Esin ERSOĞAN
708191001

Image Processing
Homework 1 Report

18.10.2019

1) Implementation

In this homework, a canny edge detector is implemented with the below steps:

1.1. Gaussian Filter:

Gaussian filter is applied to blur the image by removing the noise. 5*5 size Gaussian filter is applied with 1.4 sigma value. The blurred image after the filter is in the below, right one.

$I = I \times GaussFilter_{5x5}$ (x: Convolution operator)

1.2. Finding gradients:

Sobel filter is used to obtain edge intensity and edge directions.

Sx = [-1,0,1; -2,0,2; -1,0,1]
Sy = [1,2,1; 0,0,0; -1,-2,-1]

Gx = I x Sx (x: Convolution operator)
Gy = I x Sy (x: Convolution operator)

$I = sqrt(Gx^2 + Gy^2)$
angles = Gy/Gx


1.3. Non-max Suppression:

Non-max suppression algorithm is used to thin the edges.

The pixel values in the direction of that pixel are checked for each pixel value;

- ➢ If the pixel value is less than one of the corresponding pixel values, the value of the pixel is set to zero.
- ➢ The pixel value is retained if it is greater than both.


1.4. Double Thresholding:

To identify the strong, weak and non-relevant pixels double thresholding is used.

Two threshold value is defined as high threshold and low threshold.

- ➢ If the pixel's value is higher than the high threshold, which means it is a strong pixel, it is set to the 255.
- ➢ If the pixel's value is lower than the low threshold, which means it is a non-relevant pixel, it is set to 0.

➢ If the pixel's value is lower than high threshold and higher than the low threshold, which means it is a weak pixel, it is set to a defined weak value.

1.5. Edge Tracking by Hysteresis:

To follow the edge according to the weak and strong pixels hysteresis is used.

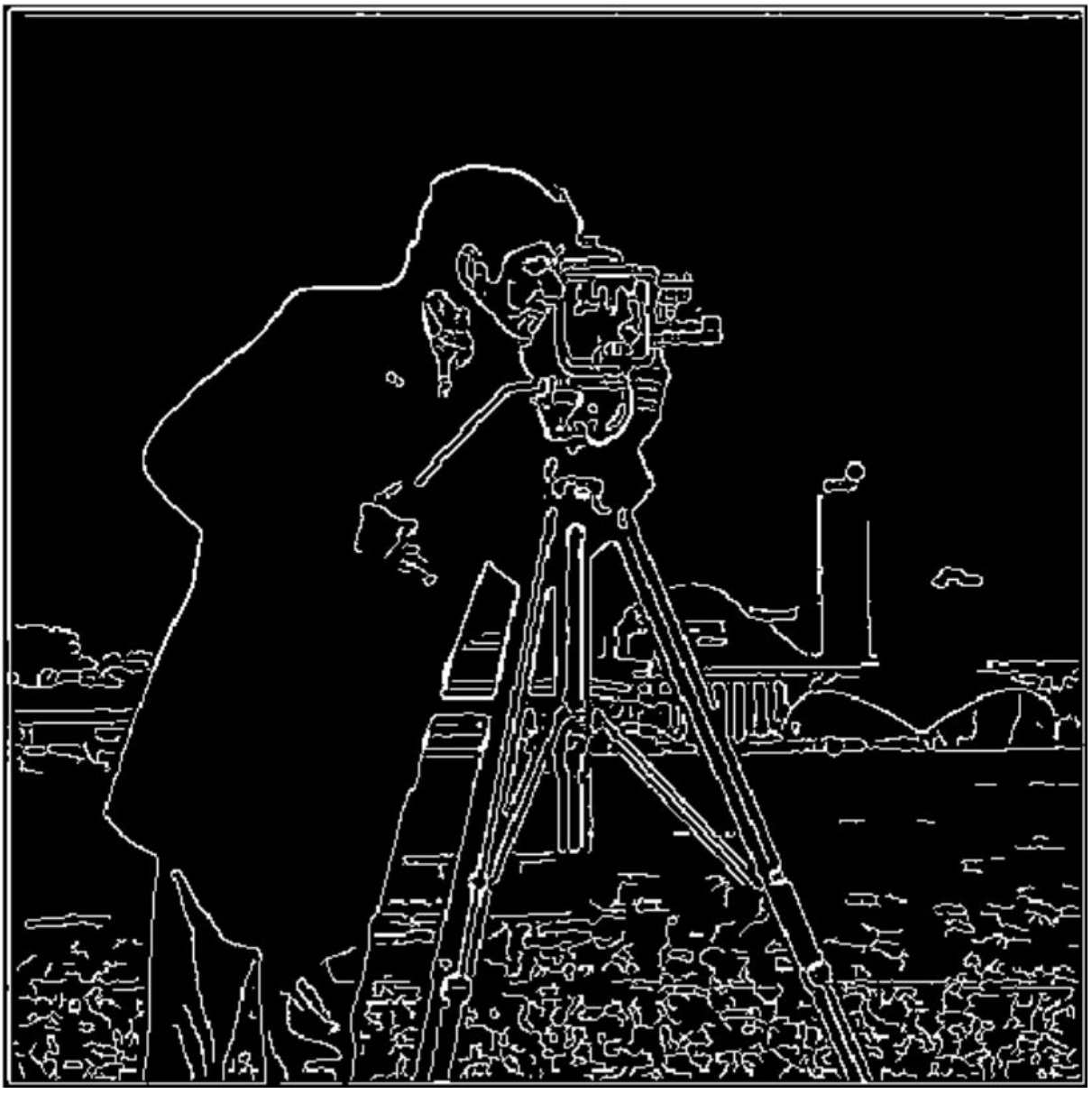
➢ If one of two pixels' intensity value is higher than the pixel, its value is set to this strong value.

➢ Else, its value is set to zero.

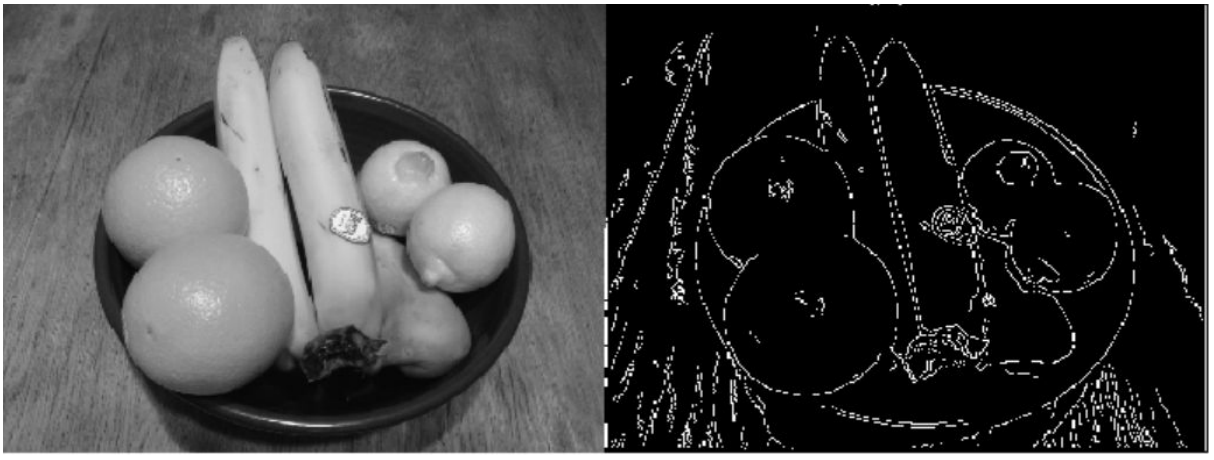
2) Results

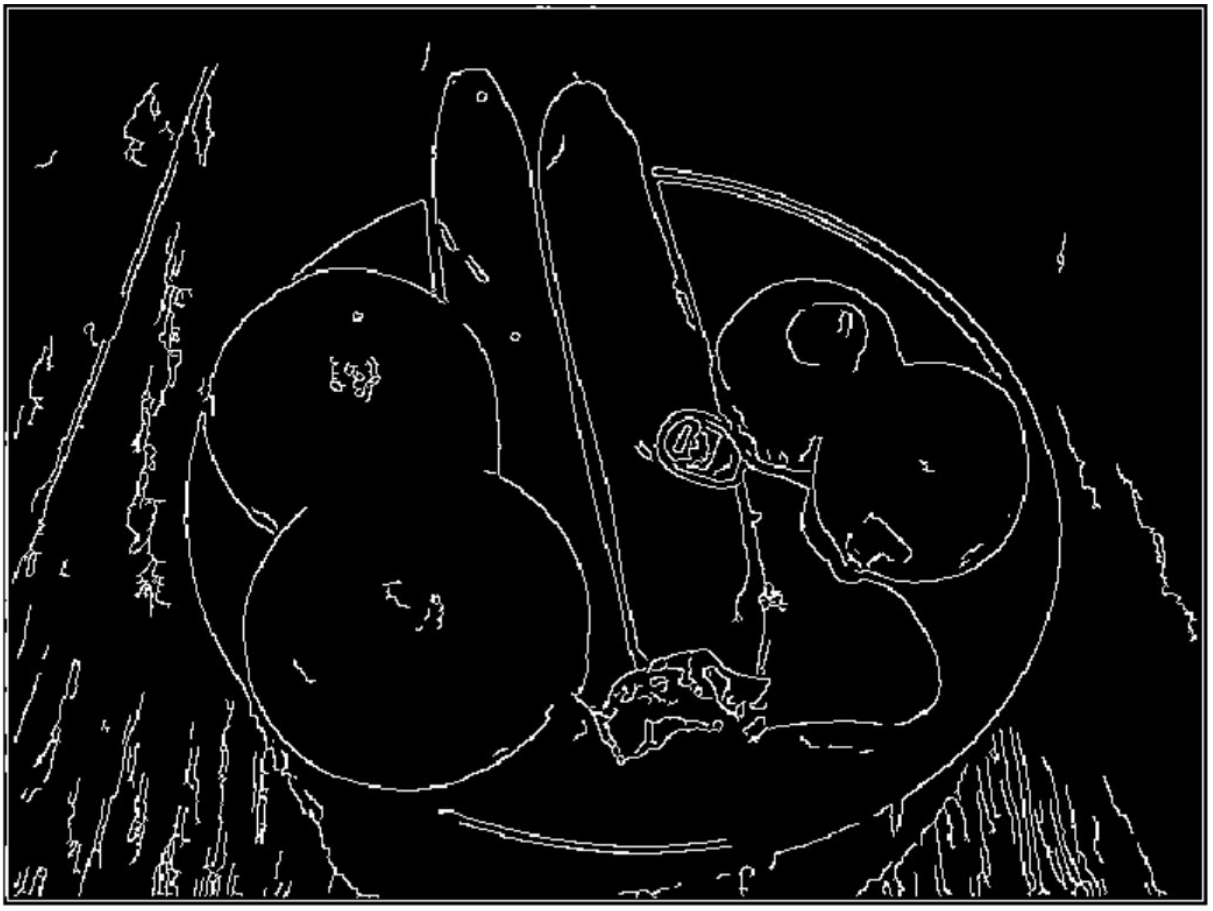
The results obtained with the Canny Edge Detection algorithm are below:

'cameraman.jpg'

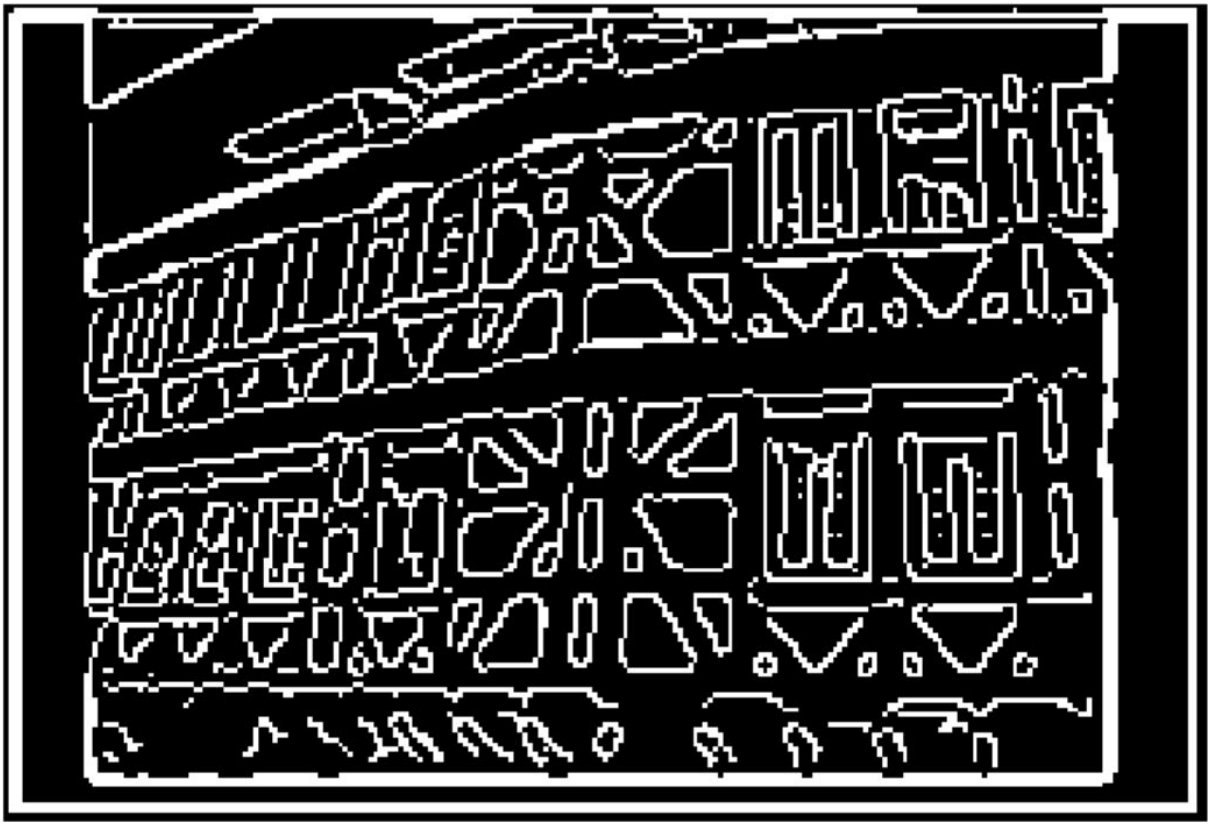


‘fruit-bowl.jpg’





‘house.jpg’



‘Lenna.png’

‘woman.JPG’

3) How to Use

You can run the edgedetect.m file with the filenames of the images as in the below:

cannyedgedetection.m | edgedetect.m

```
output1 = cannyedgedetection('cameraman.jpg');
output2 = cannyedgedetection('fruit-bowl.jpg');
output3 = cannyedgedetection('house.jpg');
output4 = cannyedgedetection('Lenna.png');
output5 = cannyedgedetection('woman.JPG');


figure, imshow(output1);
figure, imshow(output2);
figure, imshow(output3);
figure, imshow(output4);
figure, imshow(output5);


%For original image + edge detected images
% pairOfImages = [imread("cameraman.jpg"), output1];
% figure, imshow(pairOfImages);
%
% pairOfImages2 = [rgb2gray(imread("fruit-bowl.jpg")), output2];
% figure, imshow(pairOfImages2);
%
% pairOfImages3 = [rgb2gray(imread("house.jpg")), output3];
% figure, imshow(pairOfImages3);
%
% pairOfImages4 = [rgb2gray(imread("Lenna.png")), output4];
% figure, imshow(pairOfImages4);
%
% pairOfImages5 = [rgb2gray(imread("woman.JPG")), output5];
% figure, imshow(pairOfImages5);
```