Hasbi Sevinç
504161518

BLG 513E Image Processing Canny Edge Detection Algorithm

Homework #1

1) Introduction

In this project, a Canny Edge Detector has been implemented without using any image processing library such as openCV. To detect edges, some image processing methods have been implemented. These method can be investigated in six section

Python Dependencies
To use this script, you need to have these libraries on your computer

- PIL.Image
- scipy.misc and scipy.stats
- numpy

How to Use

- Open a terminal window
- Navigate to root directory of project (directory which contains canny.py file)
- python canny.py

Troubleshooting

- Make sure you installed all required libraries
- Make sure you have images into “images” folder in the project root directory

2) Implementation Steps of Canny Edge Detection

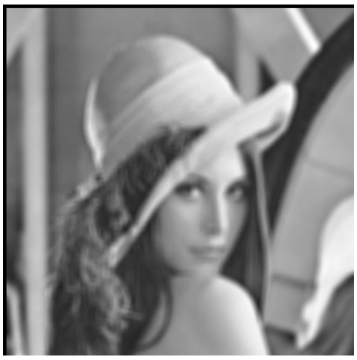
a) Converting to Grayscale

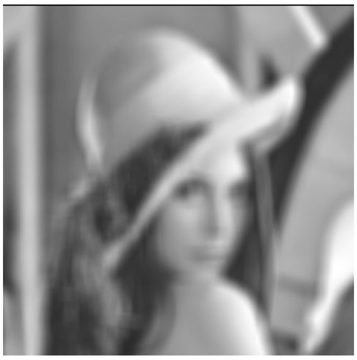
When an image opened by python script, this image may contains red green and blue values in all pixels. To make script faster, we need to combine all this values into one value. Thus, first of all, python script convert image into grayscale. On the other hand, image may already grayscaled before.

There are some methods to convert pixel into grayscale such as averaging color values with same intensity. But that approach is not enough to get nice grayscaled image. In the project, averaging with parameters (0.2126 *RED + 0.7152 * GREEN + 0.0722 * BLUE) is used.

b) Smoothing

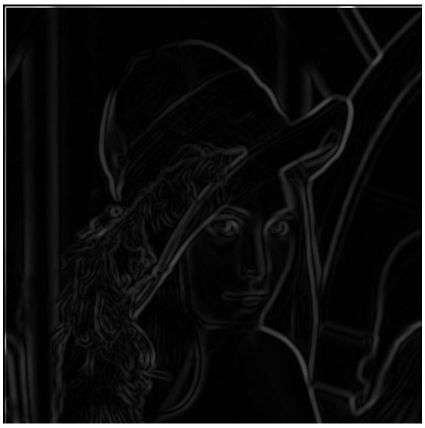
Image must be smoothed by some filters to get rid of noises. Because, these noises will make script to find edges that are not really exist. To smooth an image there are some filters to use. Gaussian filter will be used to smooth input images. This method takes sigma parameter to adjust blur effect. On below, you can see images with different parameters.

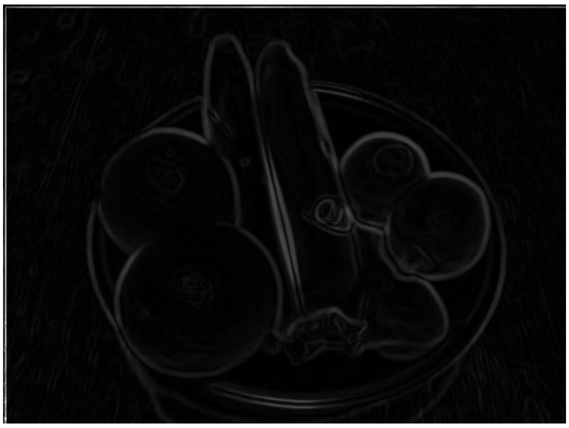




c) Finding gradients

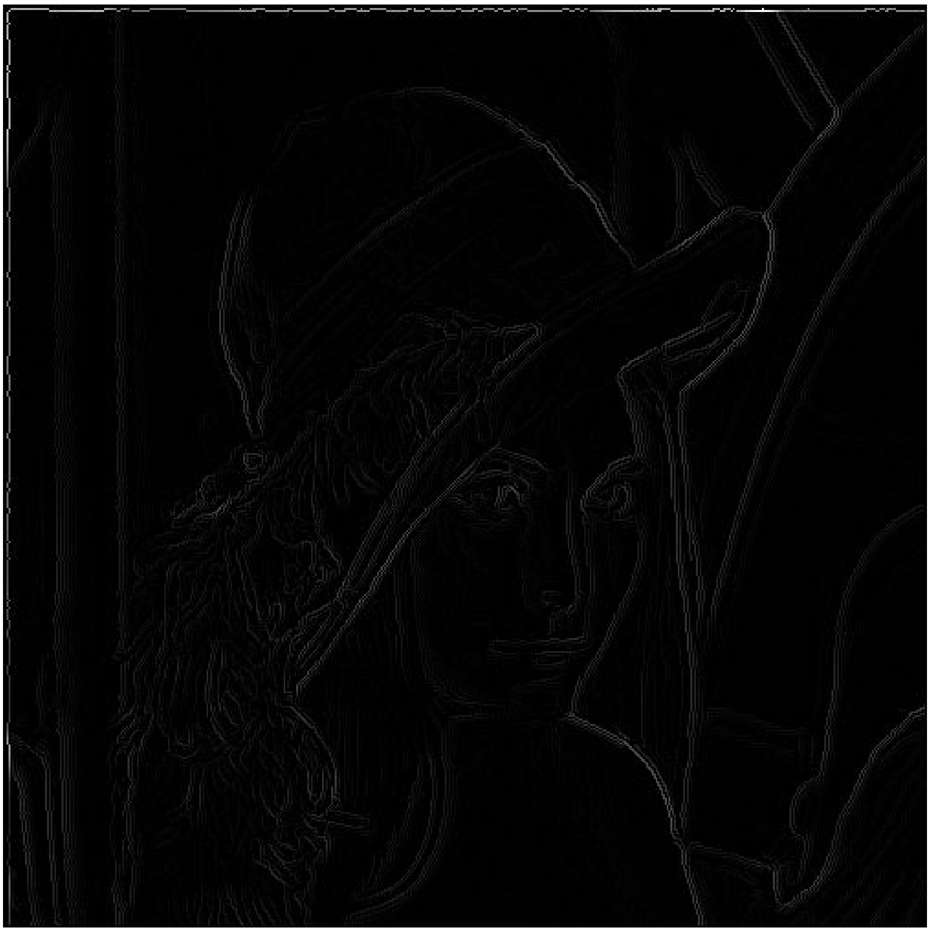
To find gradients, we can use some methods such as derivative of gaussian, sobel etc.. Sobel filter will be used sobel filter to find gradients.





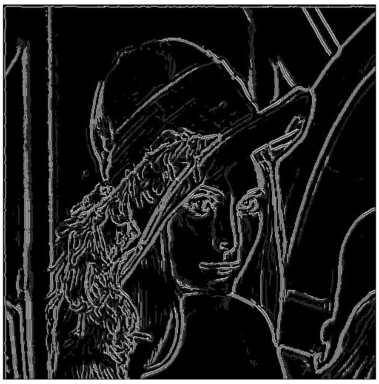
d) Non-maximum suppression

In this step, script should eliminate pixels which is not greater than neighbor pixels. gradient step outputs, magnitude and direction, will be used to suppressions. below picture shows current state after non-maximum suppression.

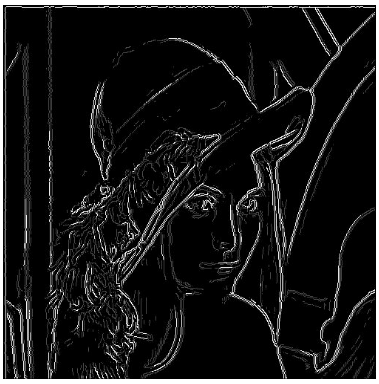


e) Double Thresholding

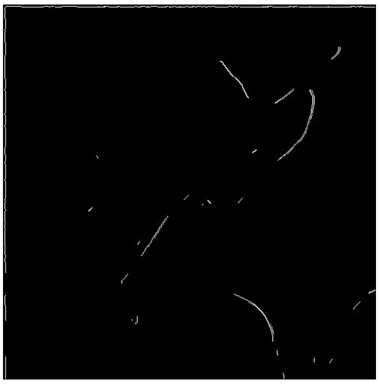
Two threshold, one of them is high and other is low, to adjust edge weight. Images with different threshold values shown below



High:40 Low:20



High:75 Low:30



High:180 Low:120

f) Linking Edges

At this step, script will suppress all edges that are not connected to very strong edge. After this step canny edge detection will be completed.

3) Output

To see full-sized output images, please look output folder in the project directory