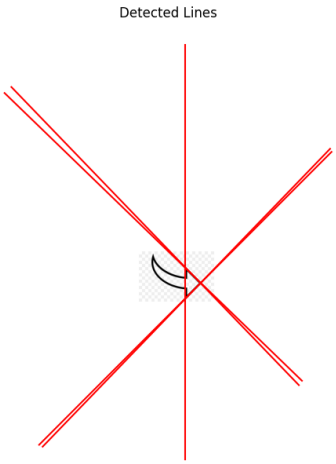

Figure 6: Line Detection on Arrow

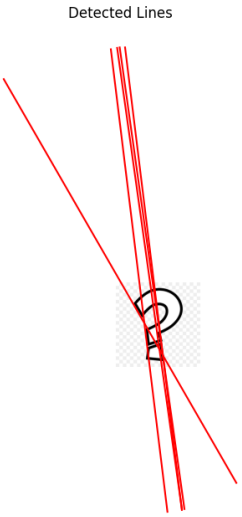

Figure 7: Line Detection on Question Mark

5 Hough Transform for Circle Detection

Objective

To detect circular objects in binary images using a custom Hough transform.

Solution

Implemented a two-step process where a training function, myHoughCircleTrain, analyzed an image containing a single circle to extract circular boundary features.
In the testing phase, applied the Hough transform to identify circular objects by comparing the boundary features learned in training.
The algorithm reported reference points for the detected circles.

Observation

The circle detection algorithm worked well for simple images containing circles with a known radius, successfully identifying the reference points for the detected circles.
It was effective at distinguishing circles when the shapes were well-defined and isolated.

Known Issues

The algorithm struggled to detect circular shapes when they were irregular or partially occluded.
In images with noise or complex overlapping shapes, the detection accuracy decreased, and the reported reference points deviated from the true center of the circles.

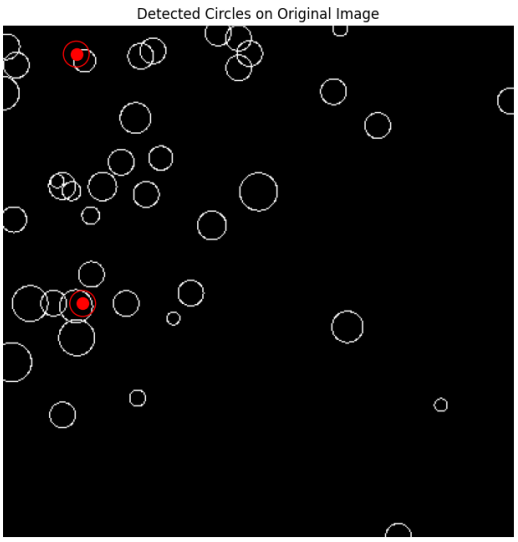

Figure 8: Hough Transform

Conclusion

Based on the results obtained throughout this project, the implemented algorithms for edge detection, boundary tracing, Harris corner detection, and Hough transform produced generally effective outcomes, with some limitations.
The gradient-based edge detector successfully identified strong edges in most images, especially when the edges were sharp and well-defined.
The boundary tracing algorithm was able to follow object contours accurately in simpler images, but struggled with broken or incomplete boundaries, especially in noisy environments.
The Harris corner detection method effectively identified corners in geometric shapes, but was sensitive to noise and often produced false positives in textured regions.
The Hough transform results for both line and circle detection were promising in controlled scenarios. The line detection algorithm reliably identified prominent lines, while the circle detection was accurate when the objects were well-defined and free from occlusions.
In conclusion, the results demonstrated that the core algorithms were effective in handling idealized cases, but encountered challenges in noisy or complex images. The project provided a valuable opportunity to explore the limitations of classical computer vision techniques, and the insights gained from the results will guide future improvements in the edge detection algorithms robustness and accuracy.