CV - Assignment 1

Eitan Kosman - 312146145
eitan.k@campus.technion.ac.il

December 22, 2019

1 Question 1

The plots are in 1. The output of the program is:
LS : intercept = [5.21289072] slope = [0.82921182]
TLS : intercept = 4.686578364579161 slope = 0.9406227881691073
It's noticeable that the slope for the TLS is bigger while the intercept is lower. However, from the given set of coordinates one can easily notice that it consists of two lines that are best described by:

$$\begin{aligned} y &= x + 5 \\ y &= x + 4 \end{aligned} \tag{1}$$

That means that the best line describing all points is the line that is parallel to both with the same slope, i.e.:

$$y = x + 4.5 \tag{2}$$

That line has intercept of 4.5 and slope of 1. These measurements brings me to the conclusion that the TLS fits the points better and this makes sense because the TLS is known to be less sensitive when both x and y contain noise.

2 Question 2

The resulting final image is presented in 2. ***Note*** **:** the positive-x is the right direction and the positive-y is the down direction.

2.1 Explaining the imshow's

2.1.1 First call

The first call to *imshow* relates to a zero image with one white point (the origin). The coordinates in the w-y place are $(0, 0)$, thus following the definition in the feature's domain $\rho = x \cdot cos(\theta) + y \cdot sin(\theta)$ we get $\rho = 0$ which is a straight line that is parallel to the θ axis and crosses the ρ at 0.

2.1.2 Second call

The second call to *imshow* relates to a zero image with two white points: $(0, 0), (100, 0)$. This adds a curve that describes the second point in the feature's domain. Again, using the definition of the feature's domain, we plug x and y and get:

$$\begin{aligned} \rho &= x \cdot cos(\theta) + y \cdot sin(\theta) \\ \rho &= 100 \cdot cos(\theta) + 0 \cdot sin(\theta) \\ \rho &= 100 \cdot cos(\theta) \end{aligned} \tag{3}$$

This equation can perfectly describe the second curve I got. This curve has a cosine shape and it's amplitude is 100.

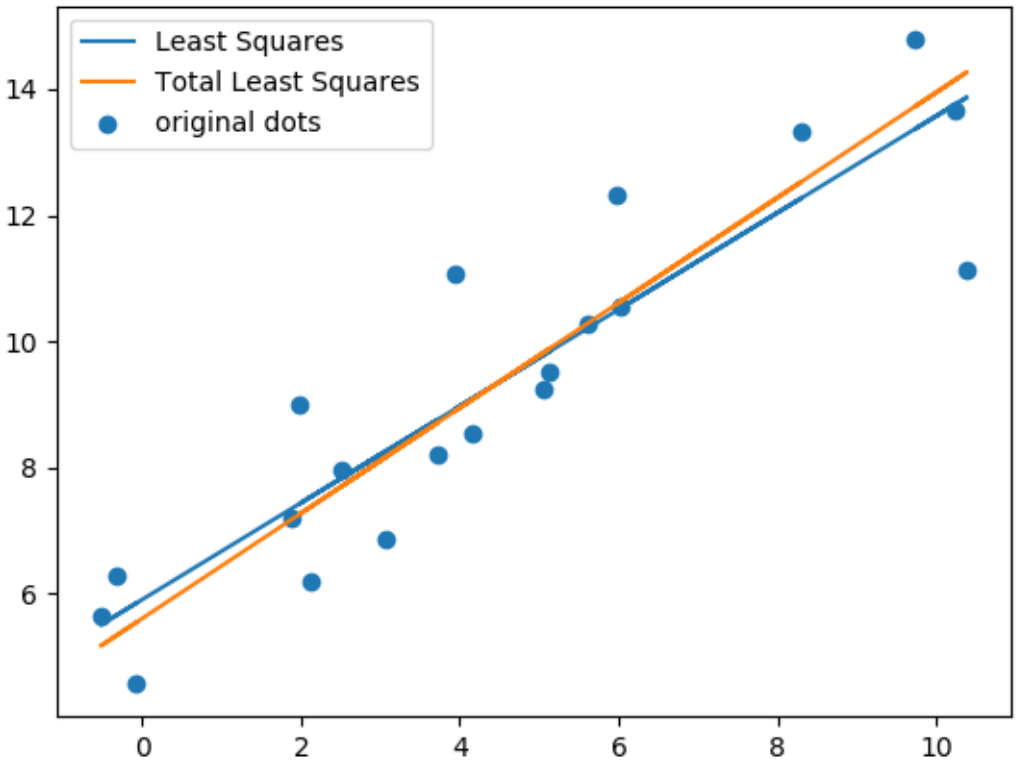

Figure 1: Diagrams for question 1

2.1.3 Third call

The third call to *imshow* relates to a zero image with three white points: $(0,0),(100,0),(0,100)$. This adds a curve that describes the third point in the feature's domain. Similarly to the second point, I plug the coordinates here:

$$\begin{aligned} \rho &= x \cdot cos(\theta) + y \cdot sin(\theta) \\ \rho &= 0 \cdot cos(\theta) + 100 \cdot sin(\theta) \\ \rho &= 100 \cdot sin(\theta) \end{aligned} \tag{4}$$

This time, I get a sinus and it's amplitude is 100 too

2.1.4 Fourth call

This call to *imshow* adds another curve that relates to the point $(100,100)$ Plugging this into the equation, I get:

$$\begin{aligned} \rho &= x \cdot cos(\theta) + y \cdot sin(\theta) \\ \rho &= 100 \cdot cos(\theta) + 100 \cdot sin(\theta) \end{aligned} \tag{5}$$

Using the identity $cos(x) + sin(x) = \sqrt{2} \cdot sin(\frac{\pi}{4} + x)$, I get $\rho = 100 \cdot \sqrt{2} sin(\frac{\pi}{4} + \theta)$, thus I end up with a sinus-like curve which is similar to the previous curve I just described, but it has a greater amplitude.

2.2 A More Complex shape

The rectangle I draw is presented in 3. This image resulted after running canny edge detector is presented in 4 and it's Hough transform is presented in 5. As shown is the last diagram, I observe that there are 4 bright spots. This is expect since the rectangle is formed by 4 straight lines that might be well described by the bright spots. One dominant feature is that there are two pairs of bright spots, each pair share the same $\theta value, which indicate two pairs of parallel lines.$

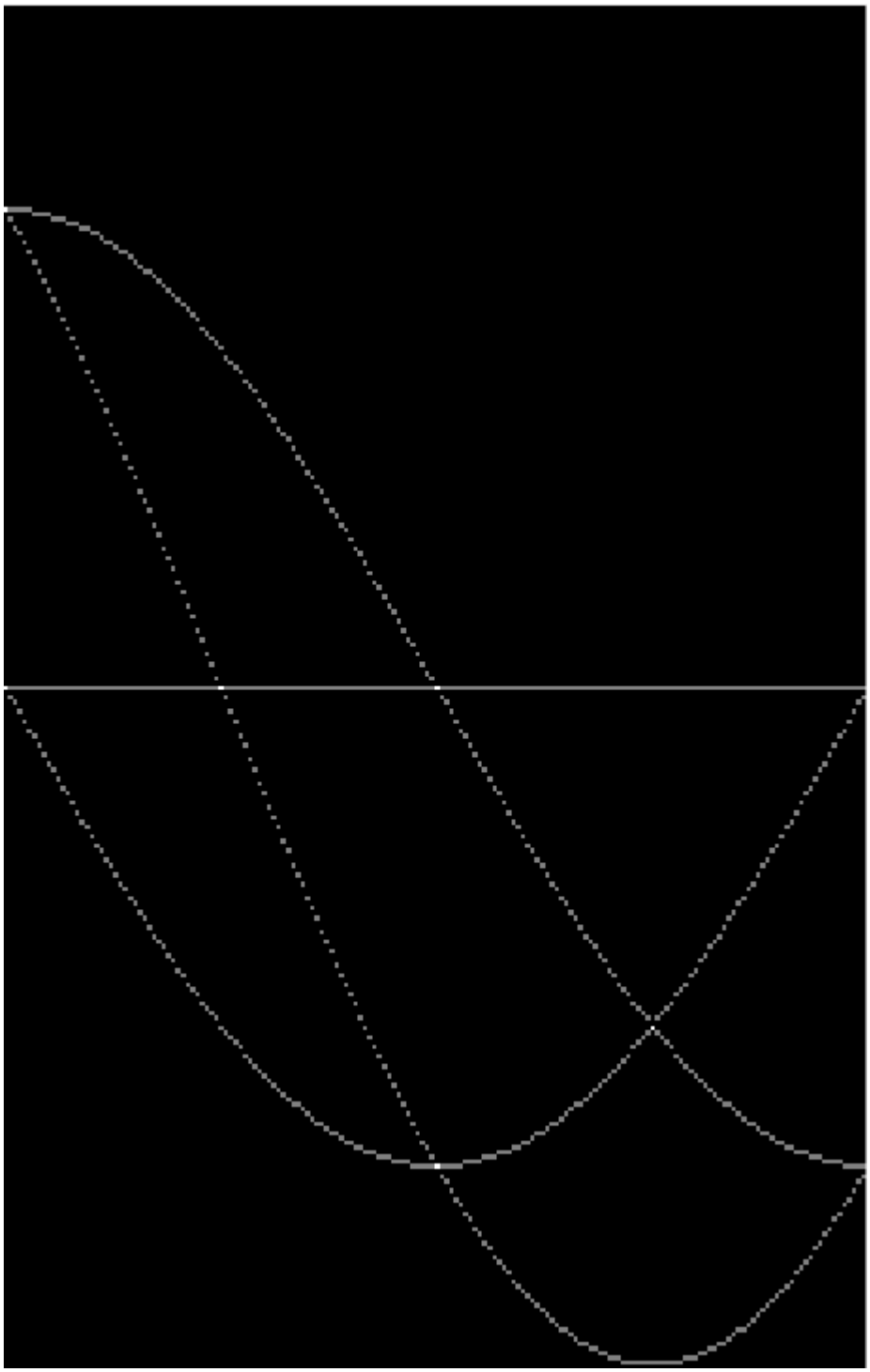

Figure 2: Diagrams for question 2

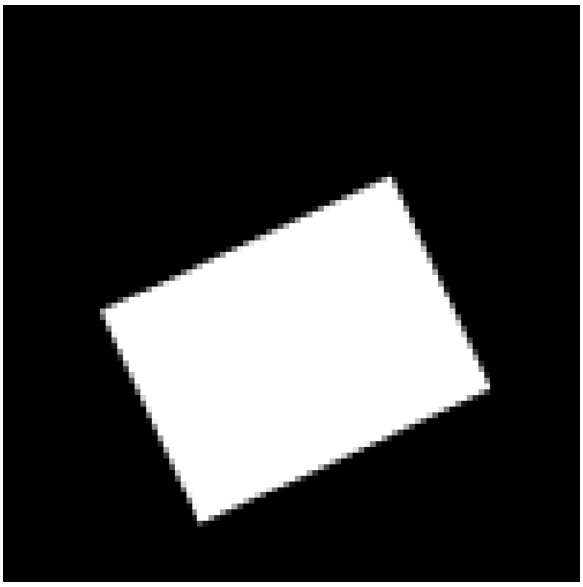

Figure 3: Diagram of the square

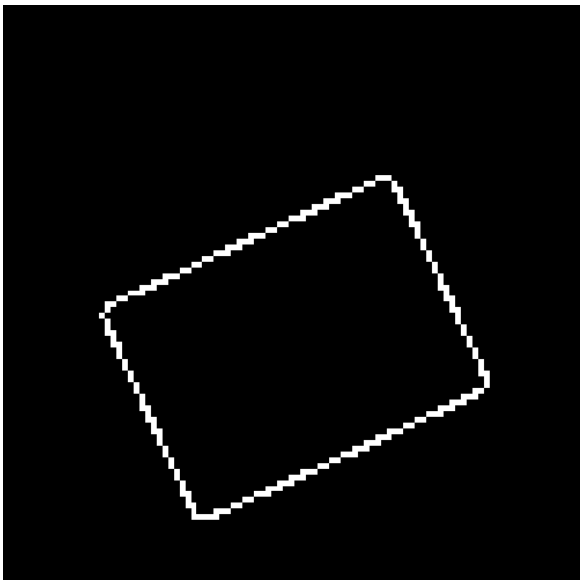

Figure 4: Diagram of the edges of the square, using canny edges detector

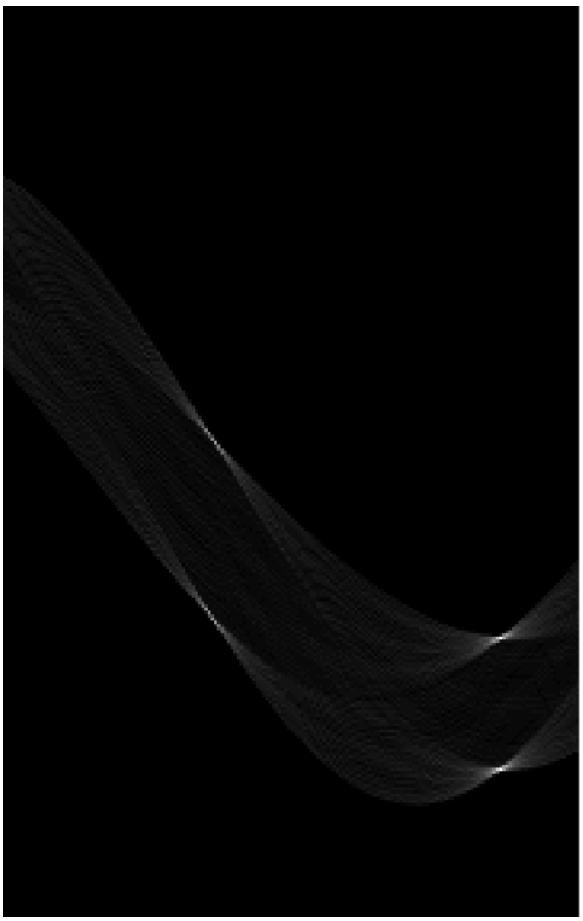

Figure 5: Diagram of the Hough transform of the edges of the square

Figure 6: Edges after Hough transform and voting

2.3 Explain why lines in the image space are represented as the maximum point of the Hough transform

Recall the formulation of the Hough transform:

$$\rho = x \cdot cos(\theta) + y \cdot sin(\theta)$$

For each line, we have a set of points that lie on that line. Each of these points would give another vote for the r and θ value corresponding with that line. All other candidate lines will have fewer votes.

For reconstructing the image, I find the maximal votes by using the function *houghpeaks* and get the points marks in 7. Later, I use the function *houghlines* to reproduce the lines and loop over the result for plotting them over the original image, presented in 8.

2.4 Reproduce the results for input image building.jpg

I set the thresholding value to 0.15 for the canny edge detector. I use the same code for the detection, however I choose to only plot the edges for more clarity. The result is represented in **??**

3 Question 3

3.1 Implement a function that decomposes a gray-level image to its Laplacian pyramid

Implemented this function laplacian_pyramid.m as:

$$function\ pyramids\ =\ laplacian\_pyramid(input, nLevel, mask) \tag{6}$$

3.2 Implement a function that reconstructs an image from its Laplacian pyramid

Implemented this function in reconstruct.m as:

$$function\ output\ =\ reconstruct(pyramid) \tag{7}$$

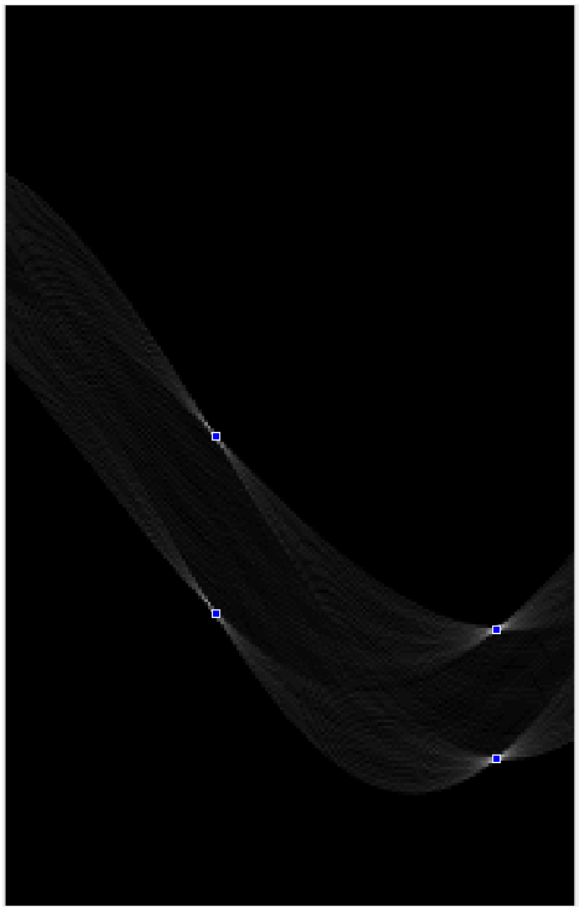

Figure 7: Peaks of the Hough transform of rectangle edges

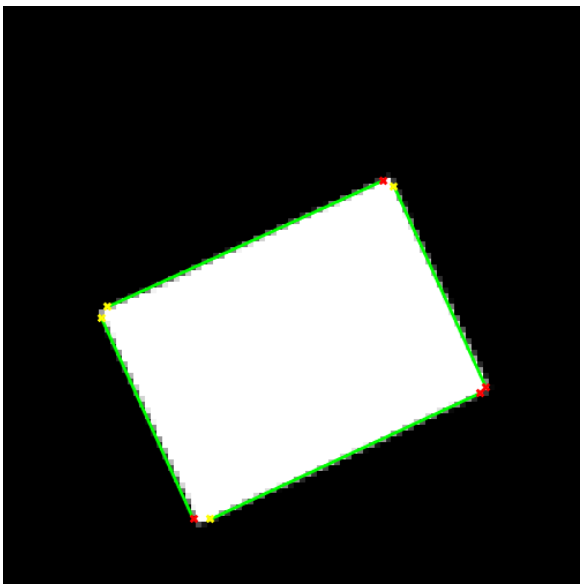

Figure 8: Detected edges of the rectangle in green

This function simply returns the sum of the pyramid's elements along the first axis. This is valid since we use no up/down scaling.

3.3 Implement a function that accepts an input image, its mask and the example background

Implemented this function in style_transfer.m as:

$$function\ output = change\_background(img, bg, mask) \tag{8}$$

This function simply multiplies the elements of the two images by zeros and ones, depending on the image. The formula I use is:

$$output = mask. * img + (1 - mask). * bg; \tag{9}$$

Where all math operators are applied point-wise and I is the original image.

3.4 d-h

All the steps from these sections are implemented in the body of the style_transfer function which is implemented in style_transfer.m with the signature:

$$function\ out = style\_transfer(name, img, example) \tag{10}$$

The function returns the transferred image, and saves the output to a file with the name passed in the parameter "name", i.e. "name".jpg The results are presented in 9

3.5 i - Pyramid Blending

I choose the most successful parameters from the previous sections (this is subjective, in my opinion, this images combination has the best results). The results are shown in 10, 11, 12, 13, 14.

$$function\ out\ =\ pyramid\_blending(left, right) \tag{11}$$

Figure 9: Style transfer examples

Figure 10: 1 levels

Figure 11: 3 levels

Figure 12: 6 levels

Figure 13: 8 levels

Figure 14: 10 levels