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

Name: Ait lahmouch Nadir **Program**: Machine Learning master

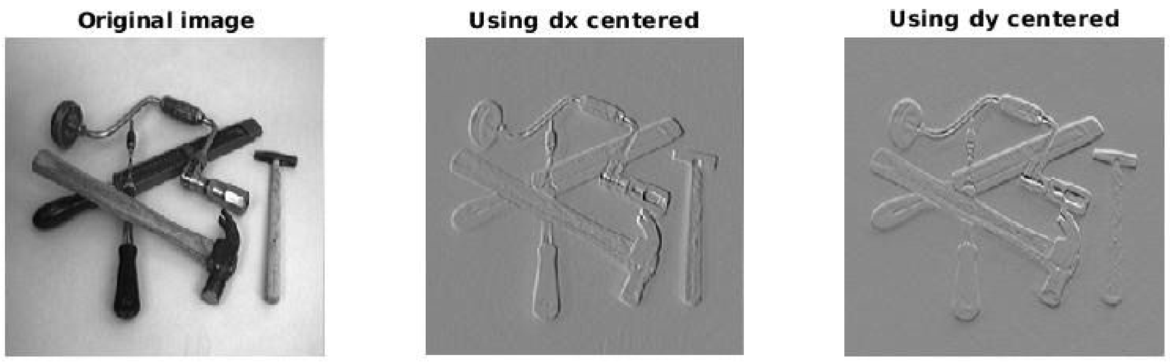
Instructions: Complete the lab according to the instructions in the notes and respond to the questions stated below. Keep the answers short and focus on what is essential. Illustrate with figures only when explicitly requested.

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:

For this question, I used the central differences operator.

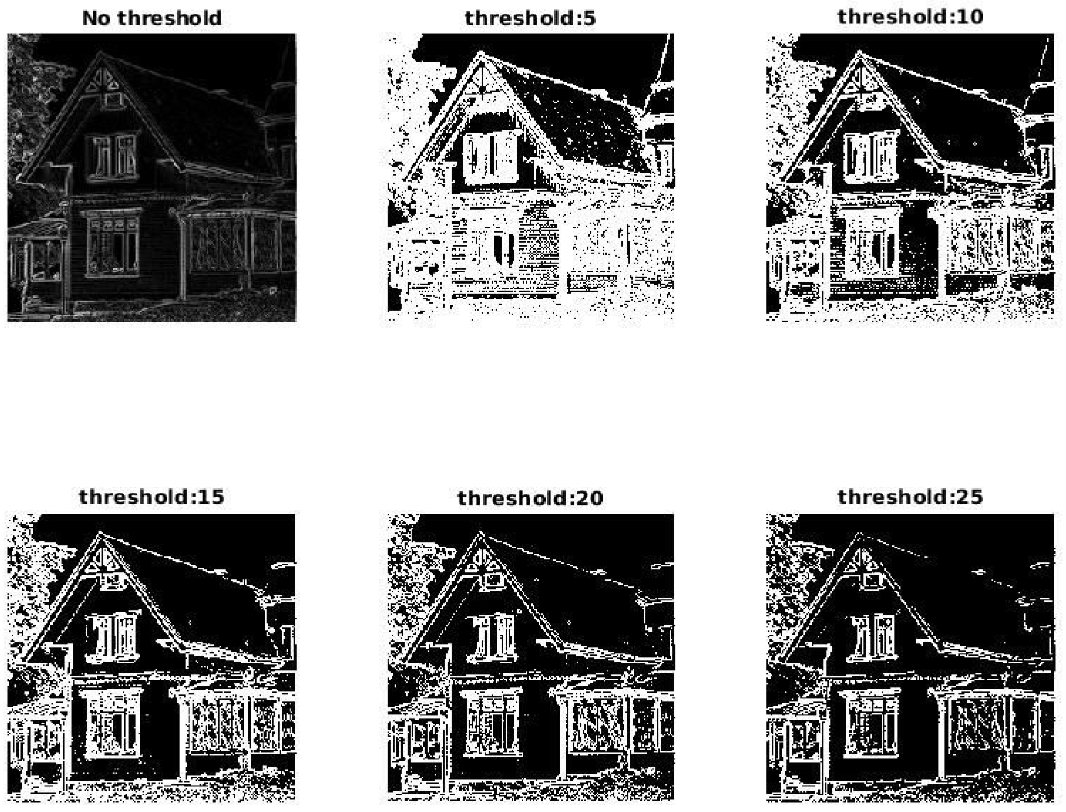


What was expected:

- The picture shows highlights where the edges are because the derivatives we are computing are sensitive to color changing => large magnitude near edges.
- Dxtools emphasizes *x-wise* variations (vertical lines are detected better) and Dytools emphasizes *y-wise* variations (horizontal lines are detected better).
- Dxtools and Dytools sizes are smaller than tools size, because when using the shape parameter as '***valid***' parameter of the conv2 function, this convolution is computed without zero-padded edges, making the result size smaller. Indeed, borders pixels don't have all the neighbors the masks require when computing the convolution.

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

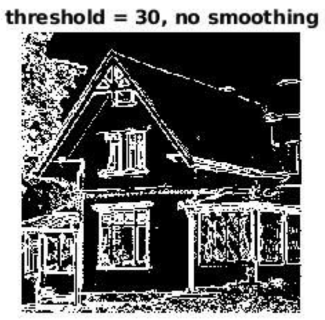
Answers:

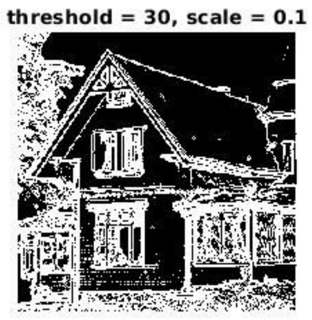


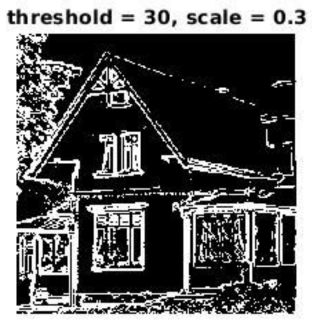
In the figures above, I'm using different thresholds on a non smoothed image.
It's hard to find a perfect value for the threshold to have thin edges, but easy to find a reasonable one (t = 20 seems okay for the house image).
Indeed, when the threshold is low, the thin edges get thick. And when we use a high threshold, some important edges might break or fade (for example, the house roof for t = 25).

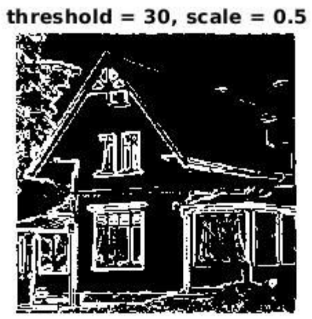
Question 3: Does smoothing the image help to find edges?

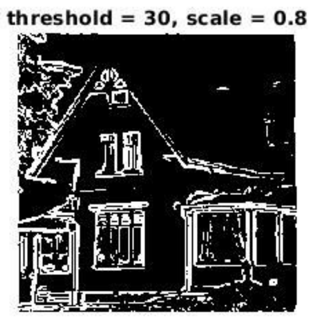
Answers:

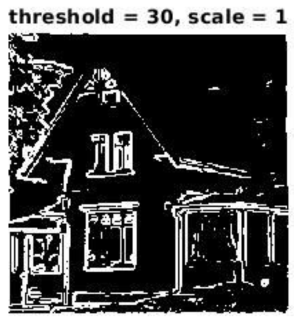








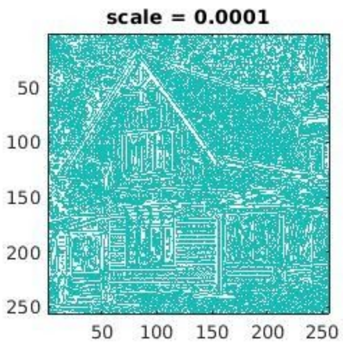


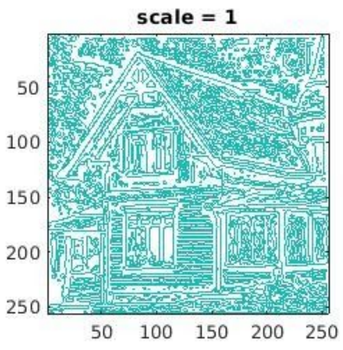


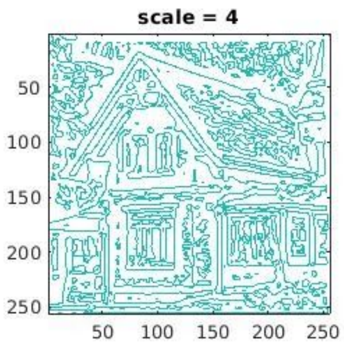
Smoothing might be helpful as it reduces noise (noise in high frequencies). But it also creates a trade off between the scale and the threshold values (makes the task of choosing a threshold value even harder). And it might remove important edges (ex: the house roof when the scale gets bigger).

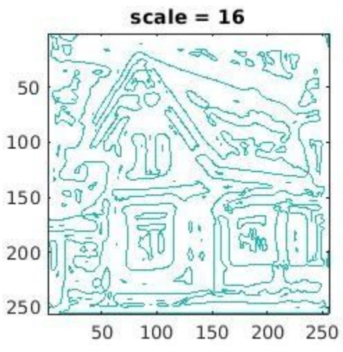
Question 4: What can you observe? Provide explanation based on the generated images.

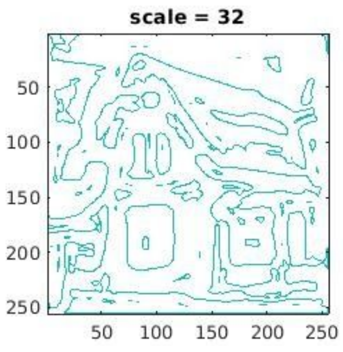
Answers:

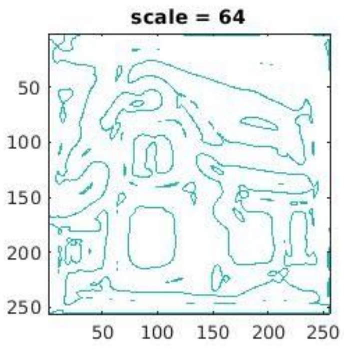








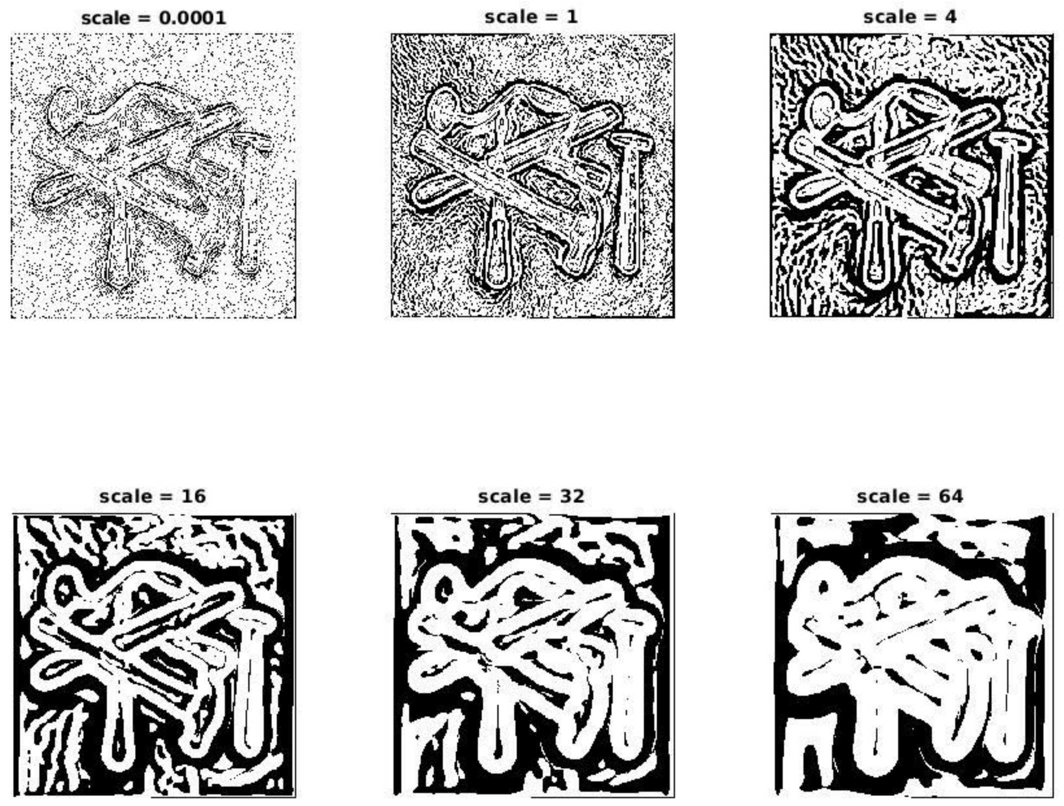




The figure above shows the zero-crossings of the second order derivative Lvv of L, i.e. points where the gradient magnitude (Lv) is at a local extrema.

We can see that when t increases (t being gaussian smoothing variance) i.e the more smoother the image is, , the sparser zero-crossings are and the more they seem distorted (as the edges get smoother and smoother), also some edges fade.

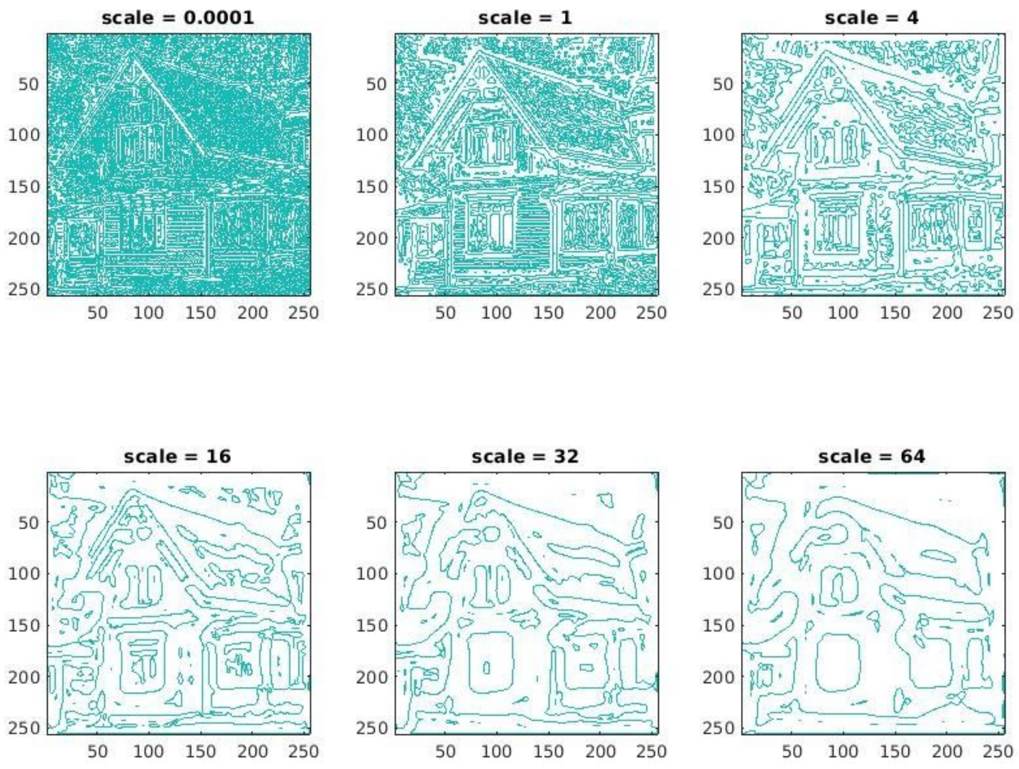
And the reason for that is, in an image a lot of zero-crossings are due to noise and texture details, and with smoothing, there’s less noise and therefore the edges are fewer.

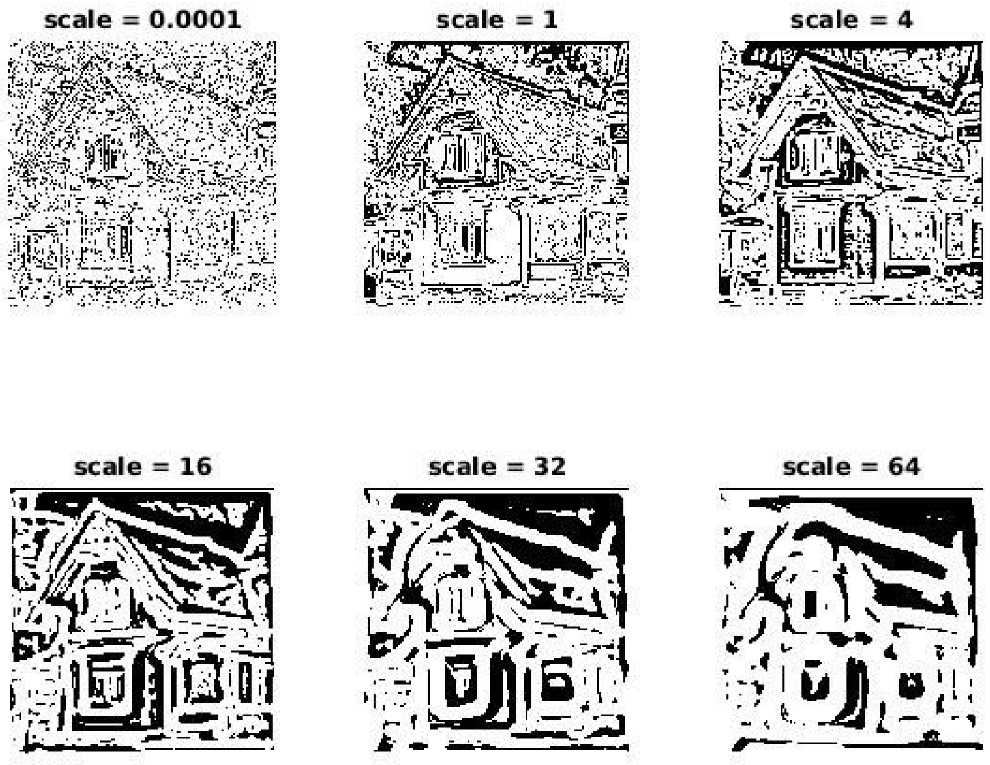


In the figure above, the white areas corresponds to the edges and the all pixels with negative Lvvv.
We can also say remark that when the image is smoother, the edges gets wider.

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

Answers:





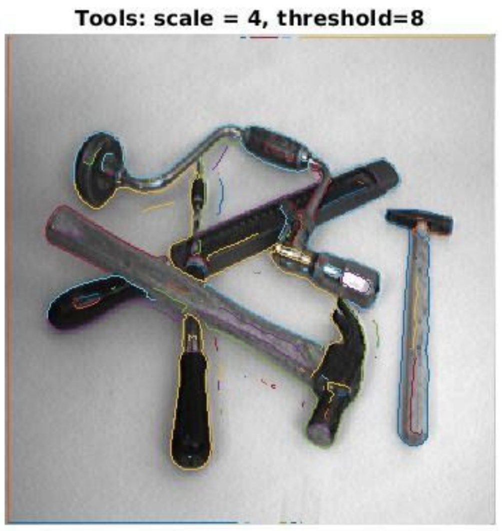
After seeing the plots of Lvvv < 0 and Lvv = 0, we can conclude that a combination of these two conditions can give a good result.

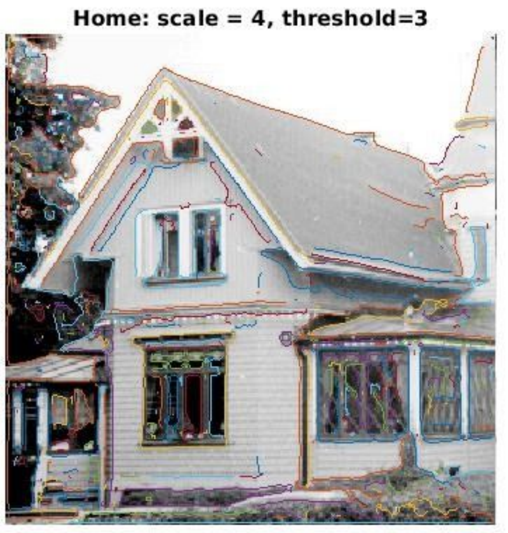
Question 6: How can you use the response from *Lvv* to detect edges, and how can you improve the result by using *Lvvv*?

Answers:
We can first compute the zero crossing from Lvv, and then add the constraint (Lvvv < 0) to only get the local maximas and not all the local extremums.

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

Answers:

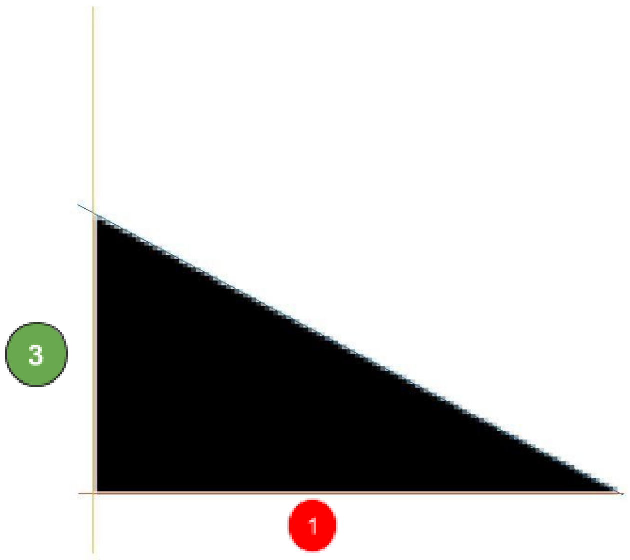


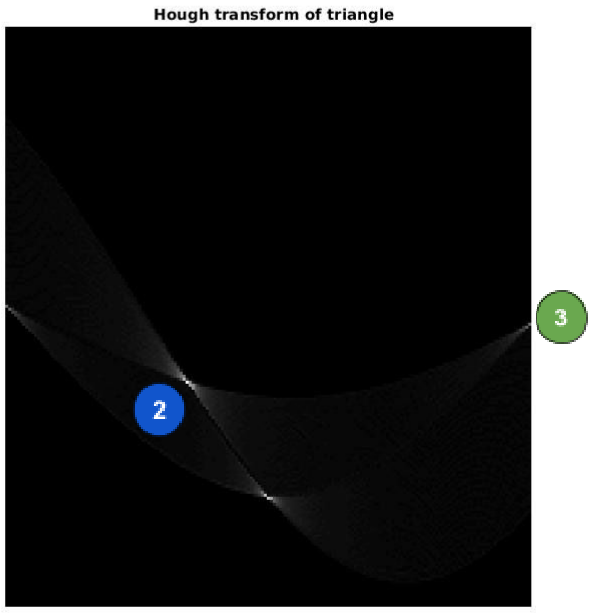


I reached the best results with the parameters above.

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

Answers:

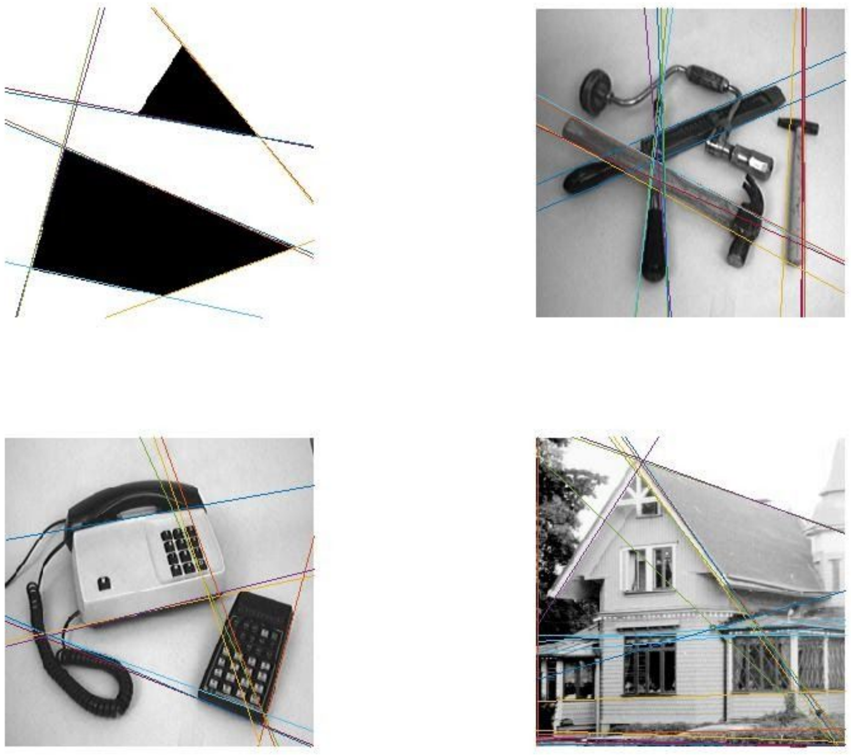




Best matching lines for the triangle test image and corresponding Hough space.

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

Answers:

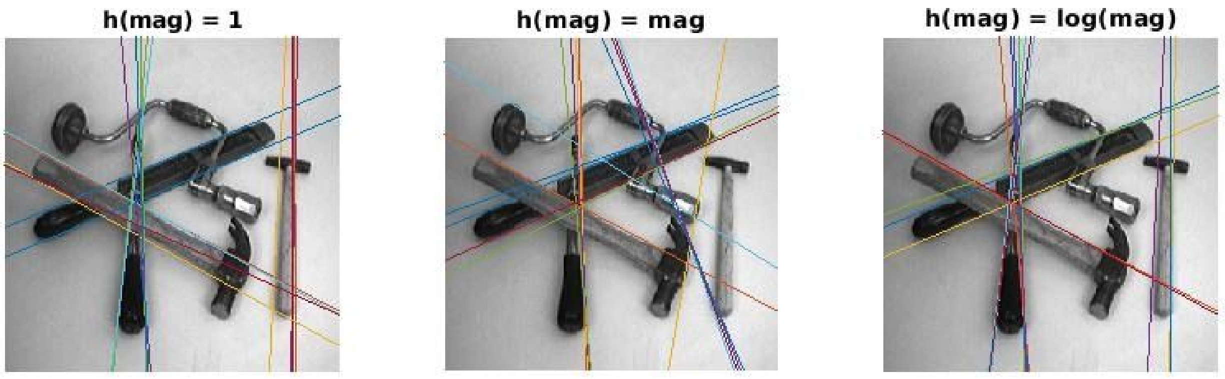


When we increase cells, we get a longer computation time, also we get a better accuracy but we tend to have as a result more lines gathering around sharper edges.

When we decrease them, we get shorter computation time but it cannot detect some edges properly.

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 tried in the figure above different functions h() in incrementing our accumulation matrix. Here's my remarks:
Incrementing with gradient magnitude reduces critical dependency on threshold.
The function h(mag) = 1 treats noisy pixels in the same way as it treats the rest of the points.
When we use h(mag) = log(mag), we helps finding edges with lower gradient magnitude.
