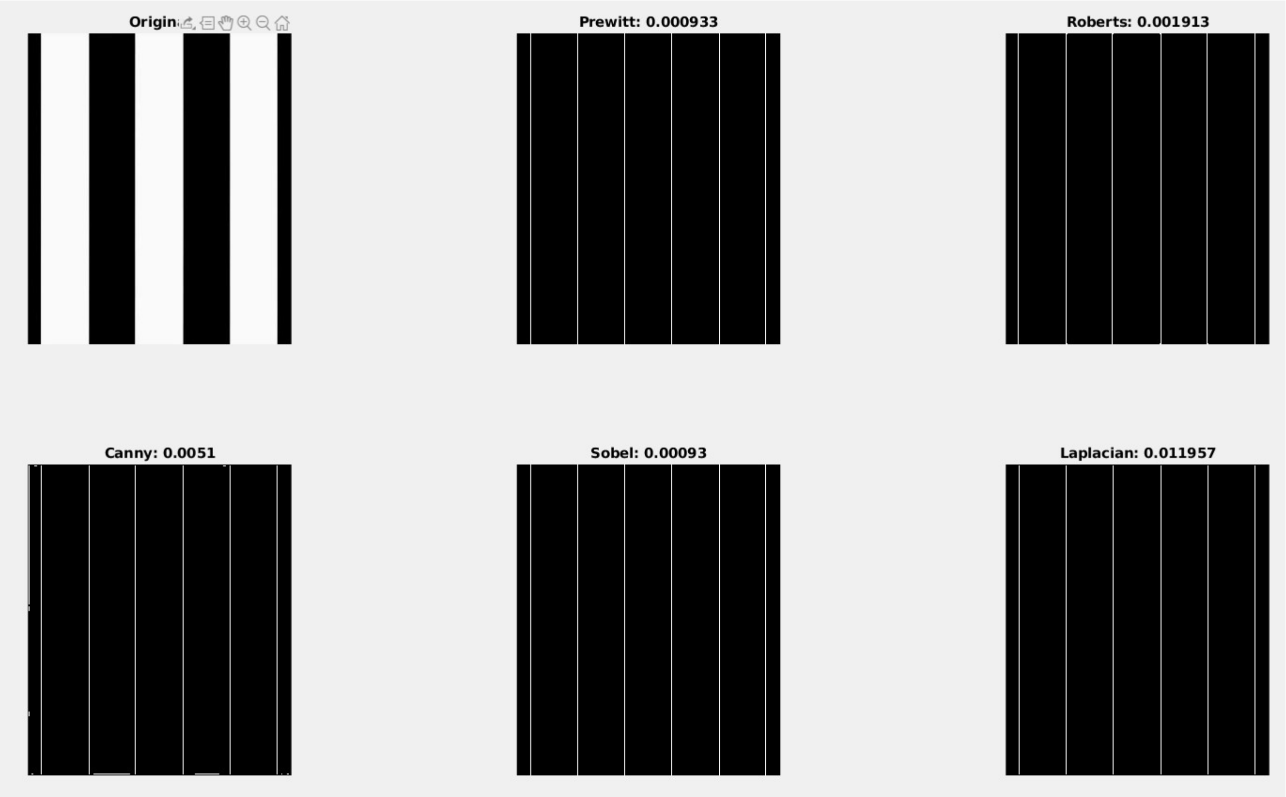
EE417 – Computer Vision Homework #1
Hakan Buğra Erentuğ – Sabancı University
25 / 10 / 19

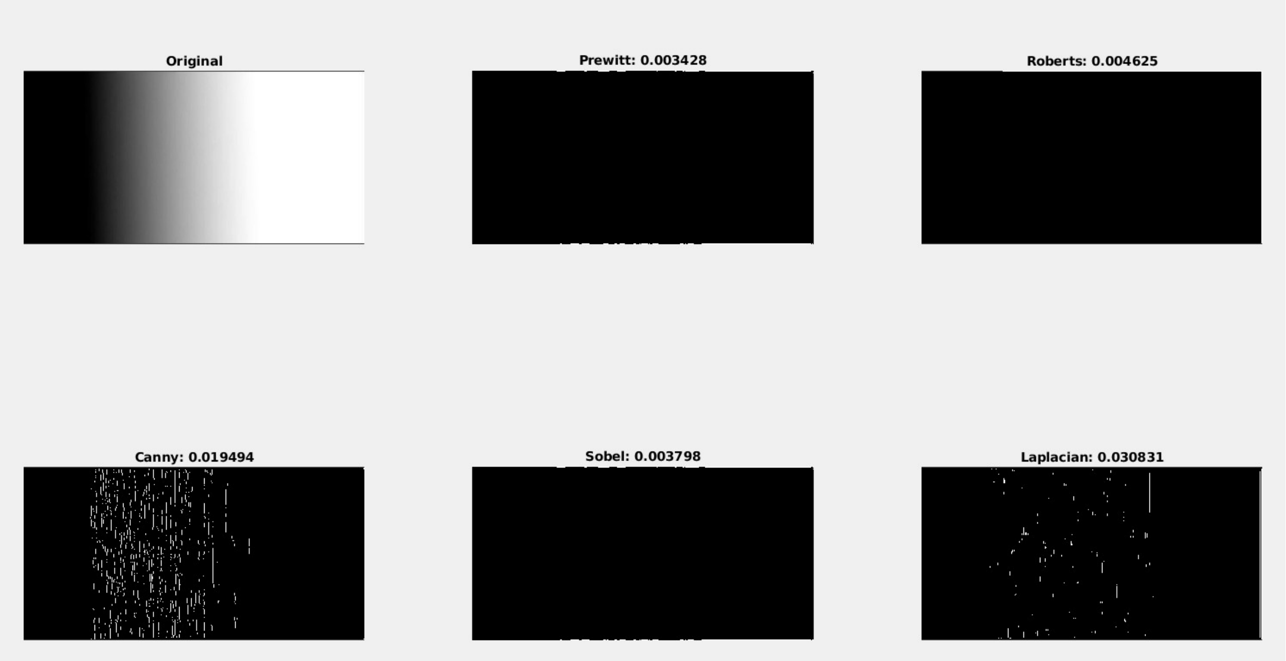
In this assignment, the comparison of various edge detection analyzed with the figures and performance results. In the figures, there is a elapsed time (in ms), next to title which is related to each edge detectors' performance. During the assignment, different type of images like 1D, portrait, landscape, highly-detailed etc, has been chosen as a test images. I assure that any statement which do not have reference, is my own statement. You can find the executer main MATLAB code, all images and screenshots of results in the zip file. You are free to test, in your computer and verify solutions. The test environment is Lenova-Y520- 15IKBN, 15.% GiB memory, Intel® Core™ i7-7700HQ CPU @ 2.80GHz × 8, Intel® HD Graphics 630 (Kaby Lake GT2) ,Ubuntu 18.04.3 LTS ,251 GB free disk space and '9.7.0.1190202 (R2019b)' MATLAB version. You can find all related work in my GitHub page https://github.com/herentug/EE417.

Test 1



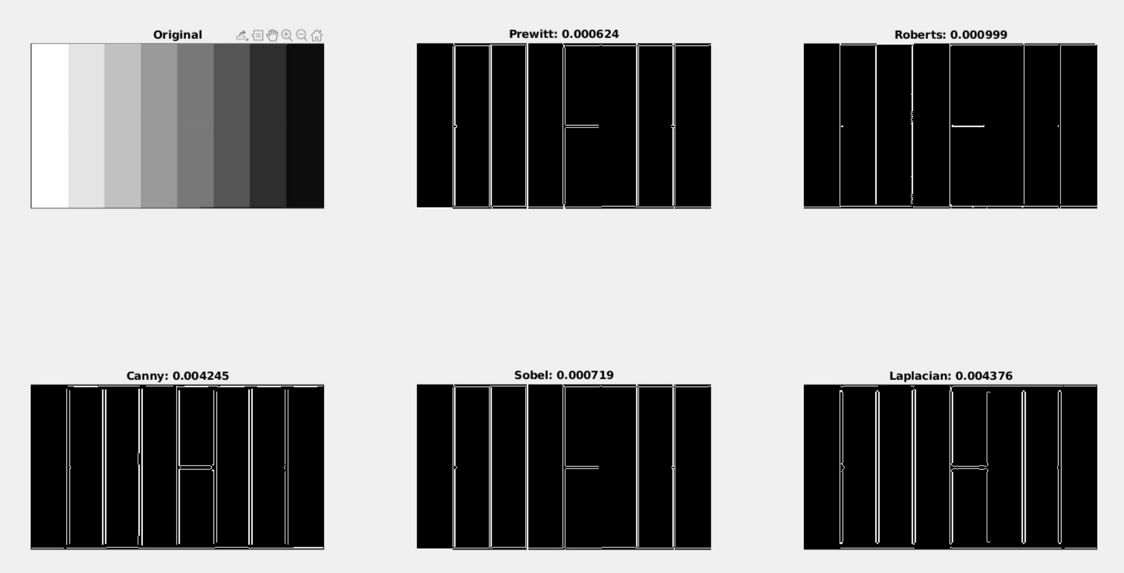
Above, the one of the basic 1 D image has been tested. Obiviously, all of the edge detectors could detect the edges correctly. However, the best performance obtained by Sobel and Prewitt operetors due to their simple filter and algoritm design.

Test 2



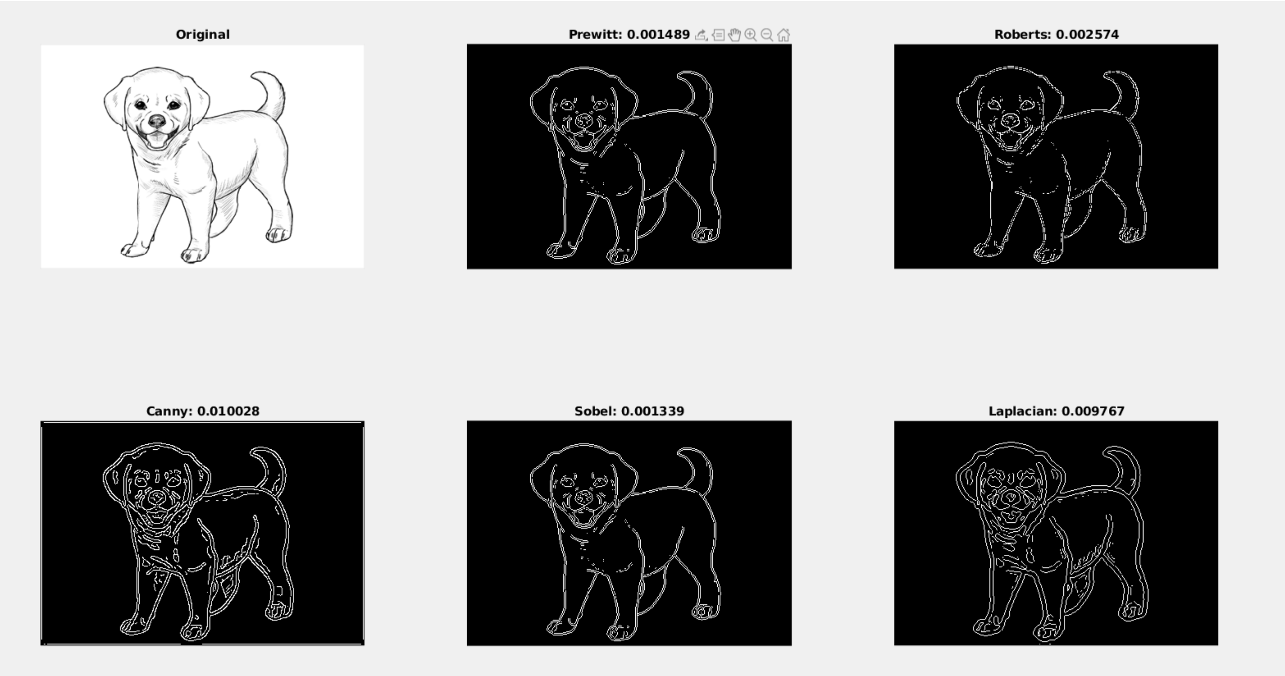
In the test 2, another 1D image has been selected. The image has been made by author with Gimp Gradient Tool. Unfortunately, due to low pixel value difference between neighbors, most of the edge detectors cannot detect edges. In fact, it is hard to detect edges even for humans. Nevertheless, we can say, the canny operator is more sensible to gradient images. It is preferable depends on what you want to detect.

Test 3



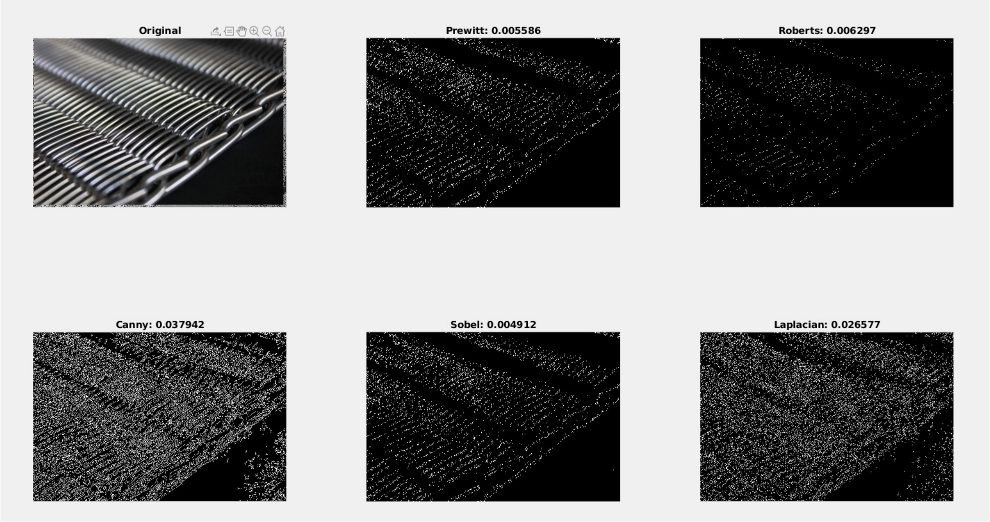
In test 3, another gradually change has occurs. However, in this example it is 1D image which has step graph. It is more easy to detect edges for operators due to rapid change in pixel value. However, Canny, Laplacian and Prewitt operands detect two parallel lines because of the low quality edges of image. The best performance results are still owned by Sobel and Prewitt. However zero crossing operations such as Laplacian could detect the one more line in the image.

Test 4:



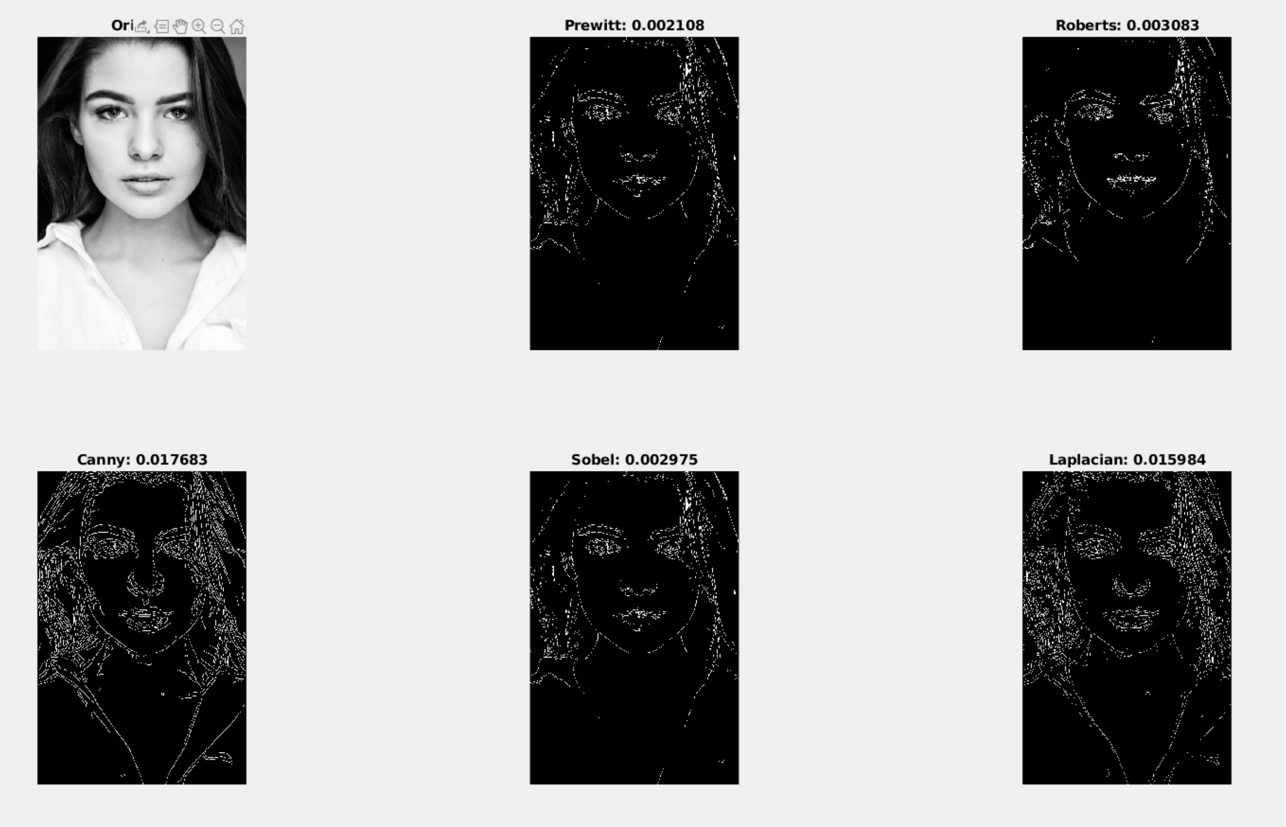
In test 4, a cartoon character which made of black and gray lines, has been selected. Roberts, Prewitt and Sobel have eliminated the gray strikes in puppy and extra parallel line for the drawing. The best satisfactory results has come up by Roberts. We can conclude the Laplacian and Canny is better to detecting edge in drawings due to lack of too many responses and poor localization.

Test 5:



In test 5, the hard HD quality image has been used. The image has many consecutive rapid changes in pixel values. Still, Canny and Laplacians takes more time to calculate due to expensive operations in thie algoritm as expected. In addition, Canny and Laplacian is a bad choice because they give too many responses which is a bad criteria for optimized edge detection. Also, Robert make poor localization acroos the edges which makes the best choices, Sobel and Prewitt.

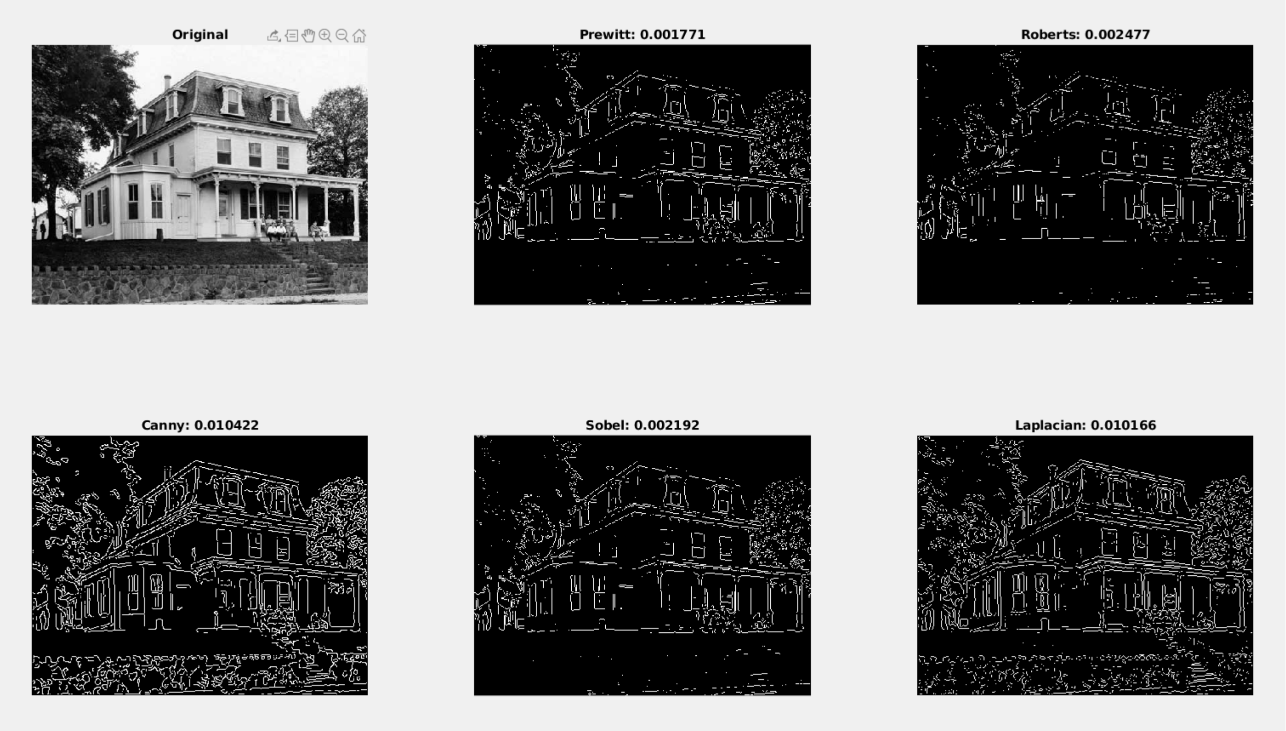
Test 6:



In test 6, we tested a portrait image of a woman to test best choice for faces’ edge detection. The best choice is depends on how much detail you wanted. The less detailed image is formed by Roberts algorithm whereas the high detail is formed by Canny. No need to say about elapsed times of algorithms.

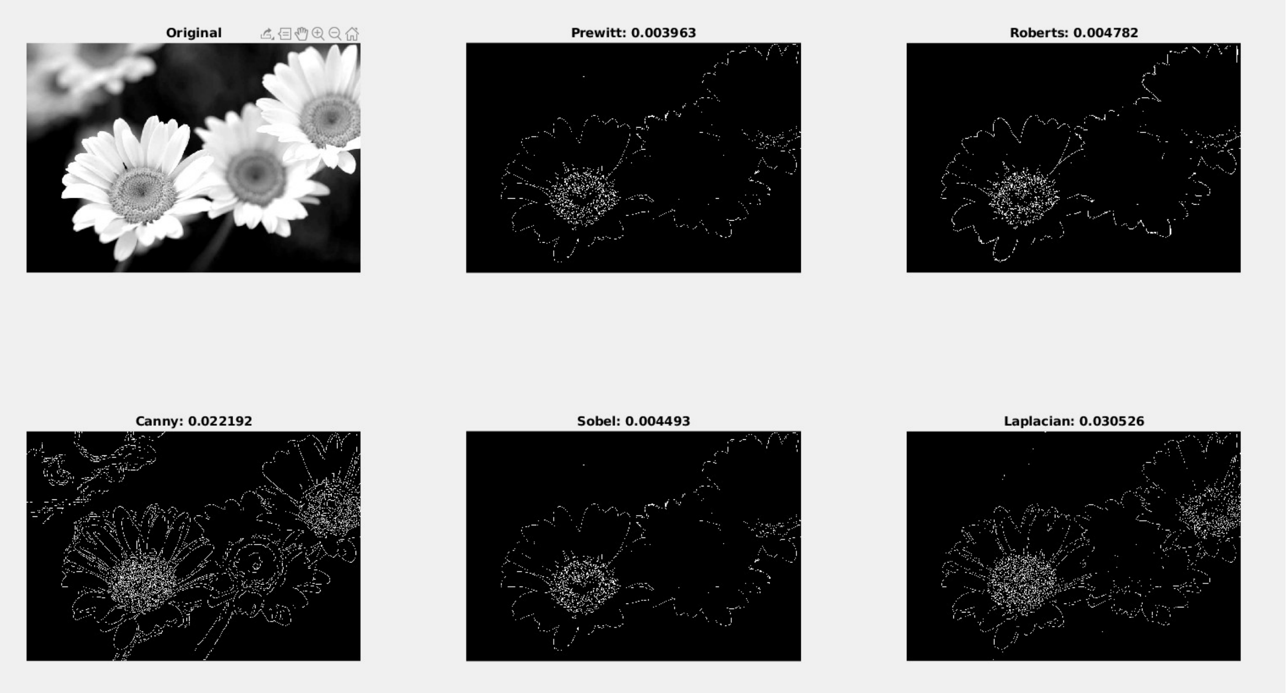
EE417 – Computer Vision Homework #1
Hakan Buğra Erentuğ – Sabancı University
25 / 10 / 19

Test 7:



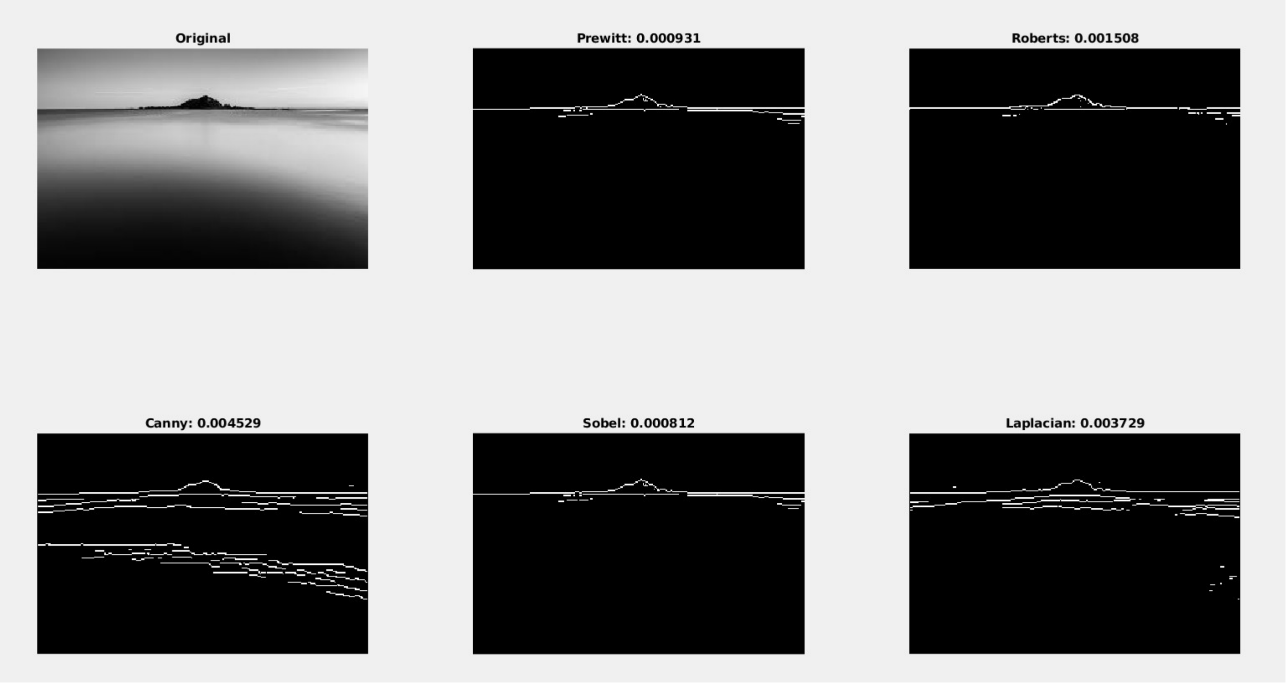
In test 7, we have tested the edges of a house. Apparently, no matter the image, Cany and Laplcian gives high amouth of detail with many responses. However, In prewitt Roberts and Sobel, some of the important edges did not detected.

Test 8:



In test 8, another default simplistic photograh has been selected. In Laplacian and Canny, the petals can clearly distinguish. Also, It is beneficial to say that with sobel, prewitt, robert the important informatios about background become hidden.

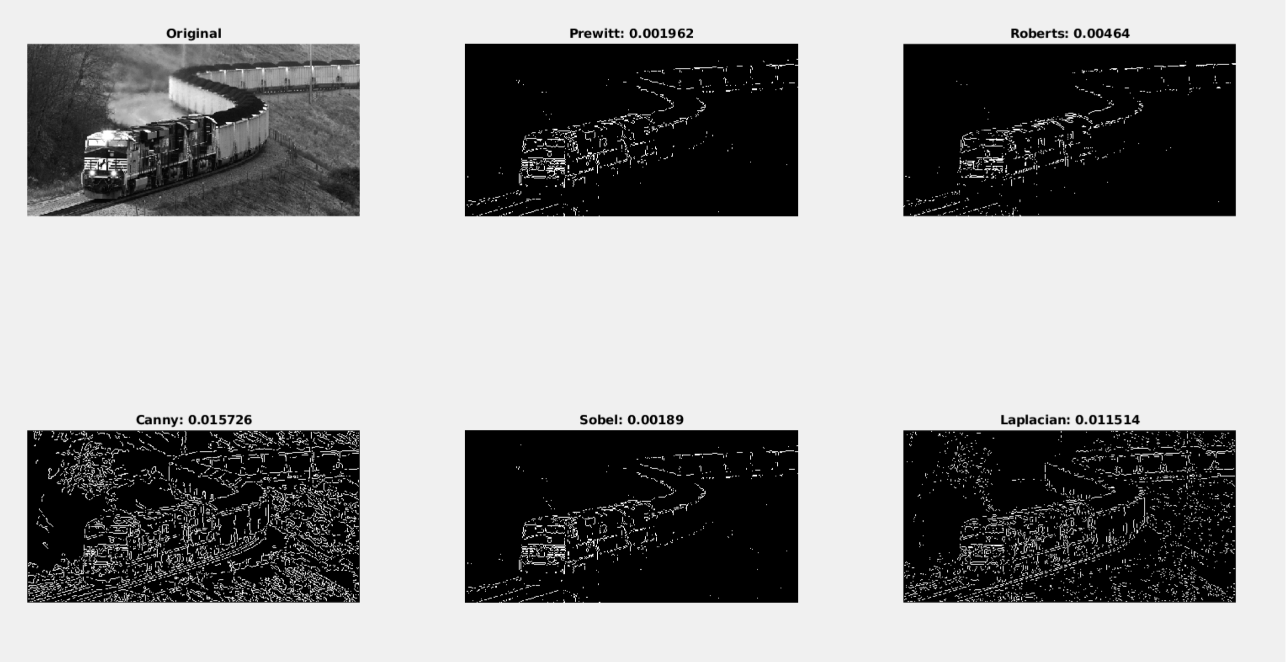
Test 9:



In test 9, a landscape image has been selected. However, it id hard to detect edges even with human eyes. While Prewitt, Roberts, Laplacian and Sobel was only able to distinguish horizon line and mountain, Canny was able to colour change in water, which made Canny the most detail-giving edge detector.

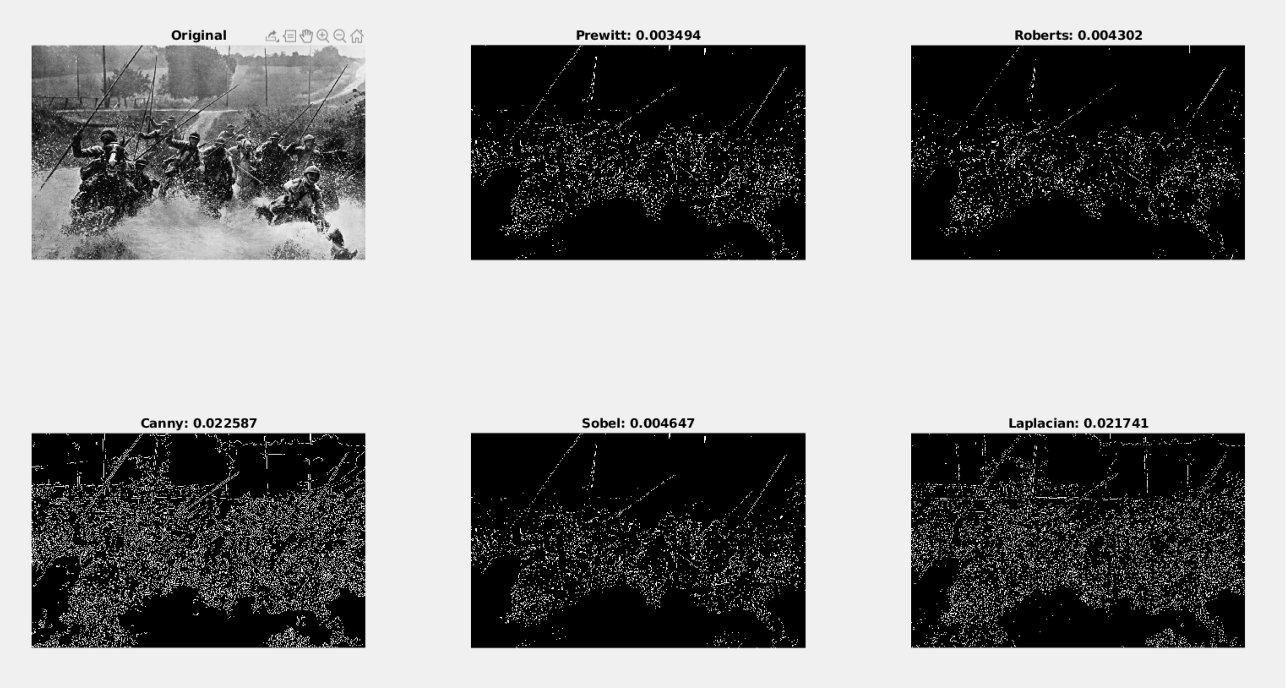
Test 10:

In the test 10, a view who focus on centralized object has been selected. Possible use case is distinquish some objects along the highly-detailed backgrounds. As predicted, the simplistic edge detectors such as Sobel Prewitt and Robert focus on object while eliminate the unnecessary information. Canny and Laplacian still consist many unnecassay information about train.



Test 11:

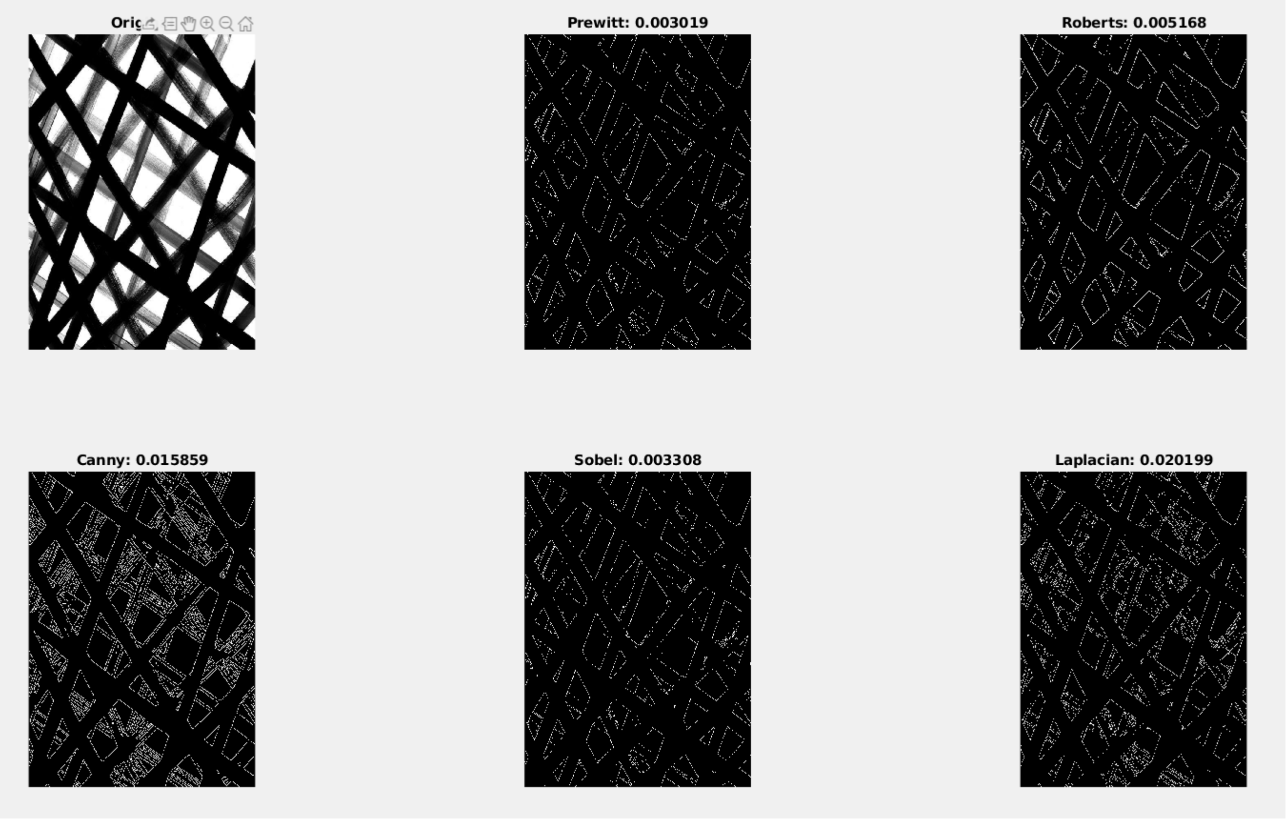
In test 11, the higly confusing war scene has been selected, to analysis edge detector’s behaviour agaisnt crowded images. As expected, edge detectors which gives many responses is a bad choice in this use case. Simplistic algoritms like prewitt and Roberts are more preferable.



EE417 – Computer Vision Homework #1
Hakan Buğra Erentuğ – Sabancı University
25 / 10 / 19

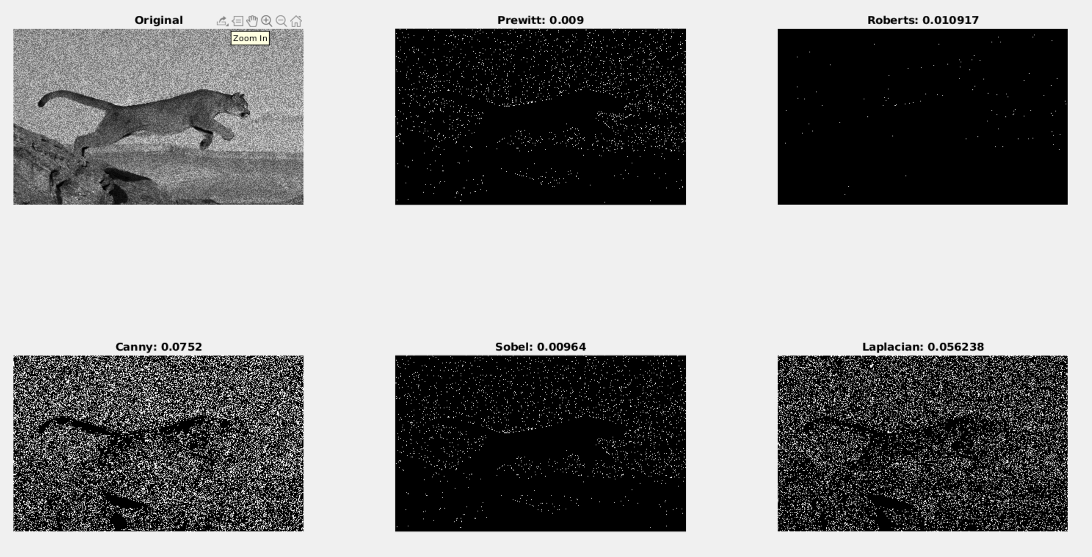
Test 12:

In test 12, I want to point the difference Prewitt and Roberts algoritms which have many similarties. To achieve that I selected a blurry image. The difference is Prewitt create more thin edges unlike Roberts. Which means, prewitt is more capable to give single response which is beneficial property in edge detectors.



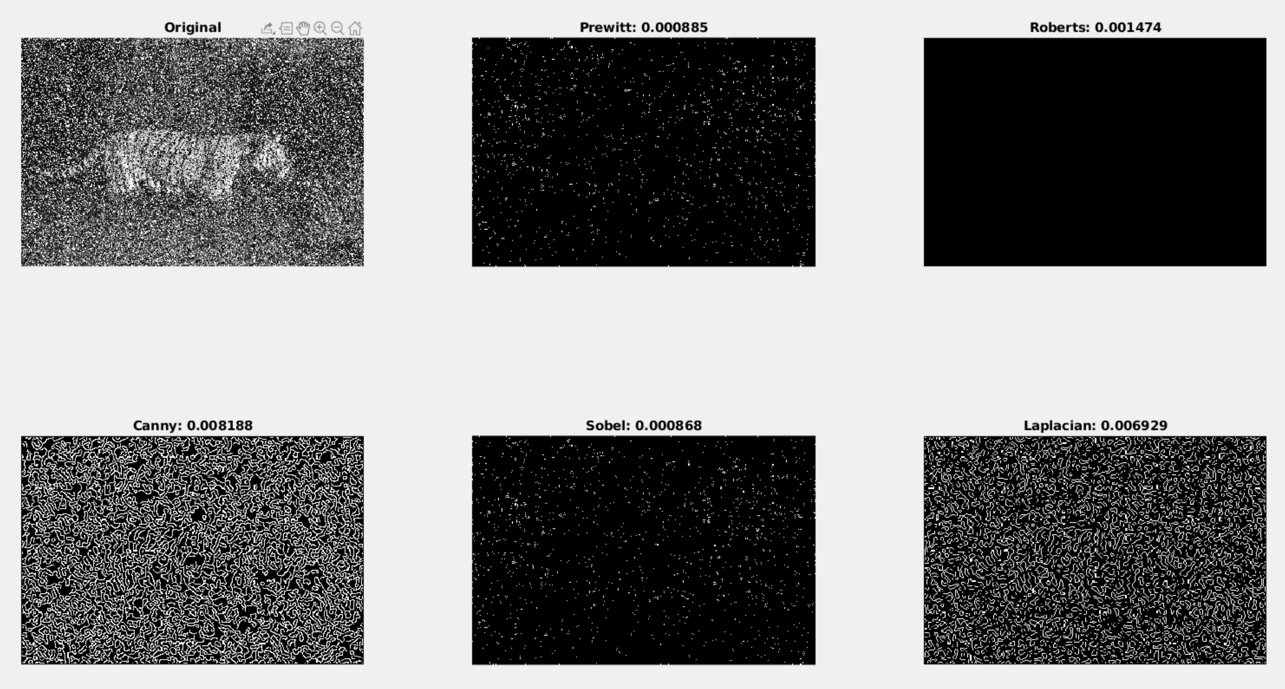
Test 13:

In test 13, I wanted to examine the behaviours of edge detectors against noisy image. Apparently, Roberts Laplacian and Canny are bad algoritm to use in blurry image. The most object-detectable built in functions are prewitt and sobel which are almost same.



Test 14:

In test 14, I choseed an image with hight amounth of slat&pepper noise to examine behaviours of Edge detectors more against noise. However, none of the edge detectors, cannot detect the tigers edge. Without object knowledge, these algoritms output is indetectable to tiger. In fact, Robert even cannot give any single response to that highly noised image. While others seems like an black image with random points. That is why, smooting or filtering is crucial before detecting edges for noisy images.

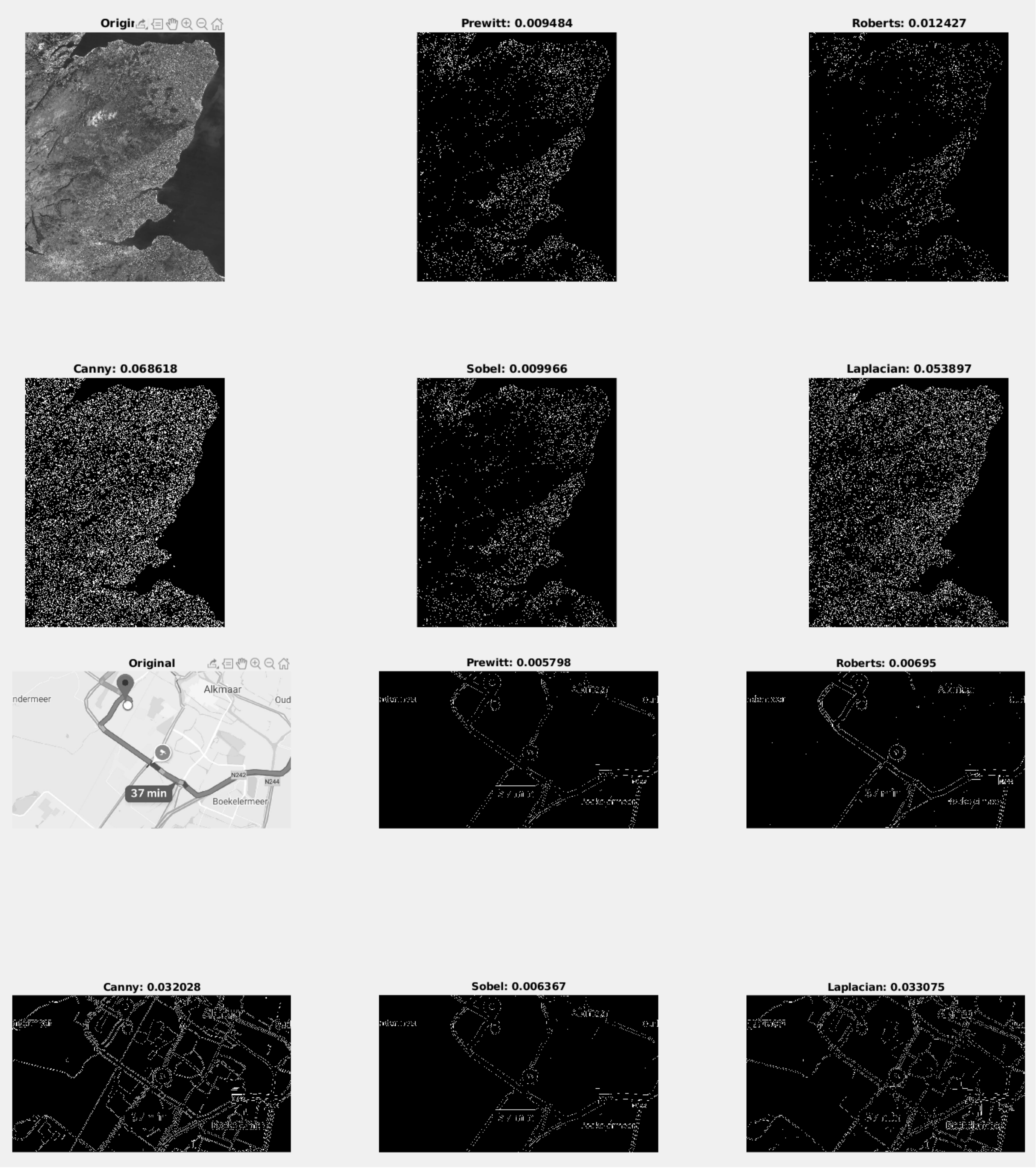


Special Cases:

a) Map Projection

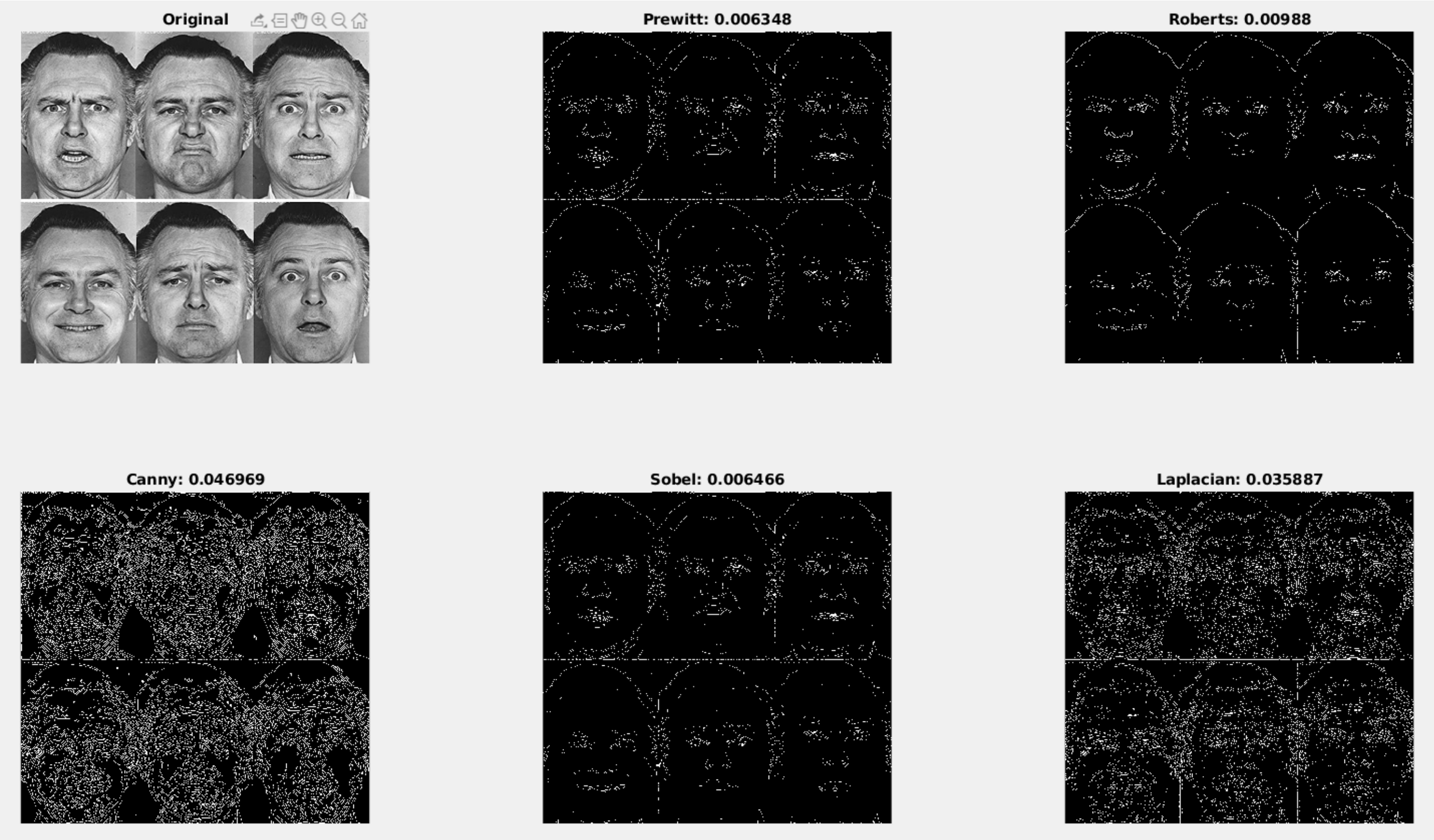
Below, there are two different image which taken from Google Maps and Google Earth for map projecting technology. For the digital simulation of maps, canny and robert are the best choice due to high performance localization and robustness. However, in satelite views, it is better to have less detail to distinguish geographical structures. Therefore, simple algoritms are better. Please check results in below.

EE417 – Computer Vision Homework #1
Hakan Buğra Erentuğ – Sabancı University
25 / 10 / 19



EE417 – Computer Vision Homework #1
Hakan Buğra Erentuğ – Sabancı University
25 / 10 / 19

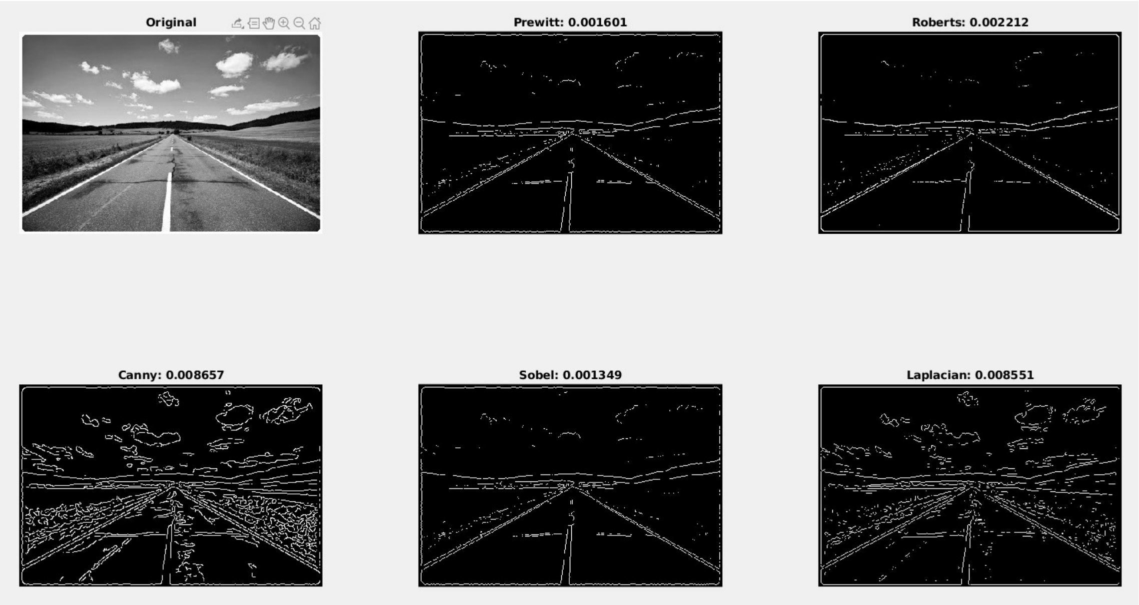
b) Emotion Detection



Without any object information, it is better to use simplistic algorithms for the emotional detection due to detailed and complex structure of faces. Still, Edge detection is not a advisable in comparison to object detection for emotion analysis. I took this topic into consider because of my heavy interest.

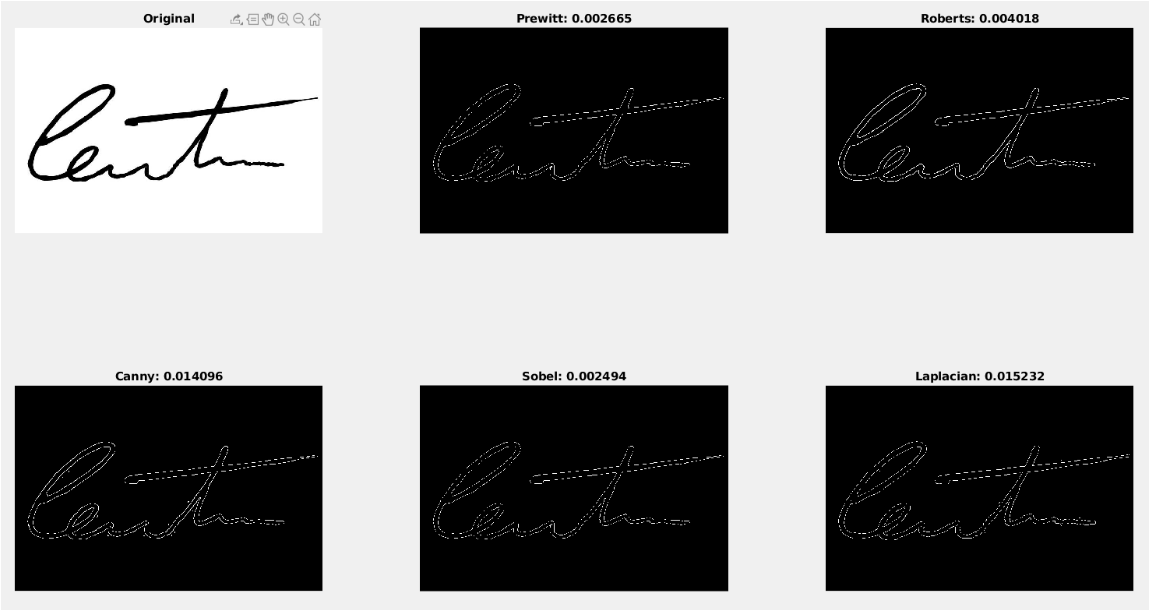
c) Self Driving Cars

Edge detection can be use case for detecting carriageway. Nevertheless, object based neural networks, line detection and object annotation reinforced learning are better technologies for self driving cars. Still, the simplistic algorithms may be better for road line detection due to the fact that they eliminate unnecessary information such as cloud, soil types etc.



d) Hand-Writing and Signature Verification

Edge detection is terrible solution to hand-writing image on its own because on handwritten papers there are many noice which makes the writing undetectable. The best used algortim is neural networks. However, for digital based signatures, almost all of the edge detectors could be utilize, due to appear rapid changes on one colored background. In this case, the performance should be the criteria that we are looking for. It makes Sobel and Prewitt the best candidates.

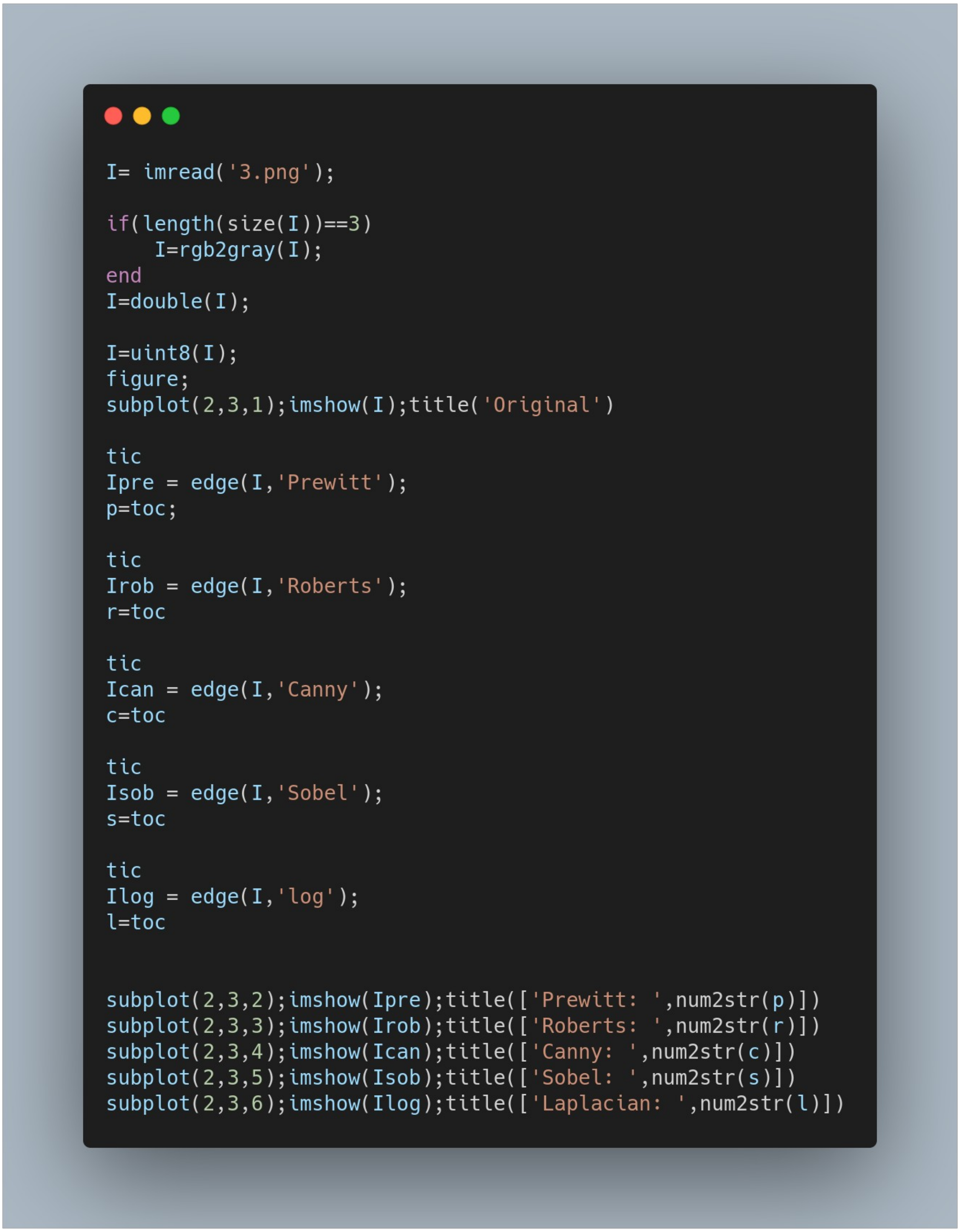


EE417 – Computer Vision Homework #1
Hakan Buğra Erentuğ – Sabancı University
25 / 10 / 19

Conclusion

While the classical operators like Sobel, Prewitt or Robert are one of the simplistic edge detection which search for small area. Also, their sensitivity to noise and inaccurate localization could be disadvantages. However, these are successful to detect of the edges and their orientations. In addition, thanks to their cheap algorithms, elapsed time is very low and wonderful choice which requires large image analysis and low detailed image. Secondly, LoG Appalachian of Gaussian can find the correct place of edges with ability to testing wider area around pixel. However, may malfunction at the corners. It also have probability that not finding orientation because of laplacian filter. And lastly, Canny, have huge advantage about localization and response. It is the one often most sensitive against noise and may detect the edges better on noisy image. However, it require complex computation and more time for the algorithm. Shortly, it is better the know the properties of algorithm and find the most suitable algorithm for edge detection. Unfortunately, no algorithm is best for all images, it depends on image property and use-case.

EE417 – Computer Vision Homework #1
Hakan Buğra Erentuğ – Sabancı University
25 / 10 / 19



```
I= imread('3.png');

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

I=uint8(I);
figure;
subplot(2,3,1);imshow(I);title('Original')

tic
Ipre = edge(I,'Prewitt');
p=toc;

tic
Irob = edge(I,'Roberts');
r=toc

tic
Ican = edge(I,'Canny');
c=toc

tic
Isob = edge(I,'Sobel');
s=toc

tic
Ilog = edge(I,'log');
l=toc


subplot(2,3,2);imshow(Ipre);title(['Prewitt: ',num2str(p)])
subplot(2,3,3);imshow(Irob);title(['Roberts: ',num2str(r)])
subplot(2,3,4);imshow(Ican);title(['Canny: ',num2str(c)])
subplot(2,3,5);imshow(Isob);title(['Sobel: ',num2str(s)])
subplot(2,3,6);imshow(Ilog);title(['Laplacian: ',num2str(l)])
```