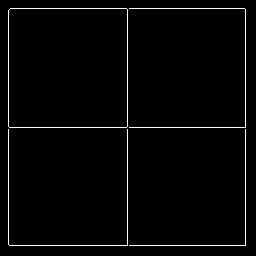
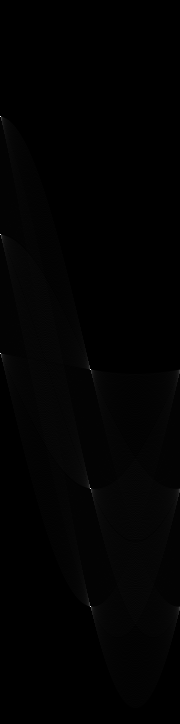
**Homework Assignment 1: Edges and Lines**

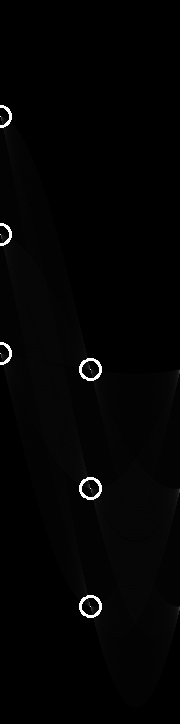
1. Edge Detection
   1. Ps1-1-a-1



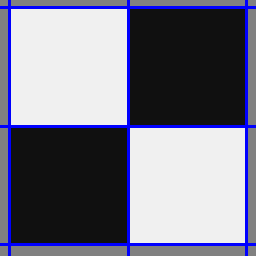
1. Line Detection – Vertical and Horizontal Lines
   1. Ps1-2-a-1



* 1. Ps1-2-b-1



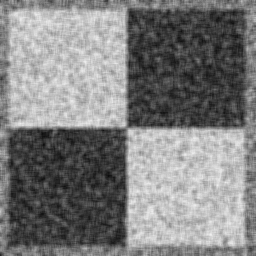
* 1. Ps1-2-c-1



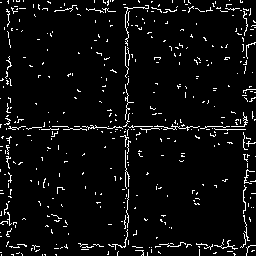
* 1. What parameters did you use for finding lines in this image?

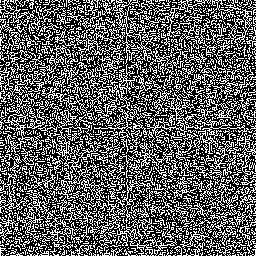
**All the parameters were kept standard, except for Canny Edge Detector’s minimum and maximum values. They were set to 100 and 200 respectively. Default accumulator’s bin sizes, threshold, and neighborhood size worked well because vertical and horizontal lines are easier to identify, especially when they are far apart from each other.**

1. Line Detection on Noisy Image – Vertical and Horizontal Lines
   1. Ps1-3-a-1

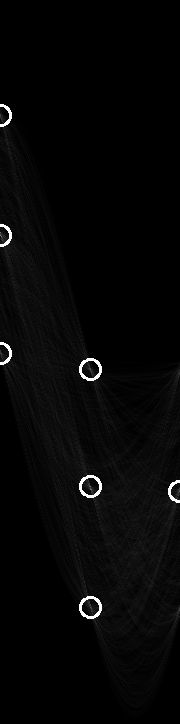


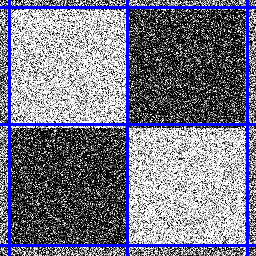
* 1. Ps1-3-b-1/2





* 1. Ps1-3-c-1/2





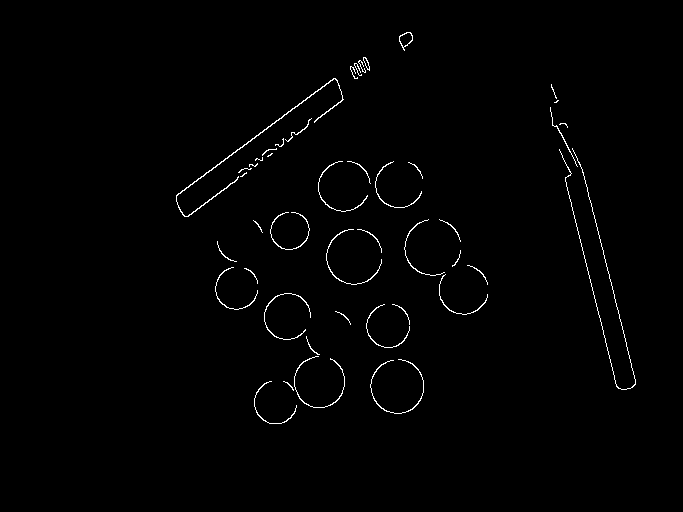
* 1. Describe what you had to do to get the best result you could.

**For Gaussian blur, I had to change my sigmaX and sigmaY to be a value of 30 with a [5 x 5] kernel to get a smooth image while still leaving the visibility of the outside borders. Canny edge detection was changed from the previous task and min and max values were set to 155 and 200 respectively to help avoid picking up extra noise in the middle of the squares and have a thinner edge on the outside. For hough\_peaks, nHoodSize had to be variated to [18, 5] to get rid of extra lines in the same neighborhood.**

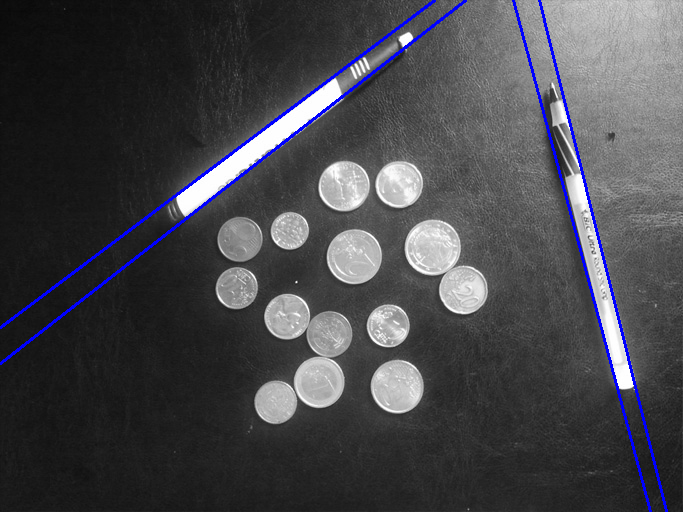
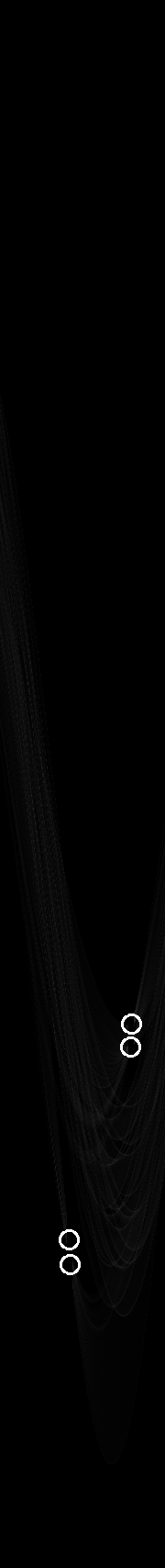
1. Line detection – Slopped Lines
   1. Ps1-4-a-1



* 1. Ps1-4-b-1



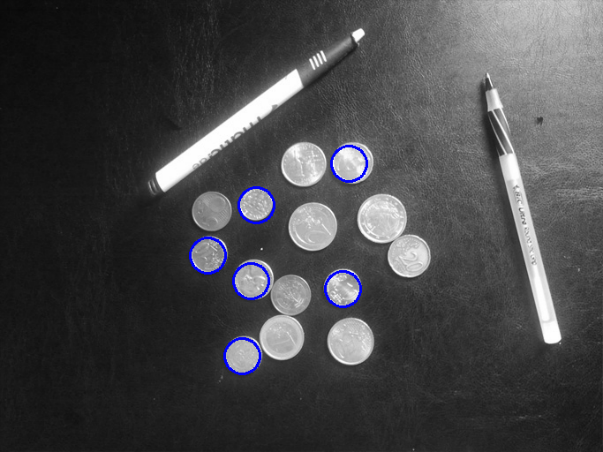
* 1. Ps1-4-c-1/2

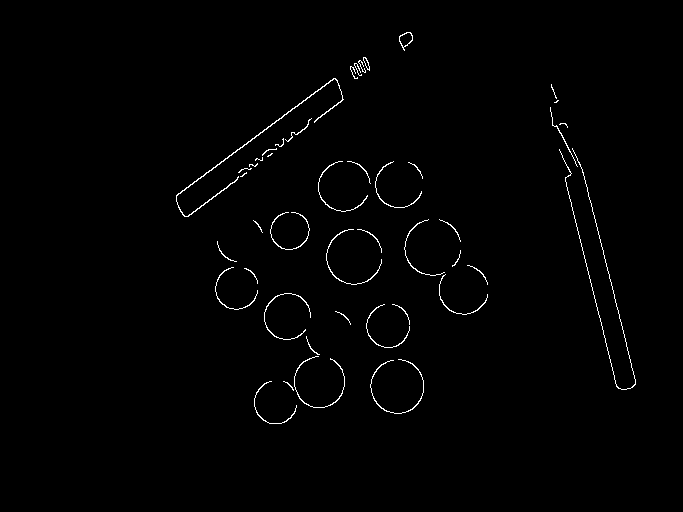
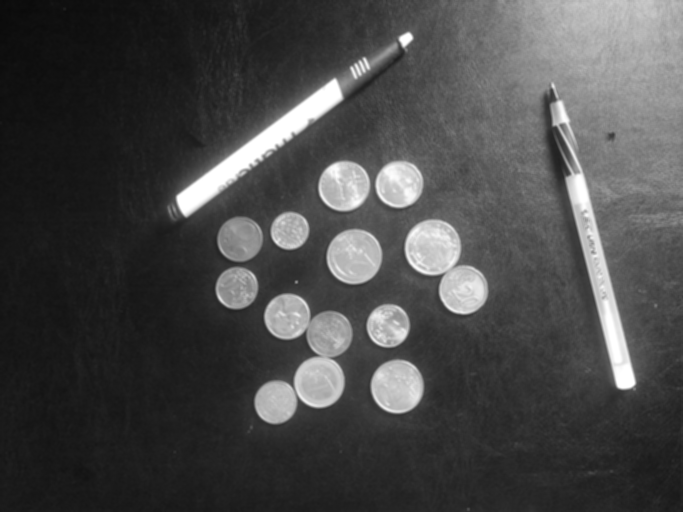


* 1. Describe what you had to do to get the best result you could.

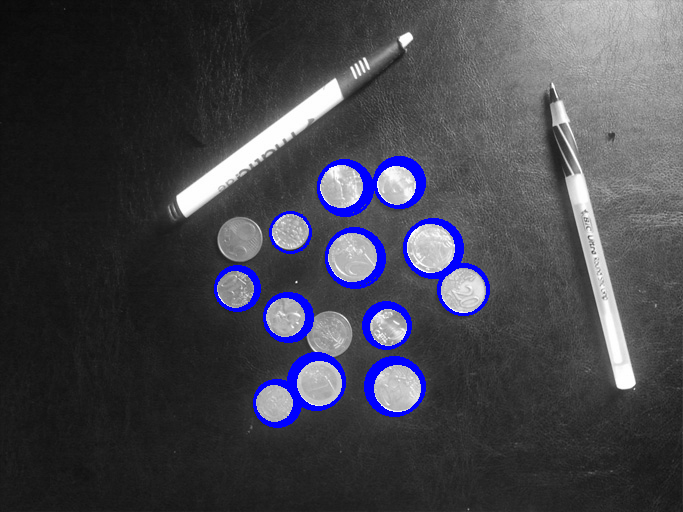
**For this problem, I selected a smaller kernel [3 x 3] and a smaller sigma to help me smooth out the image slightly without losing the contours of the lines. I used a larger set of min and max values for Canny Edge Detection to get rid of extra noise inside the pens and to take attention away from the coins. I used nHoodSize [30, 30] to decrease the neighborhood and don’t get rid of our second pen line by a mistake. I also used threshold of 60.**

1. Circle Detection
   1. Ps1-5-a-1/2/3

****



* 1. Ps1-5-b-1



* 1. Describe what you had to do to find circles.

**To find the circles, I tested every edge point at the provided radius(es) and selected the top circle centers that received the most votes. When dealing with a range of radiuses, I simply had to adjust my neighborhood size and threshold more to be able to get rid of all the extra circles. For a single radius, I used neighborhood [30, 30] and threshold of 85. For multiple radiuses, used neighborhood [30, 30] and threshold of 100.**

1. Finding pen lines - busy background
   1. Ps1-6-a-1



* 1. Likely the last step found lines that are not the boundaries of the pens. What are the problems present?

**The problem is that other objects create a stronger edge and get more votes than the pens. If the threshold was increased, we would first loose pen edges before all the other ones.**

* 1. Ps1-6-c-1

1. Finding circles – busy background
   1. Ps1-7-a-1
   2. Are there any false alarms? How would/did you get rid of them?
2. Sensitivity to distortion
   1. Ps1-8-a-1
   2. What might you do to fix the circle problem?