Answers to questions in Lab 2: Edge detection & Hough transform

Name: Ramadhani Pamapta Putra (930213-6552) Program: TMLEM

Instructions: Complete the lab according to the instructions in the notes and respond to the questions stated below. Keep the answers short and focus on what is essential. Illustrate with figures only when explicitly requested.

Good luck!

Question 1: What do you expect the results to look like and why? Compare the size of *dxtools* with the size of *tools*. Why are these sizes different?

Answers:
The different size is caused by 'valid' argument of convolution function which returns only those parts of the convolution that are computed without the zero-filled edges. Using this ‘valid’ argument :

size(dxtools) = [u_{tools}-u_{deltax}+1, v_{tools}-v_{deltax}+1] ; when size(tools) > size(deltax).
Similar calculation also applies to *dytools*.

Question 2: Is it easy to find a threshold that results in thin edges? Explain why or why not!

Answers:
To find a right threshold for finding thin edges is not easy, since the pixel values on some of the object textures are having a similar value with the edges of the objects. Also, around some of the edges there are greyish textures around them.

In order to have thin edges we need to find sweet spot between higher and lower threshold so that we may acquire edges without sacrificing too much weak ‘edges’ and without showing filtered textures.

Question 3: Does smoothing the image help to find edges?

Answers:
Smoothing the image reduce the pixel value difference between edges and the surrounding textures. This means the gradient magnitude will also be reduced as well which means weaker edges will not be well-preserved. However, smoothening the image results in reduced overall amount of noise. Noise is likely to be detected as an

edge in edge detection process due to noise's strong pixel value difference with its surrounding. In conclusion, if we want to search for only strong edges and we also want to remove noise, smoothening the image can help.

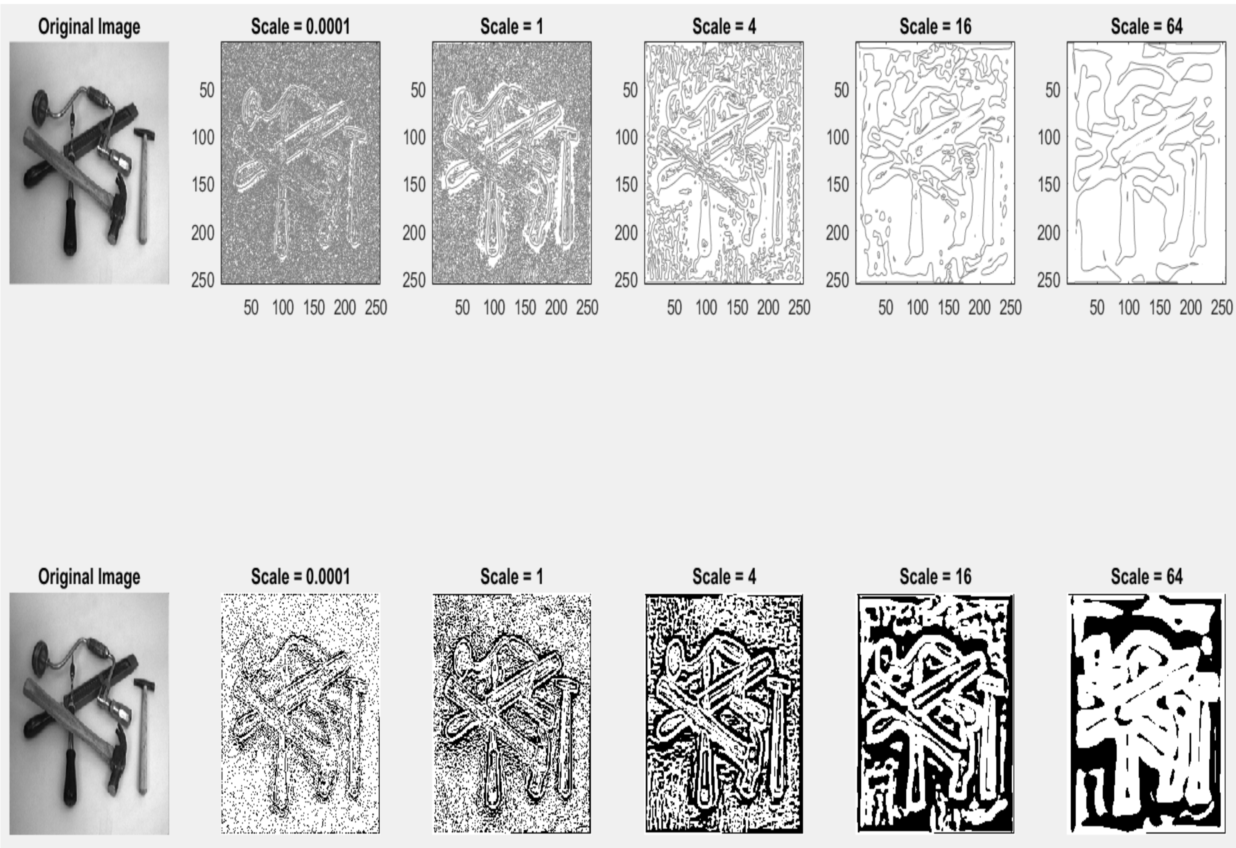
Question 4: What can you observe? Provide explanation based on the generated images.

Answers:
By increasing the value of 'scale' parameter, we can see that the resulting L_{vv} zero crossings will looks simpler and only show clear edges from the original image. However, if we apply 'scale' parameter too high then it will be resulted in oversimplification of edges (some edges will disappear). And if we set the 'scale' parameter too low then a lot of textures(which is not actually edges) will be detected and showed as 'edges', hindering overall edge detection process.

Applying gaussian filter on image will make the frequency original image to be lower. This procedure explains how the reasonable increasing scale parameter will result in more reasonable edges (showed as contours). This is because the textures (which tend to be in higher in frequencies) will be blurred, leaving hard edges (which tend to be in lower frequencies) alone.

Question 5: Assemble the results of the experiment above into an illustrative collage with the *subplot* command. Which are your observations and conclusions?

Answers:



I observed that using greater scale of smoothing, the resulting contour will become more simplified or even distorted, showing only contours from stronger edges only. This is because when we apply smoothing, the overall frequency will become lower, thus making the white edges to be thicker. In case of Lvvv usage, we only detect for a negative value from third derivation signal. The negative value itself is showed by white pixels .

Question 6: How can you use the response from *Lvv* to detect edges, and how can you improve the result by using *Lvvv*?

Answers:

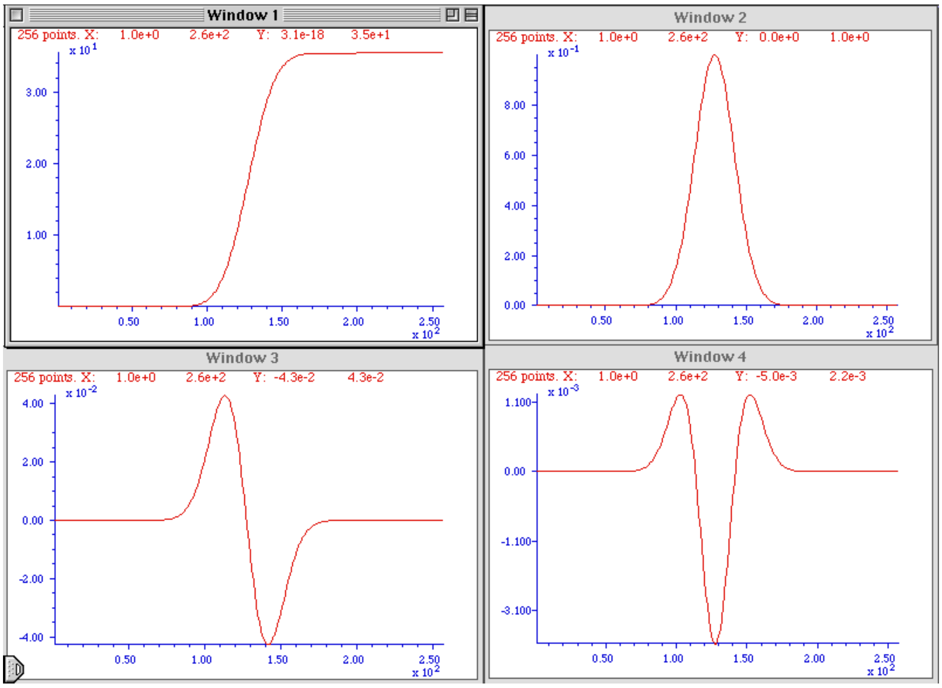


Illustration of a sigmoid functions with its first, second, and third derivative.

In Fig. 2, we can see an illustration of sigmoid function derivatives. The sigmoid can represent a condition in an image where stable lower intensity changes gradually into higher, but still stable intensity. The calculations in Lvv can be illustrated in the second derivation (lower-left), while Lvvv can be illustrated in the third derivation(lower-right)

In case of Lvv, the zero elements are showed as white pixels. In case of Lvvv, any value below zero is showed as white pixels. Both Lvv and Lvvv work in producing distinguishable pixels that represent edges.

However, Lvvv is better because the detected white pixel (as edge) have narrower width and centered at the middle of intensity transition (based on the sigmoid). Also, the negative part of the third derivation is 'protected' by its surrounding positive peaks, making the edge more independent.

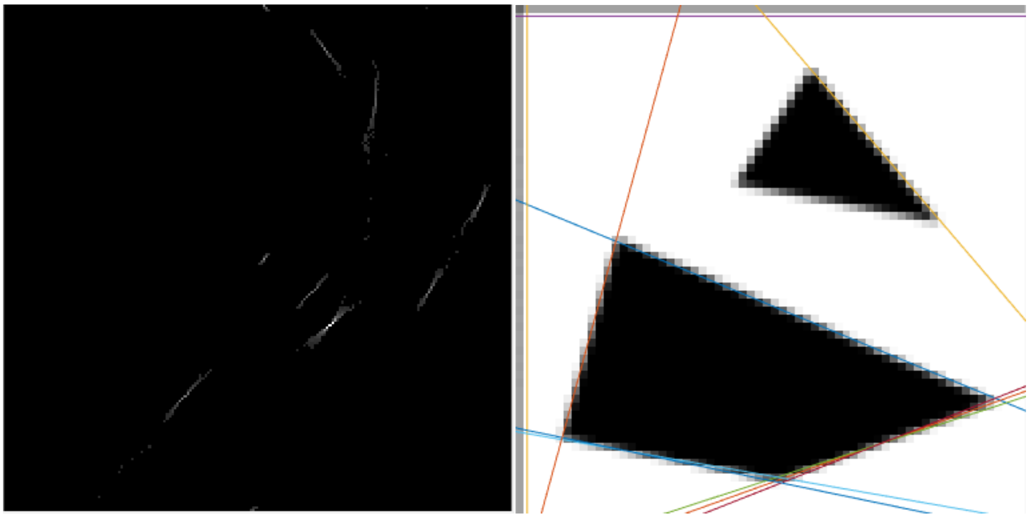
Question 7: Present your best results obtained with *extractedge* for *house* and *tools*.

Answers:

Image	Scale	Threshold	Result
‘tools’	4	10000	
‘house’	4	5500	

Question 8: Identify the correspondences between the strongest peaks in the accumulator and line segments in the output image. Doing so convince yourself that the implementation is correct. Summarize the results of in one or more figures.

Answers:

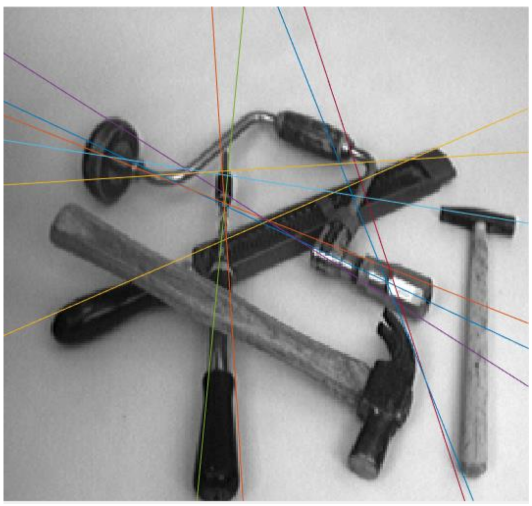


Houghtest256 image with applied Hough detection (10)

Phonecalc256 image with applied Hough detection(20 lines)

Question 9: How do the results and computational time depend on the number of cells in the accumulator?

Answers:



Nrho = 100; ntheta=100; t= 36s

Nrho = 500; ntheta=100; t= 40s

Nrho = 500; ntheta=500; t= 168s

Increasing number of cells in the accumulator makes the resolution in hough space more defined. This will tend to detect more edges accurately compared with the process using lower value of accumulator cells. However, as the resolution is becoming higher by observing more local maximas, the computational load/time may increase. Also, weaker edges are more prone to miss the detection.

Question 10: How do you propose to do this? Try out a function that you would suggest and see if it improves the results. Does it?

Answers:
We can use simple identity function on the gradient edge to increment the accumulator space. Using this function will make a local maxima cell to be stronger than its surrounding cells, if compared with using '+1' increment.