Answers to questions in Lab 2: Edge Detection & Hough Transform

December 1, 2023

Name: Leonardo De Clara

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?

Answer:
dxtools is expected to show variations of the image on the x-axis, therefore casting lights on vertical edges. *dytools*, on the other hand, should show variations of the image on the y-axis, highlighting horizontal edges.
The input image's size is 256×256, while *dxtools*'s size is 254×254: this result is due to the fact that we are performing a convolution using *valid* mode, therefore we are not padding our input with 0s in order to get an output with same size as the input. As a matter of fact, in order to compute the derivative at one point we need neighbour points before and after, which is not possible for points on the border.

Question 2.

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

Answer:
Finding a proper threshold resulting in thin edges is not an easy task: using the gradient's magnitude histogram is not helpful as it does not show any valley whose relative value could be used as threshold.
By manually tuning the threshold parameter we can see the low values lead to rather thick edges, while higher ones are able to achieve thinner edges, but they may lead not to detect some subtle ones.

Question 3.

Does smoothing the image help to find edges?

Answer:
Smoothing significantly helps detecting edges since the image could be affected by noise, leading to false edges, and Gaussian smoothing acts as a low-pass filter, reducing high-frequency noise in the image. Moreover it has the property of smoothing while preserving edges, maintaining the important structural features of the image. This effect is clear when applied, for example, to the house's image: the same threshold value can lead to very different results as in the smoothed version some "false" edges are not detected when compared to the non-smoothed version. However, the sigma needs to be properly tuned as big values may lead to distortion of true structures.

Question 4.

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

Answer:
As the images show smaller scale values make descriptors more sensitive to noise and local variations that may not be related to edges. Alternatively, big scale values lead to significant loss of details, making it impossible to recognize relevant edges, detecting only the most pronounced contours of the image.

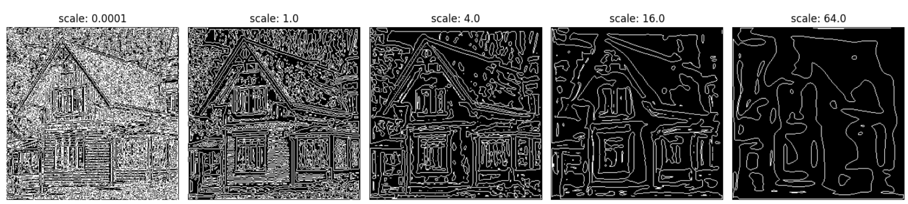

Figure 1: Zero crossing of $\tilde{L}_{vv}$ using different scale values

Question 5.

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

Answer:
We can observe that white areas in the plots correspond to edges and pixels

with $\tilde{L}_{vvv} < 0$, thus negative third order derivatives in the gradient direction. In the case of the sign of the third derivative, as the scale increases contours become thicker and thicker, making it difficult to visualize the image's edges. Small scale values instead are not able to properly define correct edges.

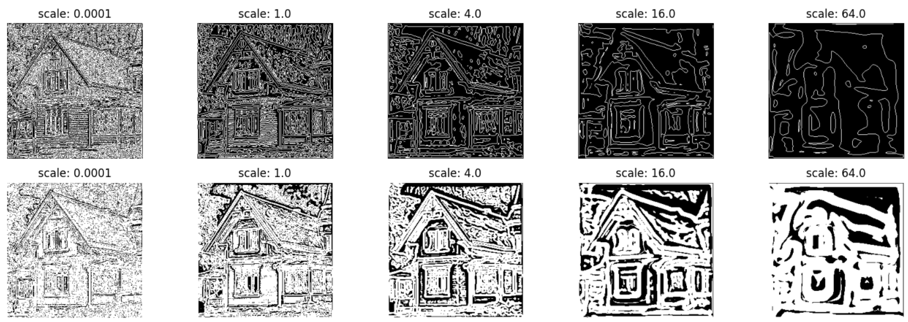

Figure 2: Comparison between zero crossing of $\tilde{L}_{vv}$ sign of the third order derivative in the gradient direction using different scale values

Question 6.

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

Answer:
Both results obtained by setting $\tilde{L}_{vvv} < 0$ or $\tilde{L}_{vv} = 0$ are able to describe some of the image's edges, but are not really perfect individually: this is due to the fact that a single edge criteria is checked. When both conditions are satisfied we are sure that the gradient magnitude reaches local maxima in the gradient's direction, thus we are showing only the pixels representing edges.

Question 7.

Present your best results obtained with extractedge() for *house* and *tools*.

Answer:
After testing different parameter values, the best results are obtained by setting the scale to 4.0 and the gradient's threshold to 5.0 for the house picture, while a scale of 2.0 and a threshold of 8.0 are able to provide the best results for the tools picture.

(a) Scale = 4.0, threshold = 5.0

(b) Scale = 2.0, threshold = 8.0

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 in one or more figures.

Answer:
Figure 4 shows the correspondences between edges in the original image and points in the accumulator space, represented as a vector (ρ, θ). It is possible to recognize the vertical edge in the accumulator space, represented by a point with ρ value close to 0 and a θ value close to $\frac{\pi}{2}$. The horizontal edge is mapped to a positive ρ value and a θ value close to 0. Ultimately, the diagonal edge's correspondence in the accumulator space is a point with small positive ρ value and a small negative θ value.

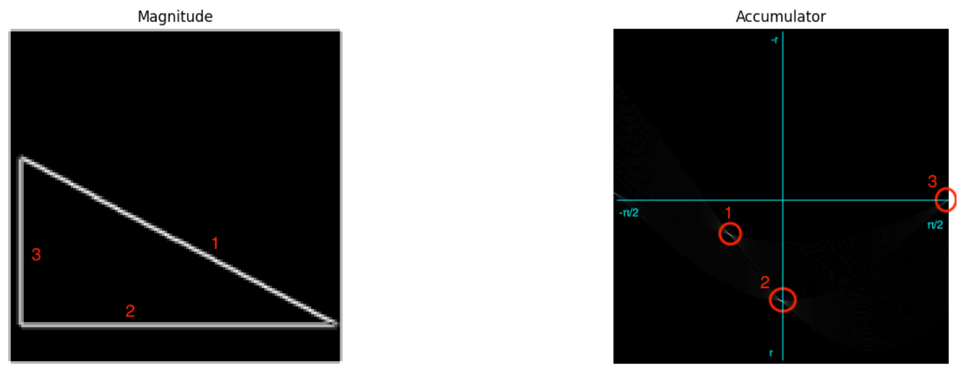

Figure 4: Correspondences between edges and points in the accumulator space

Question 9.

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

Answer:
A low number of cells, thus a coarse accumulator space, leads to poor resolution results and imprecise line directions as it more difficult to find the best-matching line with a smaller range of parameters. On the other hand, a finer accumulator space (more cells) can provide higher quality results by finding the best match for edge line. However, with too many cells we tend to obtain, for a single edge, multiple lines that are slight variations of each other since we have too many accumulators and too few samples per accumulator.

The computational time for computing the Hough transform of an image increases with the number of cells in the accumulator space since the algorithm will need to check an higher number of cells corresponding to (ρ, θ) combinations. As a matter of fact the core of function `houghline` has a time complexity of $\mathcal{O}(M n_{theta} n_{rho})$ for an image with M edge points.

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?

Answer:
By taking as weight the gradient's magnitude (opposed to a unitary weight as

before) we tend to highlight more defined edges as their gradient's magnitude will be bigger: the results are lines limited to more decisive edges, while lines for lighter edges tend to disappear. Opposed to that, by talking as weight the logarithm of the gradient's magnitude the difference in magnitude between clearer and lighter edges is reduced and they are assigned more similar weights when compared to the previous case: this way ligher edges are more probable to be recognized and assigned a line.