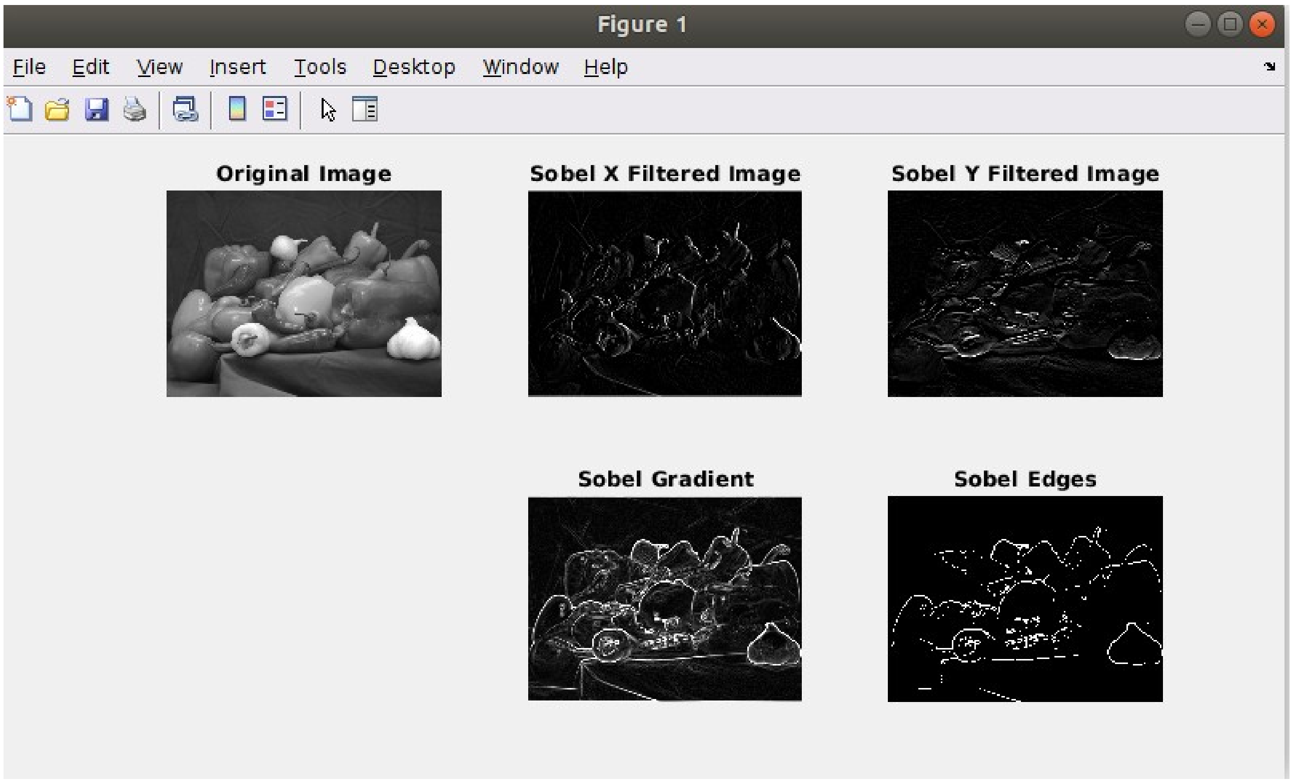
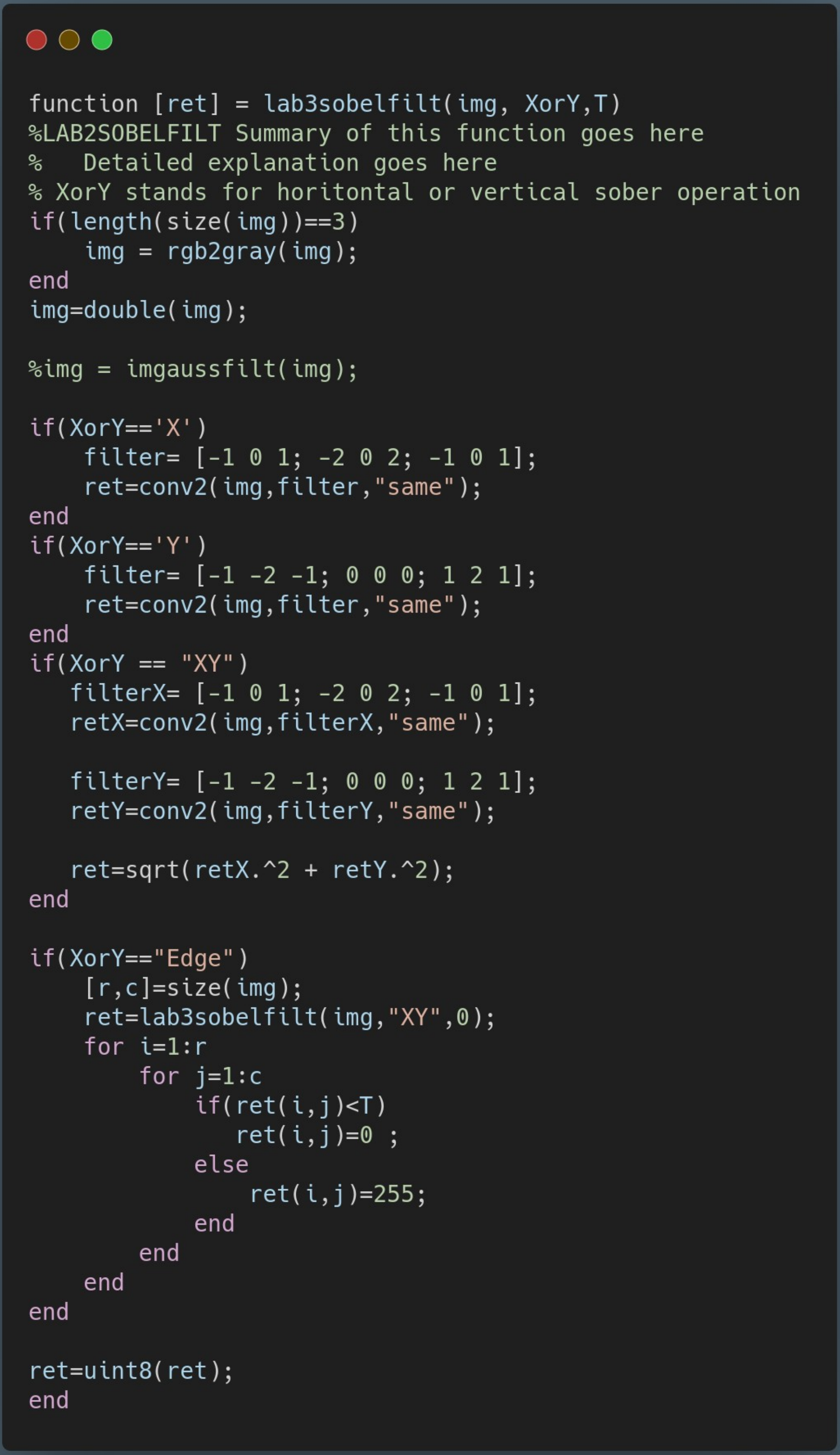
EE417 -Computer Vision Lab 3 Report
Hakan Buğra Erentuğ 23637
22 / 10 / 19

Assignment #1

In the first assignment, the sobel operator (edge detecting) is asked to us. To implement the sobel operations, previous sobel-X and sobel-Y filter has been used. The method of using edge detecting is taking square root of horizontal and vertical sobel filtered image for each pixel. In the design of the algorithm, we added two mote input which is XorY and T for threshold. XorY is a kind of sting if condiiton for the extendiblity of the function. 'X' char means use X filter and give the output, 'Y' char means use Y filter, "XY" string use square root of (X^2 + Y^2) and calculate the sobel gradient image and "Edge" means reduce the low gradients by looking the threshold and categorize each pixel by full black and full white. In our experiment, 150 threshold value has been selected. That means if any pixel value in gradient image is less than 150, equalize it to 0, otherwise equalize that pixel to 255 which is the Gmax for our 8bit test image. The higher threshold makes the detection more naive and reduce the details whereas lower threshold makes image more precise and make the image more detailed. Any smoothign operation is not been used because, the image is already noiseless and smoothing can prevent to detecting some image due to blurring. The implementation of function with command lines and results have been given below.



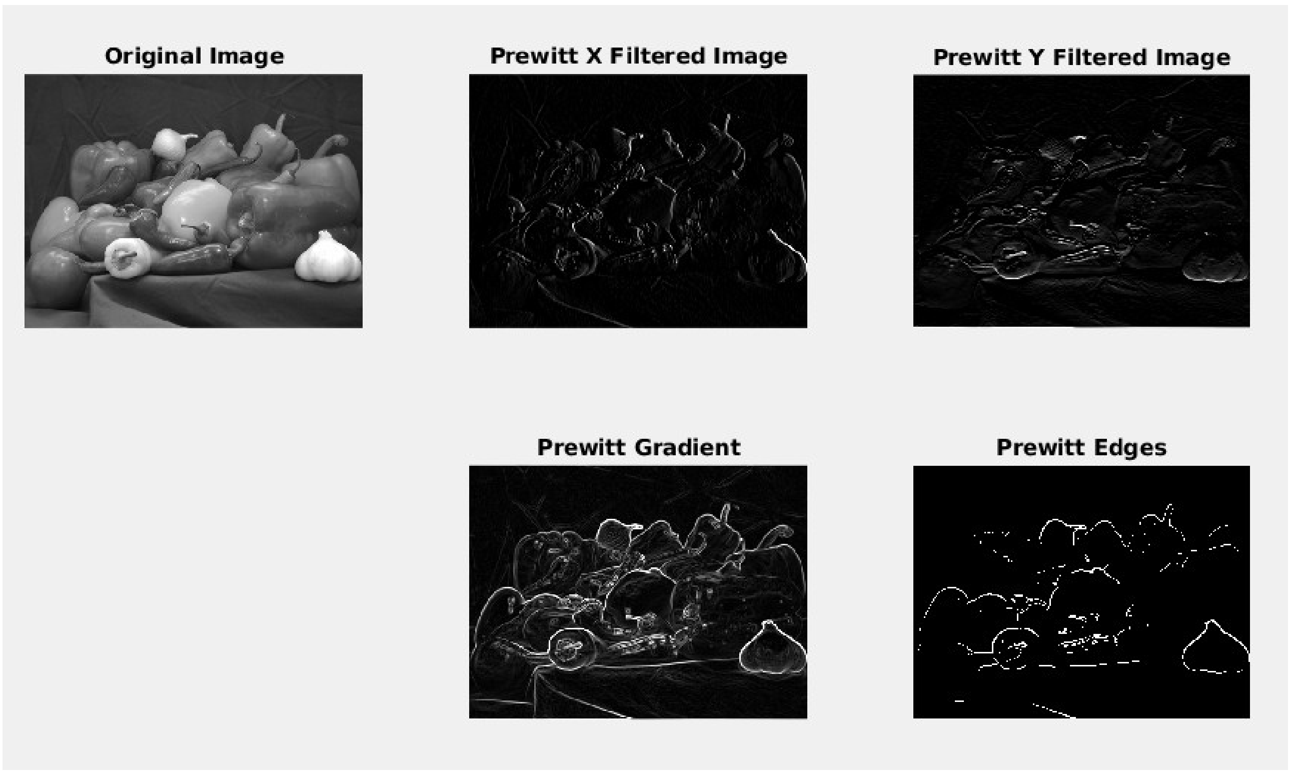


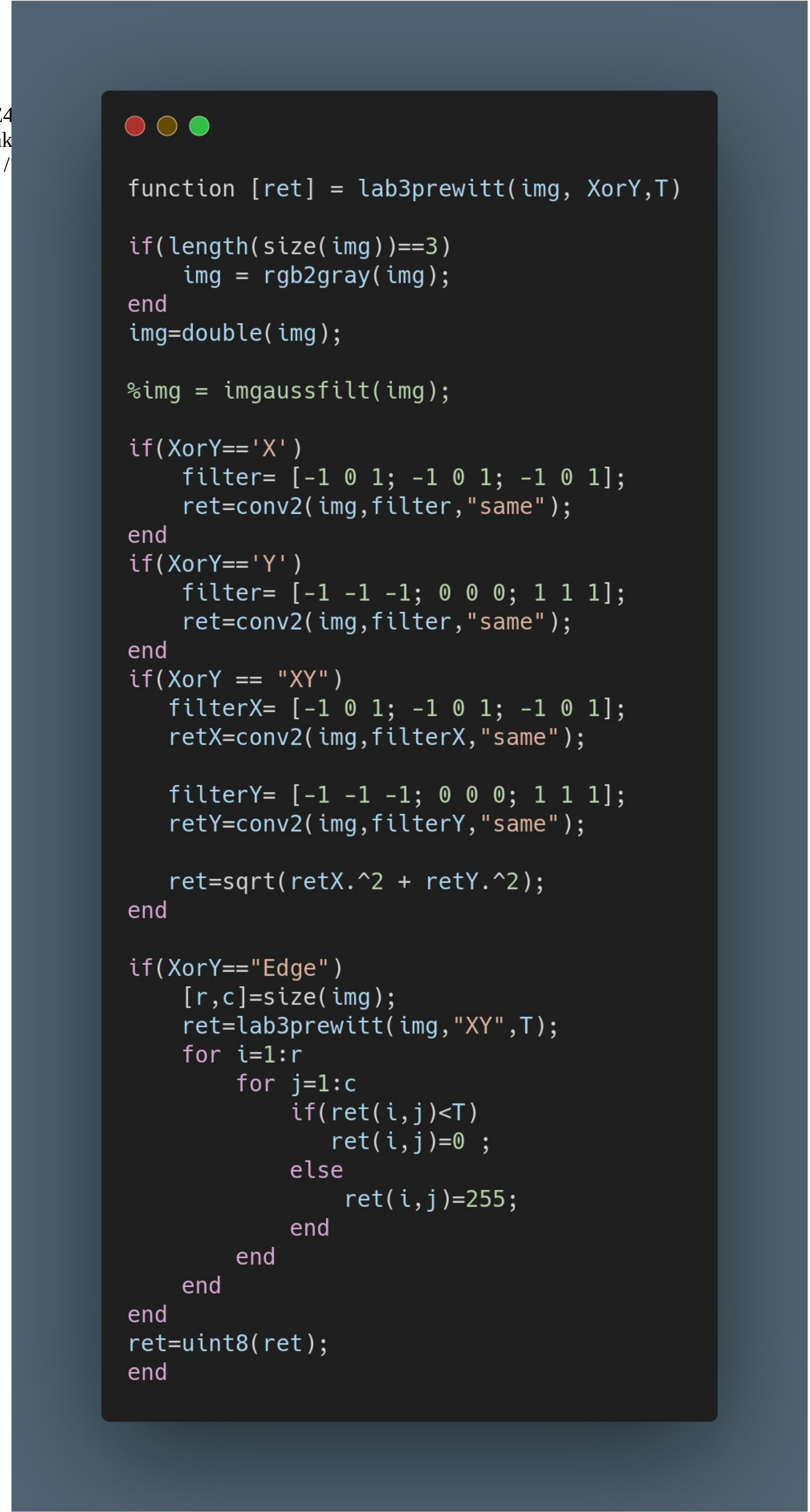
```
function [ret] = lab3sobelfilt(img, XorY,T)
%LAB2SOBELFILT Summary of this function goes here
%   Detailed explanation goes here
% XorY stands for horitontal or vertical sober operation
if(length(size(img))==3)
    img = rgb2gray(img);
end
img=double(img);

%img = imgaussfilt(img);

if(XorY=='X')
    filter= [-1 0 1; -2 0 2; -1 0 1];
    ret=conv2(img,filter,"same");
end
if(XorY=='Y')
    filter= [-1 -2 -1; 0 0 0; 1 2 1];
    ret=conv2(img,filter,"same");
end
if(XorY == "XY")
   filterX= [-1 0 1; -2 0 2; -1 0 1];
   retX=conv2(img,filterX,"same");

   filterY= [-1 -2 -1; 0 0 0; 1 2 1];
   retY=conv2(img,filterY,"same");

   ret=sqrt(retX.^2 + retY.^2);
end

if(XorY=="Edge")
    [r,c]=size(img);
    ret=lab3sobelfilt(img,"XY",0);
    for i=1:r
        for j=1:c
            if(ret(i,j)<T)
               ret(i,j)=0 ;
            else
                ret(i,j)=255;
            end
        end
    end
end

ret=uint8(ret);
end
```

EE417 -Computer Vision Lab 3 Report
Hakan Buğra Erentuğ 23637
22 / 10 / 19

Assignment #2

In the second assignment, the prewitt operator has been used. The difference between the sobel and prewitt is the kernel matrix that used for filtering. However, in prewitt kernel, the rows or colons are constant unlike sobel. The function that I wrote, has same mechanism with sobel function. It takes two more input just like sobel. When you analise the code attached below, it is the almost same as sobel and the only difference is the filter initialization. The threshold assumed 150. Please also check the result which placed below.



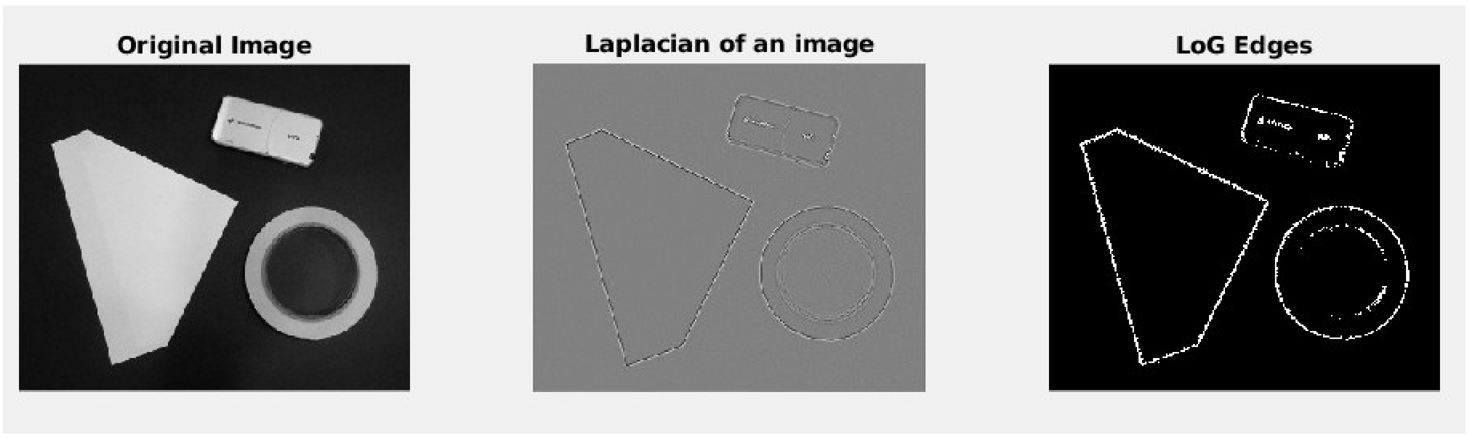


```
function [ret] = lab3prewitt(img, XorY,T)

if(length(size(img))==3)
    img = rgb2gray(img);
end
img=double(img);

%img = imgaussfilt(img);

if(XorY=='X')
    filter= [-1 0 1; -1 0 1; -1 0 1];
    ret=conv2(img,filter,"same");
end
if(XorY=='Y')
    filter= [-1 -1 -1; 0 0 0; 1 1 1];
    ret=conv2(img,filter,"same");
end
if(XorY == "XY")
   filterX= [-1 0 1; -1 0 1; -1 0 1];
   retX=conv2(img,filterX,"same");

   filterY= [-1 -1 -1; 0 0 0; 1 1 1];
   retY=conv2(img,filterY,"same");

   ret=sqrt(retX.^2 + retY.^2);
end

if(XorY=="Edge")
    [r,c]=size(img);
    ret=lab3prewitt(img,"XY",T);
    for i=1:r
        for j=1:c
            if(ret(i,j)<T)
               ret(i,j)=0 ;
            else
                ret(i,j)=255;
            end
        end
    end
end
ret=uint8(ret);
end
```

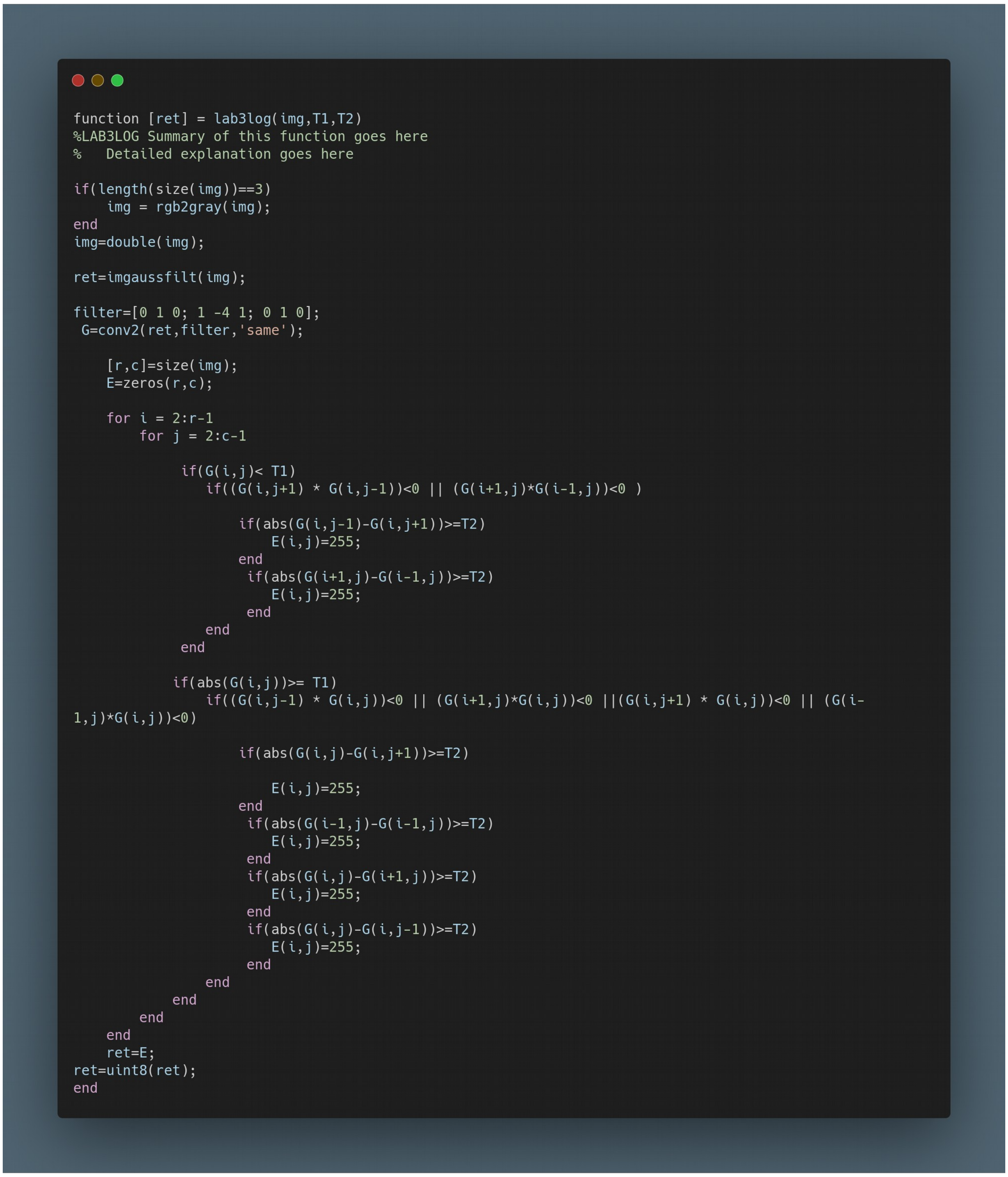
EE417 -Computer Vision Lab 3 Report
Hakan Buğra Erentuğ 23637
22 / 10 / 19

Assignment #3

The Laplacian of Gaussian is a function which looks like a pipe line. In discrete grid, for example, in 3x3 kernel it is [0 1 0 , 1 -4 1 , 0 1 0] which already defined. However, the instructors were very clear therefore the design of function is not kinda my own work. It is already given from TA. The instructors' purpose is detecting very rapid changes in the image. However, in discrete, kernel, the point that very close the zero can be in the center of kernel. Therefore we have to use a threshold which is very close the zero point to consider the continuous differential in a discrete atmosphere. If the center is small enough we have to check the neigbors of center in pairwise, whereas if it is very far from zero you have to check the each neighbors. Also, we need another threshold to check the slope of the difference. The convolution of filter and image that called laplacian of an image. The binary image which is removed according to thresholds which called LoG Edges. The threshold one is assigned to 10 and the threshold two is assigned to 90. Please check the test results and the function below.

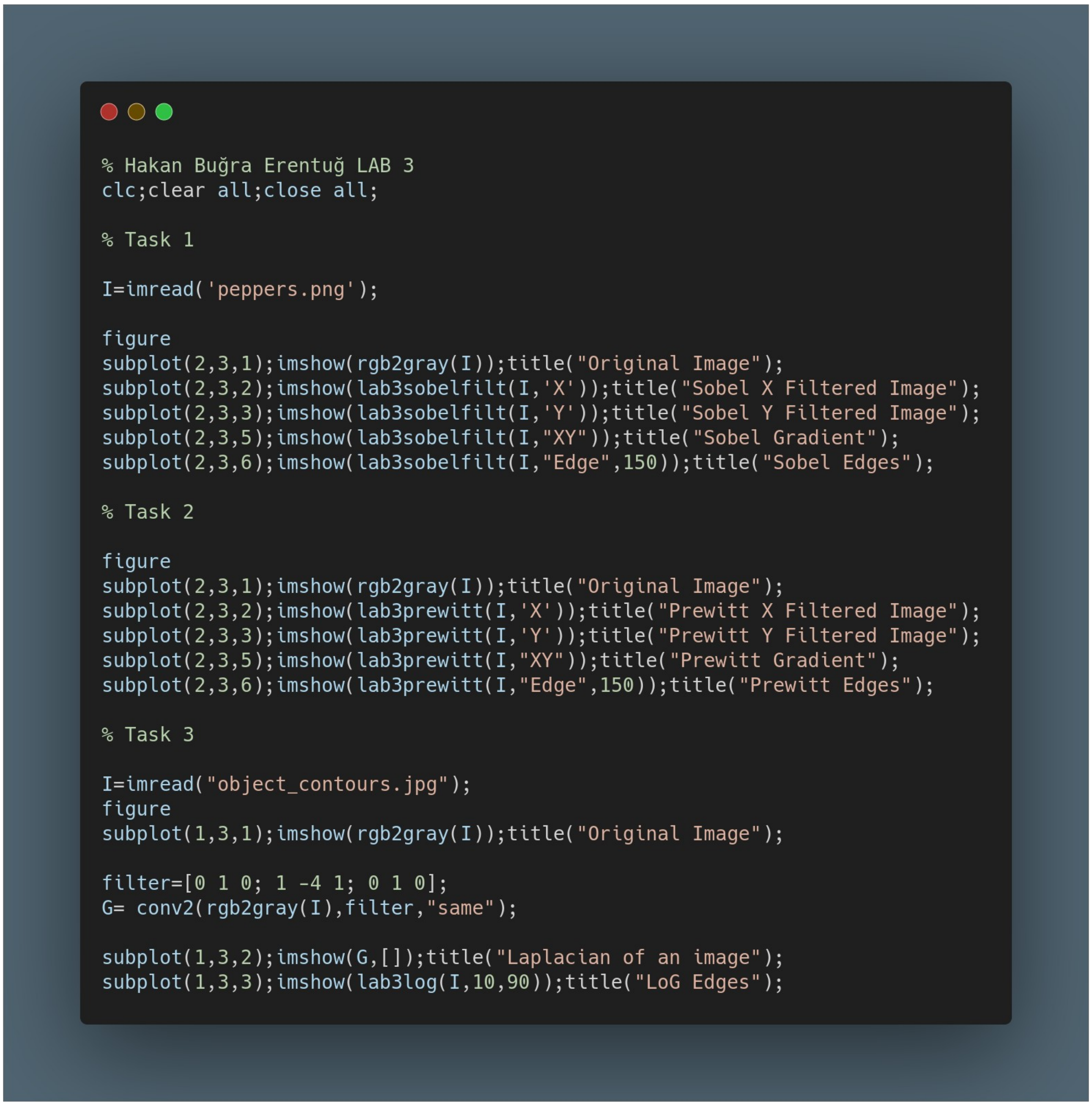


EE417 -Computer Vision Lab 3 Report
Hakan Buğra Erentuğ 23637
22 / 10 / 19



```
function [ret] = lab3log(img,T1,T2)
%LAB3LOG Summary of this function goes here
%   Detailed explanation goes here

if(length(size(img))==3)
    img = rgb2gray(img);
end
img=double(img);

ret=imgaussfilt(img);

filter=[0 1 0; 1 -4 1; 0 1 0];
 G=conv2(ret,filter,'same');

    [r,c]=size(img);
    E=zeros(r,c);

    for i = 2:r-1
        for j = 2:c-1

             if(G(i,j)< T1)
                if((G(i,j+1) * G(i,j-1))<0 || (G(i+1,j)*G(i-1,j))<0 )

                    if(abs(G(i,j-1)-G(i,j+1))>=T2)
                        E(i,j)=255;
                    end
                     if(abs(G(i+1,j)-G(i-1,j))>=T2)
                        E(i,j)=255;
                     end
                end
             end

            if(abs(G(i,j))>= T1)
                if((G(i,j-1) * G(i,j))<0 || (G(i+1,j)*G(i,j))<0 ||(G(i,j+1) * G(i,j))<0 || (G(i-
1,j)*G(i,j))<0)

                    if(abs(G(i,j)-G(i,j+1))>=T2)

                        E(i,j)=255;
                    end
                     if(abs(G(i-1,j)-G(i-1,j))>=T2)
                        E(i,j)=255;
                     end
                     if(abs(G(i,j)-G(i+1,j))>=T2)
                        E(i,j)=255;
                     end
                     if(abs(G(i,j)-G(i,j-1))>=T2)
                        E(i,j)=255;
                     end
                end
            end
        end
    end
    ret=E;
ret=uint8(ret);
end
```

EE417 -Computer Vision Lab 3 Report
Hakan Buğra Erentuğ 23637
22 / 10 / 19



```matlab
% Hakan Buğra Erentuğ LAB 3
clc;clear all;close all;

% Task 1

I=imread('peppers.png');

figure
subplot(2,3,1);imshow(rgb2gray(I));title("Original Image");
subplot(2,3,2);imshow(lab3sobelfilt(I,'X'));title("Sobel X Filtered Image");
subplot(2,3,3);imshow(lab3sobelfilt(I,'Y'));title("Sobel Y Filtered Image");
subplot(2,3,5);imshow(lab3sobelfilt(I,"XY"));title("Sobel Gradient");
subplot(2,3,6);imshow(lab3sobelfilt(I,"Edge",150));title("Sobel Edges");

% Task 2

figure
subplot(2,3,1);imshow(rgb2gray(I));title("Original Image");
subplot(2,3,2);imshow(lab3prewitt(I,'X'));title("Prewitt X Filtered Image");
subplot(2,3,3);imshow(lab3prewitt(I,'Y'));title("Prewitt Y Filtered Image");
subplot(2,3,5);imshow(lab3prewitt(I,"XY"));title("Prewitt Gradient");
subplot(2,3,6);imshow(lab3prewitt(I,"Edge",150));title("Prewitt Edges");

% Task 3

I=imread("object_contours.jpg");
figure
subplot(1,3,1);imshow(rgb2gray(I));title("Original Image");

filter=[0 1 0; 1 -4 1; 0 1 0];
G= conv2(rgb2gray(I),filter,"same");

subplot(1,3,2);imshow(G,[]);title("Laplacian of an image");
subplot(1,3,3);imshow(lab3log(I,10,90));title("LoG Edges");
```