CSI 4133 – Lab 05

Detecting Circles and Lines in an Image

Contents

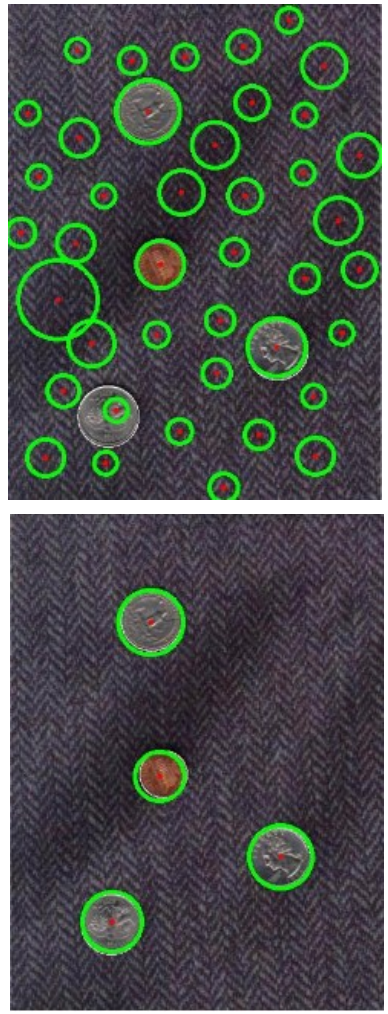
Introduction to

- Detecting Circles in an Image
- Detecting Lines and Line-Intersections in an Image

Detecting Circles in an Image

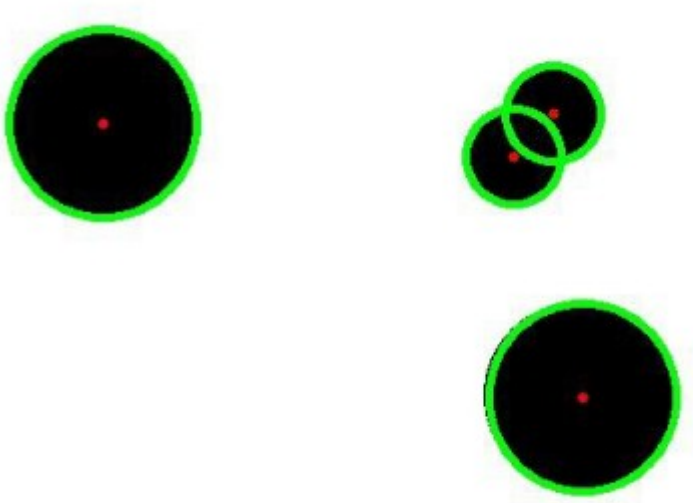
Procedure

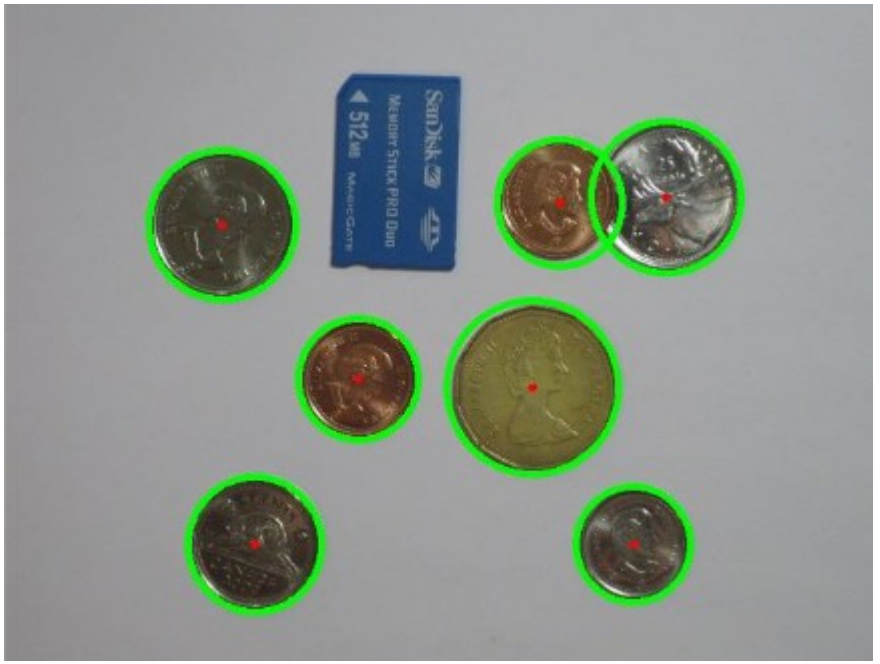
- Read the input image
- Convert from RGB image to intensity image
- Image filtering
  - Reduce noise
  - Avoid false circle detection
- Apply Circle Hough Transform to detect circles
- Display the result



Detecting Circles in an Image

Examples

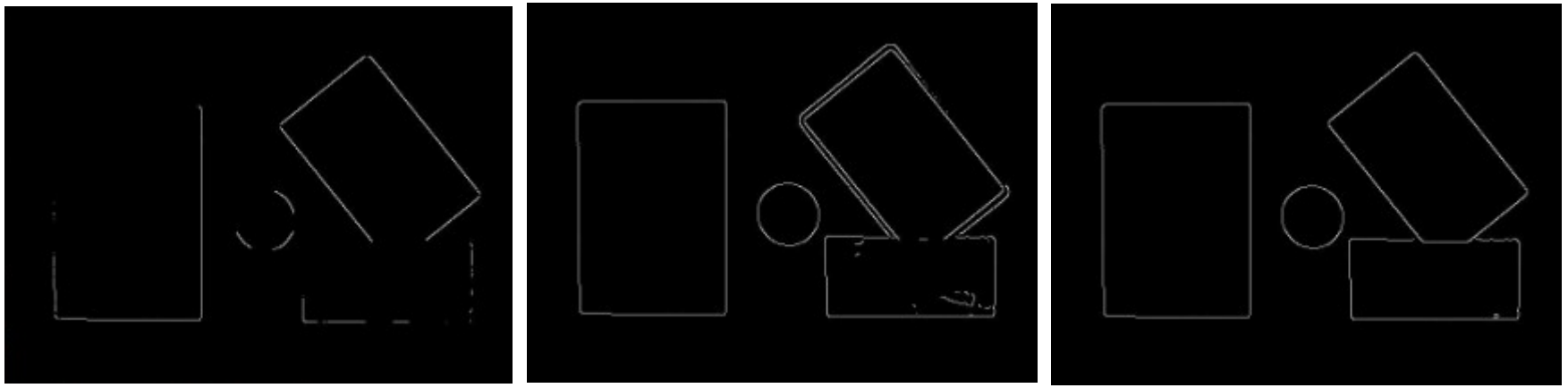




Detecting Lines and Intersections

Procedure

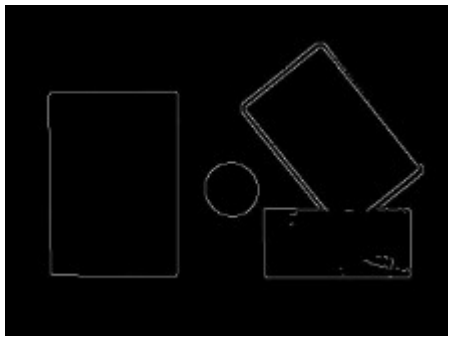
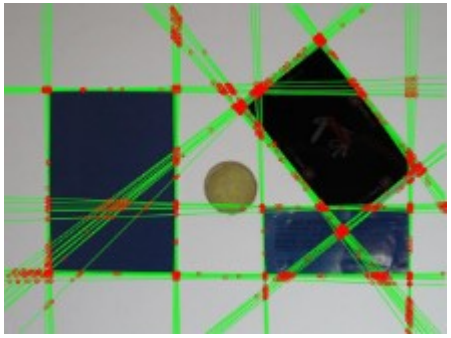
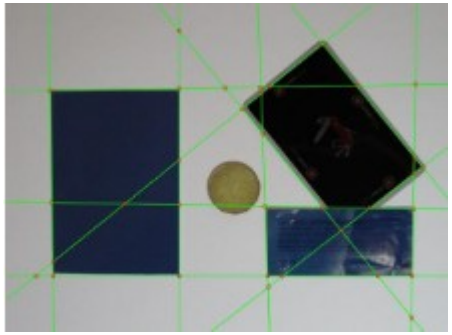
- Read the input image
- Convert from RGB image to intensity image
- Image filtering
- Edge detection (e.g. Canny)
  - Parameters of Canny edge detection method are very important!



Detecting Lines and Intersections

Procedure

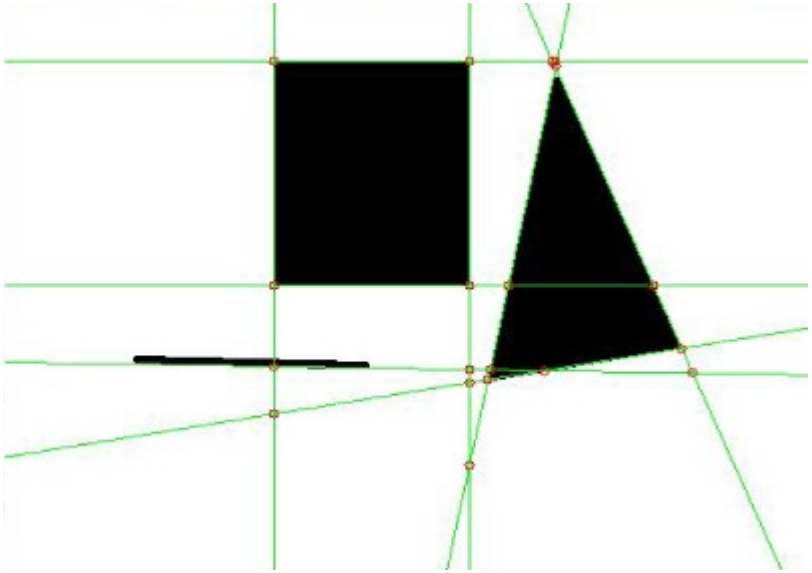
- Apply Line Hough Transform to detect lines in the image
  - Refine the detection results

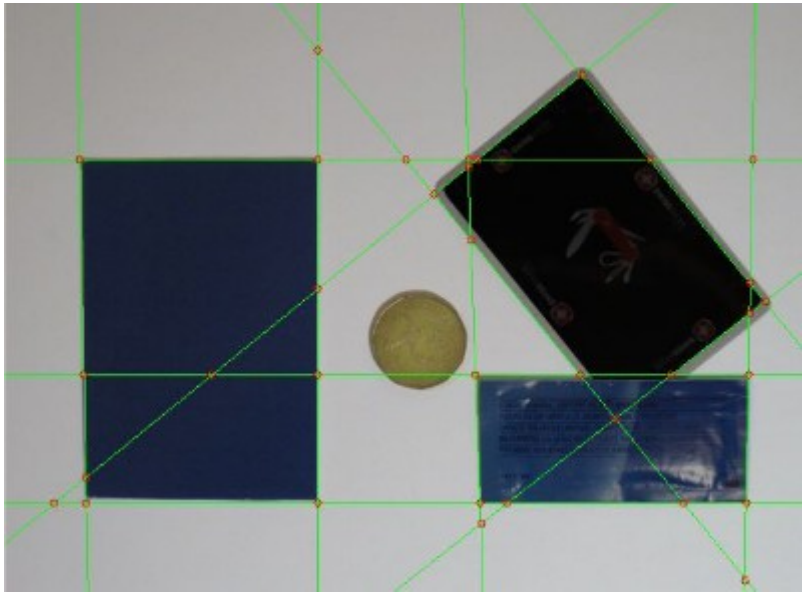




- Calculate the intersection points between each line
- Display the result

Detecting Lines and Intersections

Examples





Task (Part A)

Goal: Identify objects in an image whose contours have a circular shape.

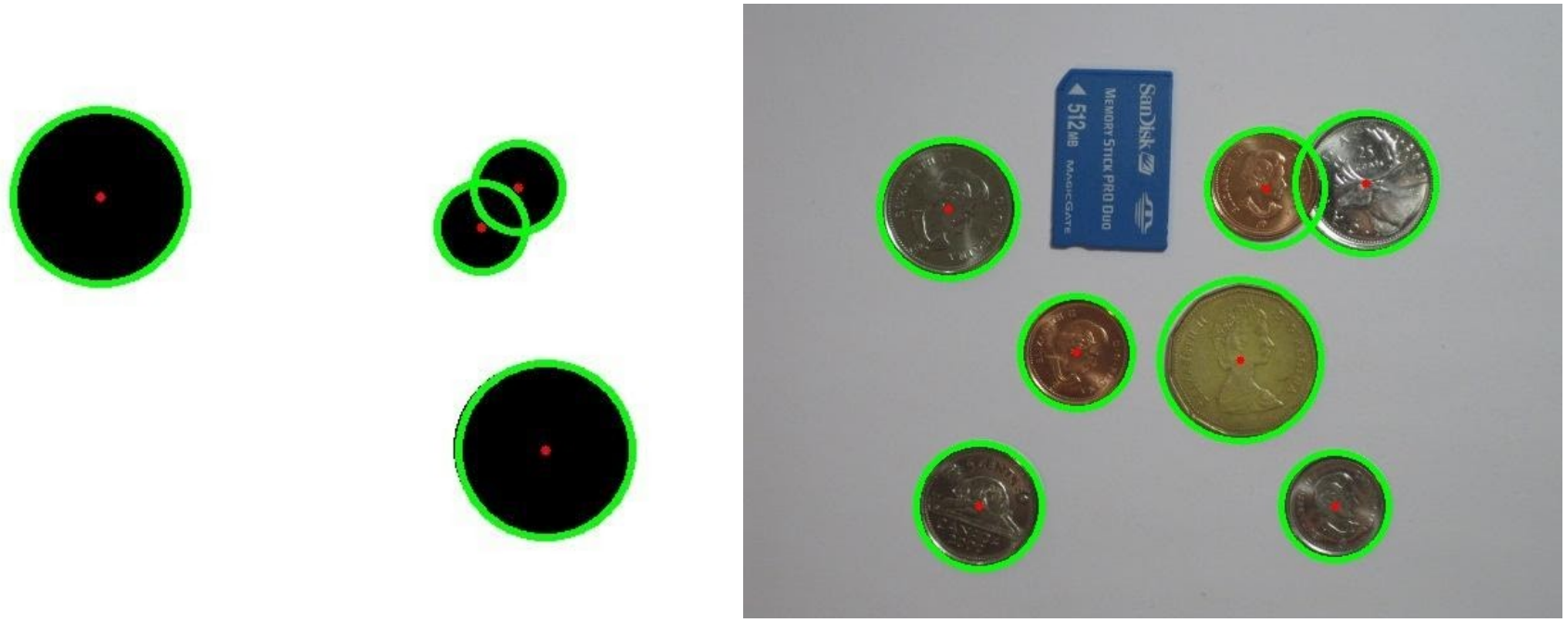


Figure 1. Circle Detection Results
(left: circles_simple.png. right: circles_target.jpg)

Task (Part A)

Idea:

1. Load Image (folder “images”).
2. Isolate objects with a circular shape.
3. Draw red circles at the centers of these isolated objects and draw green circles around these isolated objects.
4. Visualize the result.

Hints:

- OpenCV comes with a built-in method for detecting circles.
- Finding the right parameters for the above method can be key to getting good results for your solution.
- Post-processing the results from the built-in method can also improve overall results.
- There should be only one circle per object and no circles for non-circular objects.
- Initially train your solution on the image “circles_simple.png”.
- The quality of your solution will be graded based on your results from “circles_target.jpg”.

Task (Part B)

Goal: Identify object-contours in an image that resemble a straight-line and identify the points at which they intersect with each other

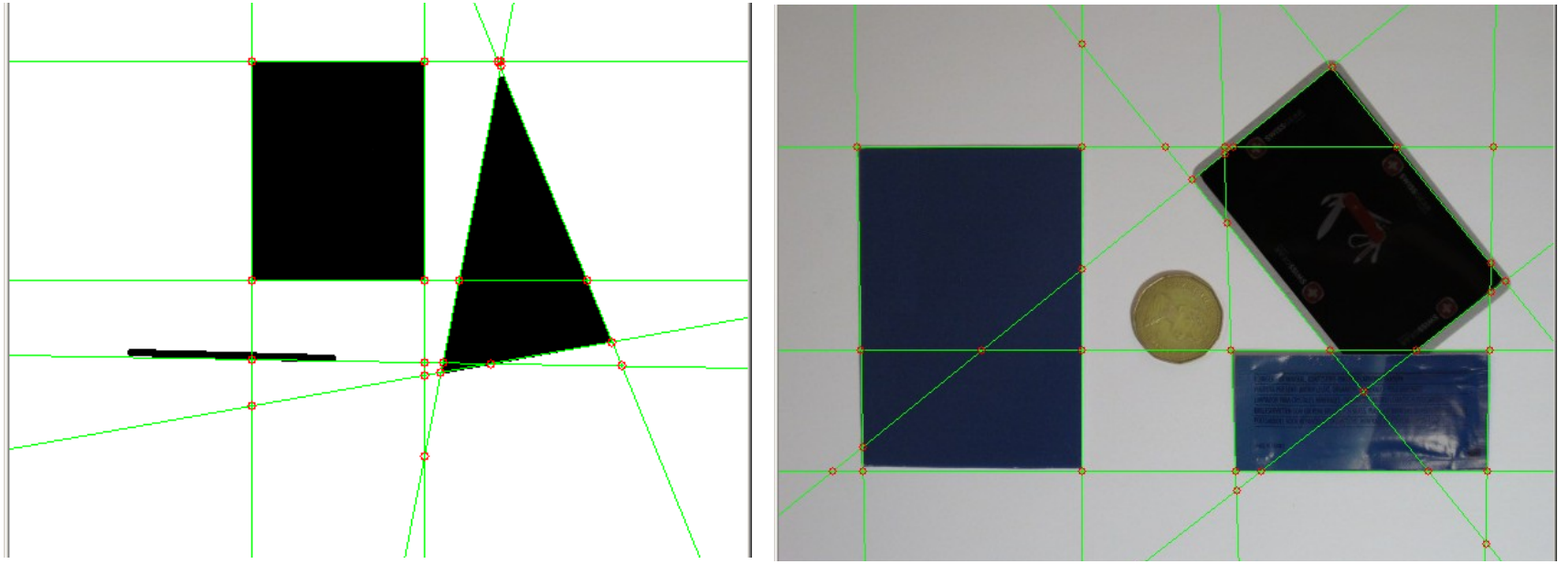


Figure 2. Line Detection Results
(left: lines_simple.png. right: lings_target.jpg)

Task (Part A)

Idea:

1. Load image (folder “images”).
2. Isolate object contours resembling a straight line.
3. Draw a green line across the image along each of these isolated contours.
4. Calculate the intersection points between each line.
5. Draw a small red circle around each of these intersection points.
6. Visualize the result.

Hints:

- OpenCV comes with a built-in method for detecting lines.
- Some pre-processing of the original image is required before using the above method.
- Post-processing the results from the built-in method can improve overall results.
- There should be only one line per straight object side and no lines for circular objects.
- Initially train your solution on the image “lines_simple.png”.
- The quality of your solution will be graded based on your results from “lines_target.jpg”.

Task

Please submit a **lab report**, **source code**, and **screenshots** of your results.

END

THANK YOU