Laura Machlab
MP5 Report
Computer Vision
April 27, 2023

Task

The task for this MP was to use the Canny edge detector algorithm to create a function that outputs a binary image showing the edges from an input image.

Algorithm

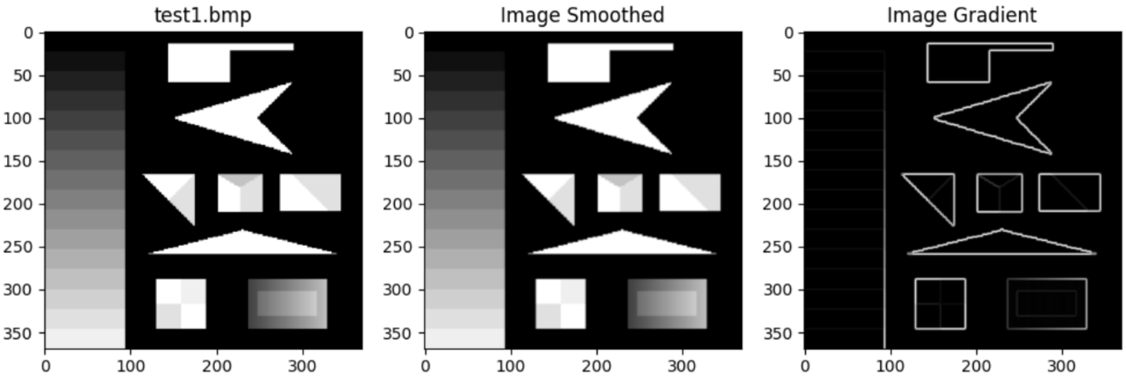
This algorithm has five steps. (1) Noise reduction - In this step, the image is converted to grayscale and smoothed to get rid of noise that might interfere with identifying edges. To do this, a Gaussian kernel is applied to filter the image. (2) Gradient calculation - This step takes the smoothed image and calculates the gradient. To do this, the image is convolved with two different sobel kernels

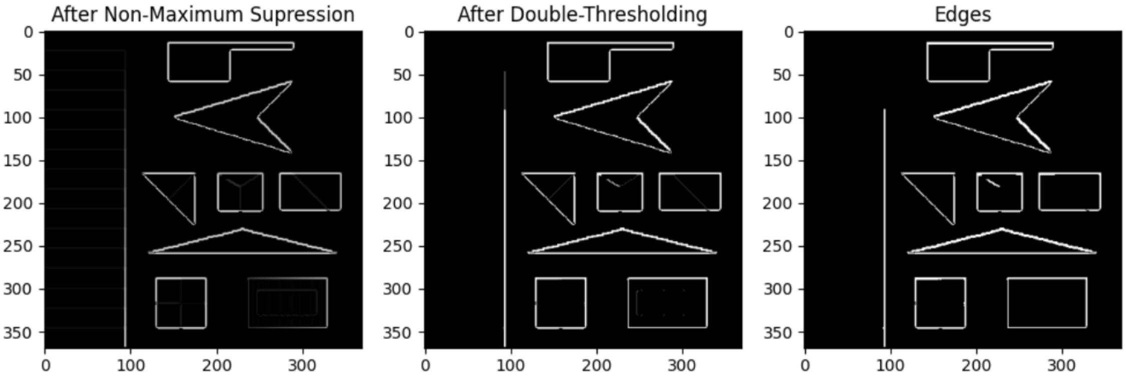
$$\text{Sobel\_x} = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix} \qquad \text{Sobel\_y} = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$

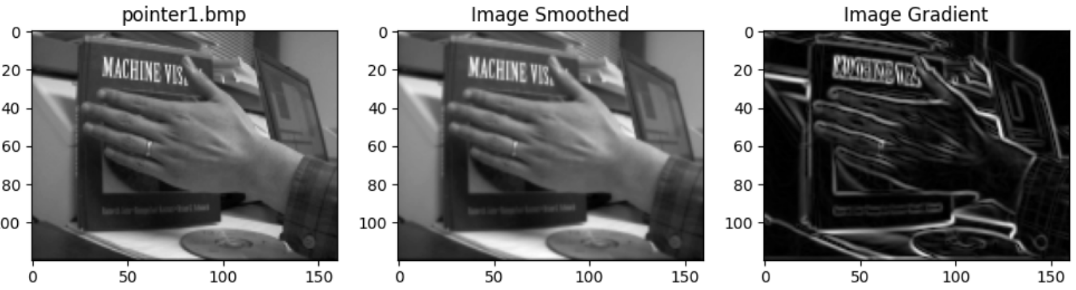
This creates an x gradient and y gradient, and from this, the combined magnitude and direction of the smooth image is calculated. (3) Non-maximum suppression - here we take the magnitude and direction information of the image and step through the image. At each pixel, the magnitude of the surrounding pixels in line with the edge direction are compared to the magnitude of the pixel of interest. If the magnitude of either of the neighboring pixels are greater than that of the pixel of interest, then only the greatest magnitude pixel is kept and the magnitude of the central pixel is set to 0. If neither of the neighboring pixels have a higher magnitude, then the central pixel is kept the same. (4) Double threshold - this step identifies strong, weak, and irrelevant pixels within the suppressed image. Based on two defined thresholds – high and low – each pixel is stepped through, and if it has a value above the high threshold, it is given a magnitude value of 255. If the pixel value is between the high and low threshold, it is given a value of 75, and if the pixel value is below the low threshold, it is set at 0. This step produces an image that has strong lines and weak lines (with values of 255 and 75 respectively. (5) Edge tracking by hysteresis - in this last step, the strong and weak thresholded image is tracked to determine which weak edges should be maintained. To do this, we step through each pixel and take note of the values of the eight surrounding pixels. If one of these eight is a strong pixel (value of 255), then the pixel of interest is converted to a strong pixel. Otherwise, it is set to 0. The output of this step is the final product of the edge detection.

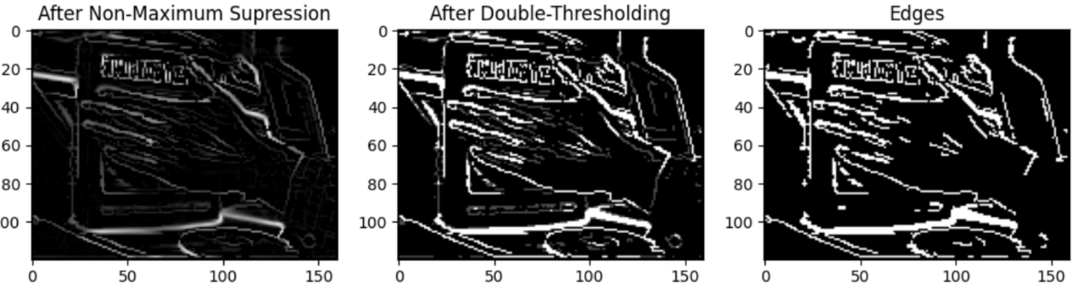
Results

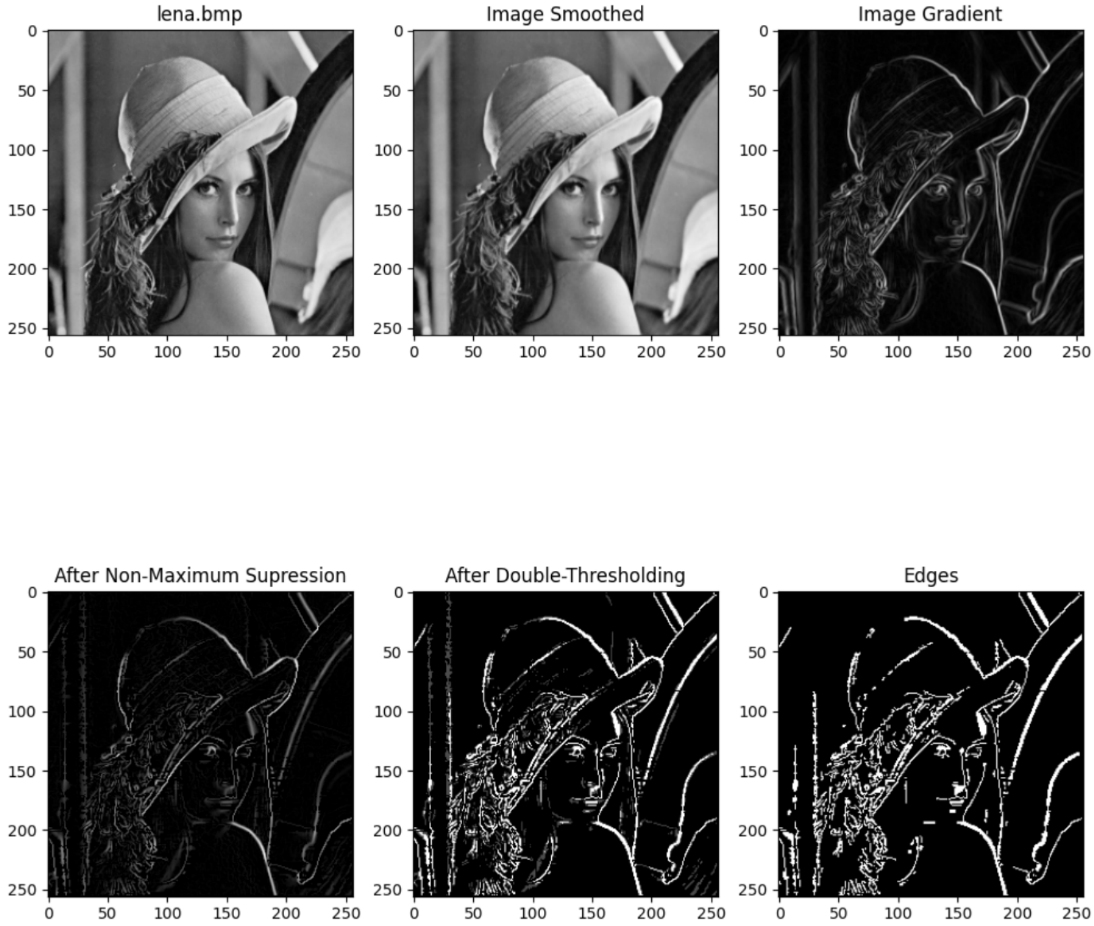
After applying this algorithm to the four test images provided, here are the results.

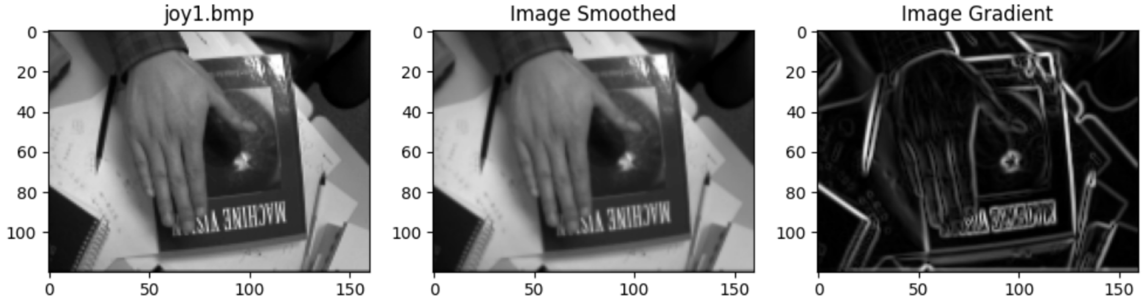


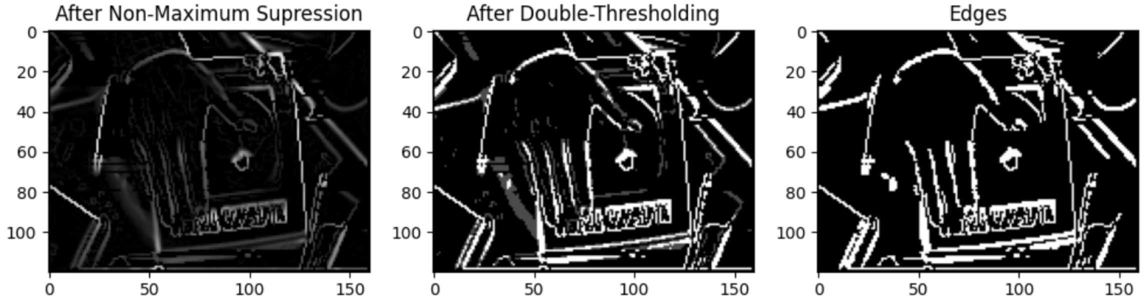



pointer1.bmp
Image Smoothed
Image Gradient
MACHINE VIS
0
20
40
60
80
100
0
50
100
150

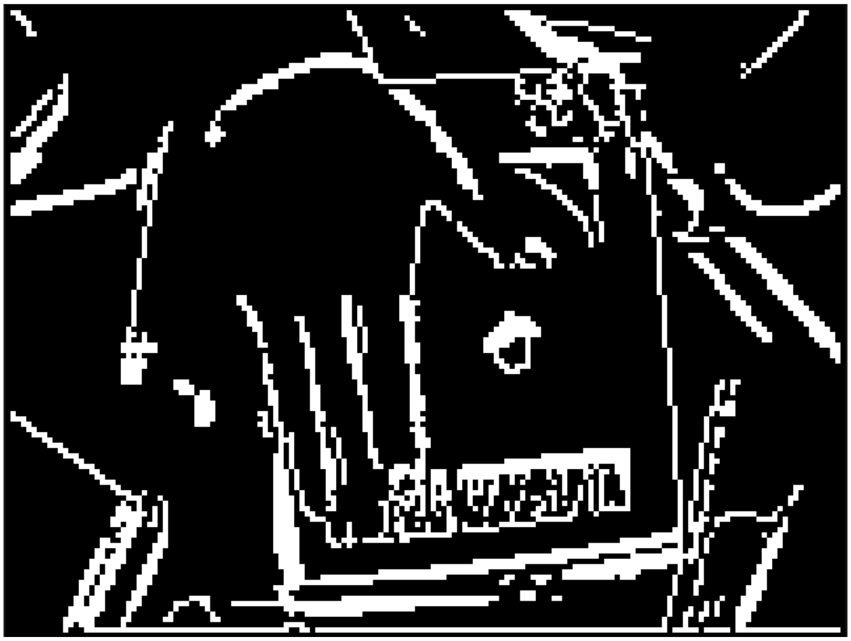

After Non-Maximum Supression
After Double-Thresholding
Edges
0
20
40
60
80
100
0
50
100
150


lena.bmp
Image Smoothed
Image Gradient
After Non-Maximum Supression
After Double-Thresholding
Edges
0
50
100
150
200
250

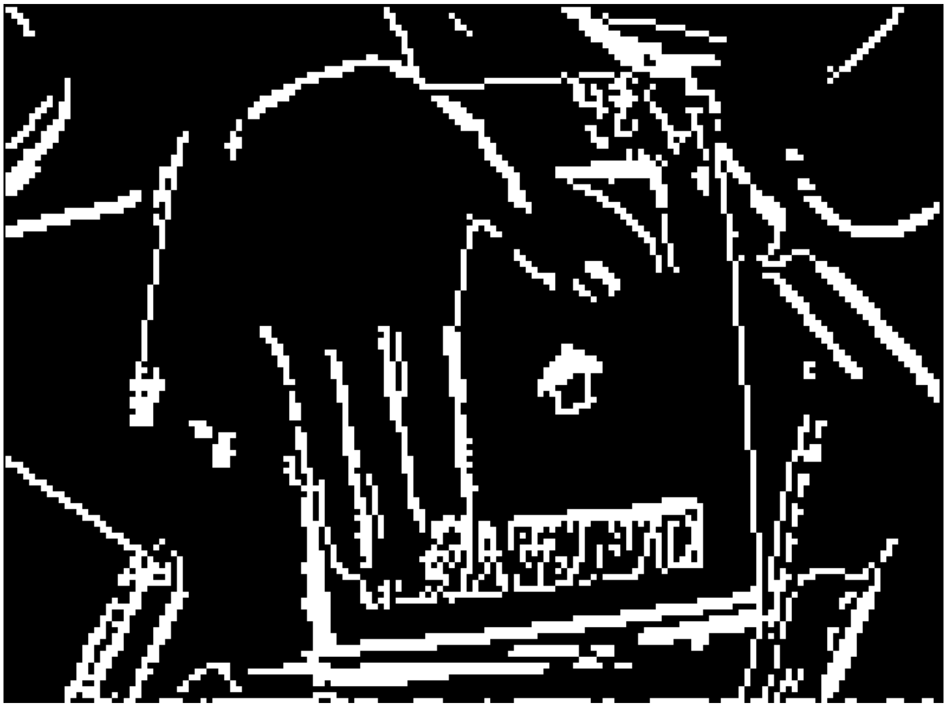




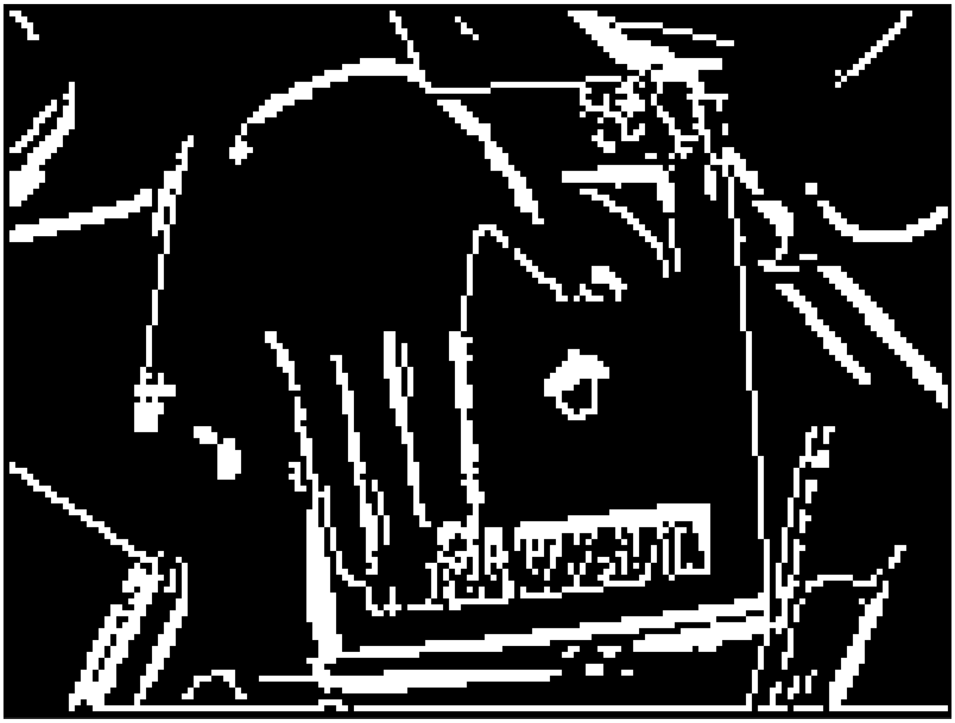
These first four examples had a gaussian kernel size of 5x5, a sigma value of 0.5, and double thresholding high ratio and low ratio values of 0.21 and 0.11 respectively. This is the outcome for the joy edges specifically:



Here is what it looks like when sigma is changed from 0.5 to 0.1:

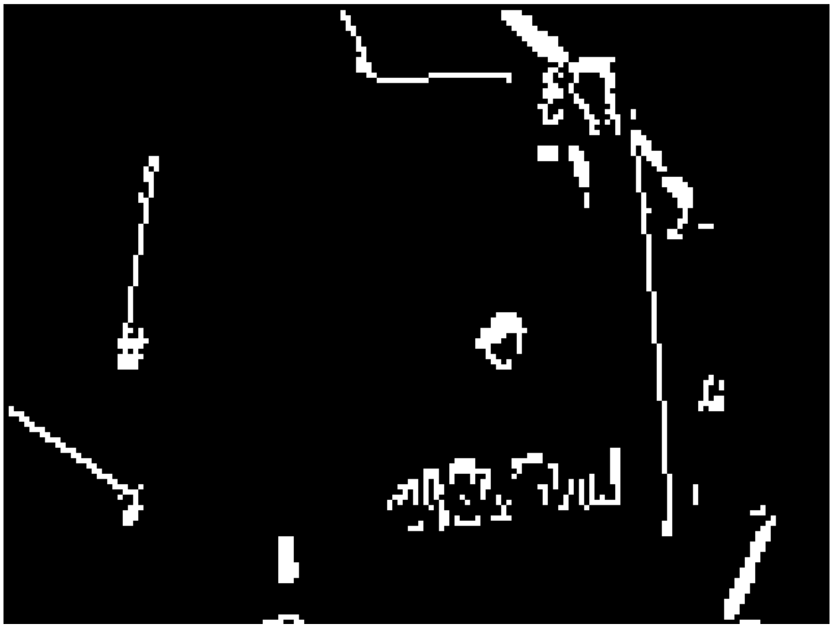


And this is what happens when the kernel size is increased to 7x7:



Both increasing the kernel size and decreasing the sigma value appear to decrease the amount of edges identified slightly.

I then tried changing the high and low ratio values. When I increased them both – with the high at 0.5 and low at 0.15 – I noticed a decreased amount of detail captured in the edges.



And when I decreased these values – high at 0.15 and low at 0.05 – I noticed an increase in edge details.