DD2423 Image Analysis and Computer Vision
Lab#2

Giovanni Castellano, giocas@kth.se

Date: 2022/11/16

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!

1 Difference operators

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?*

   We chose to use Sobel operator. As shown in figure 1 `dxtools` highlights the vertical edges. `dytools` instead, highlights the horizontal edges.
   The original size of tools is $256, 256$. After the convolution it decreases to $254, 254$. We loose two pixels for each dimension due to the "valid" parameter in the convolution function. Basically, we are not performing the zero-padding, therefore the size of the images decreases after convolution.

2 Point–wise thresholding of gradient magnitudes

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

   It is not easy to find a good threshold (figure 2). If the threshold is too low we get thick edges. On the other hand, if it is too high some edges will disappear.

3. *Does smoothing the image help to find edges?*

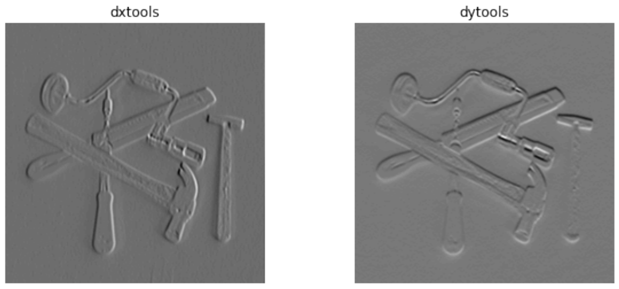

Figure 1: x and y partial derivatives.

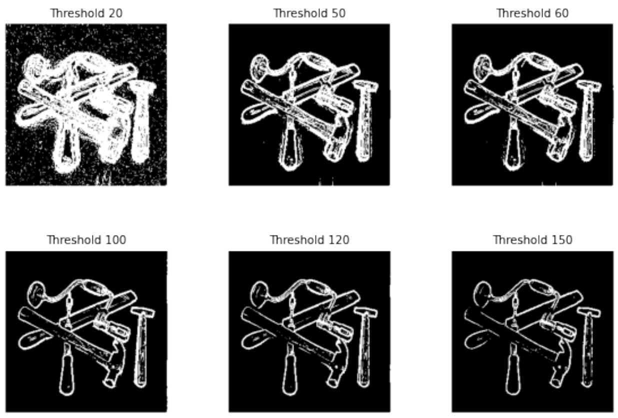

Figure 2: resulting edges as we increase the threshold on the gradient magnitude

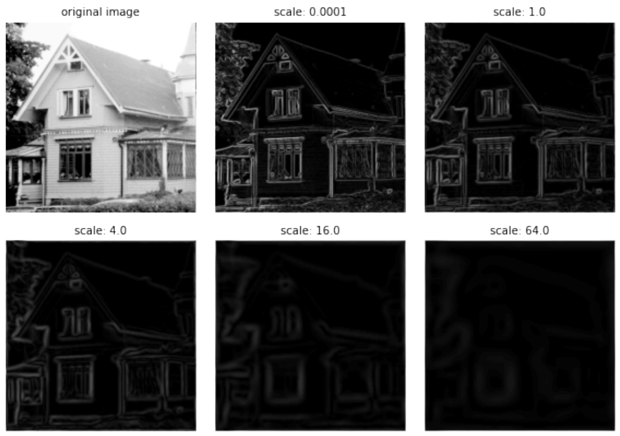

Figure 3: how filtering the image with a gaussian filter affects the results.

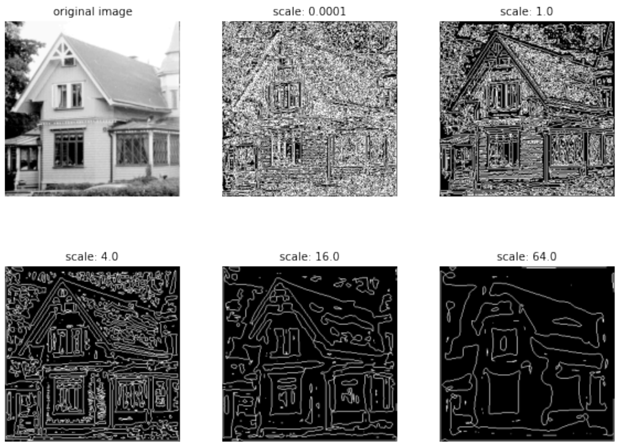

Figure 4: points where lvv=0.

The gaussian filter removes the noise from the image. This is very useful because we will remove a lot of fake edges. However, if we blur the image too much we will also lose the important edges. (figure 3)

4 Computing differential geometry descriptors

4. What can you observe? Provide explanation based on the generated images.
   Plotting all the points where the second-order derivative is equal to zero (figure 4) we observe that the resulting image highlights the borders. However, when the second-order derivative is equal to zero it could also be that we have a minimum in the gradient magnitude, which means that we are also highlighting points that are not on the edges.
   Plotting the points where the third-order derivative is less than zero (figure 5) we observe that the resulting image contains very thick edges. This happens because in general, the third derivative is less than zero around the edges. Scaling the image is very important to obtain decent results, but if we scale too much the shape of the edges changes.

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

   From the images, we can conclude that we need both conditions.

6. *How can you use the response from Lvv to detect edges, and how can you improve the result by using Lvvv?*
   We first find the points where the second-order derivative is equal to zero. Afterward, we only keep the points where the third-order derivate is less than zero. In this way, we discard all the points corresponding to a local minimum in the gradient magnitude.

5 Extraction of edge segments

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

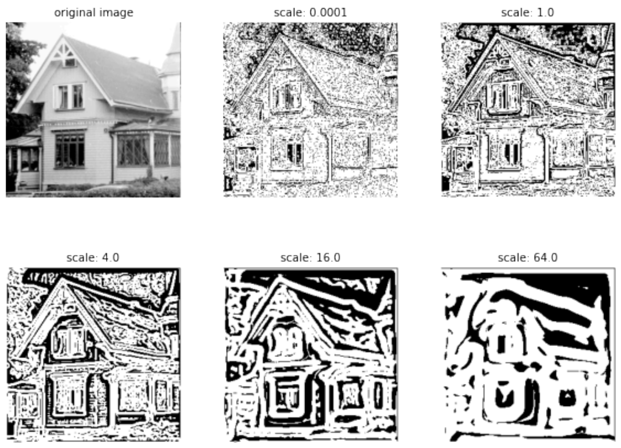

Figure 5: points where lvvv less than 0

We obtained the best results with scale = 4. In particular, we observe in figure 6 that the best results are obtained with a threshold value of 40 or 60. Increasing the threshold the edges will disappear

6 Hough transform

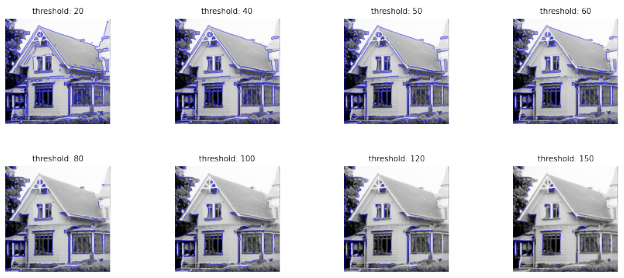

Figure 6: edges detected with scale=4 for different value of threshold

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.

   Figures 7, 8, 9 and 10 show the results we obtained. It is clear from the results that our implementation is working. We observe that the hough transform highlights some of the points. Each of these points corresponds to a line in the space domain.

9. How do the results and computational time depend on the number of cells in the accumulator?
   As we increase the resolution in the Hough domain the computational time increases. If the resolution is too high we will get multiple lines for the same edge. If the resolution is too low, however, we will not find the correct edges. The computational time depends especially on the number of theta parameters. This is due to the for loop over thetas in the implementation.

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?
   we tried to increment the accumulator according to the log of the gradient magnitude but we did not observe any big difference. However, it makes sense that the points with higher magnitude in the space domain should have a higher importance in the hough domain. This is because if the magnitude is high the point is more likely to belong to an edge.

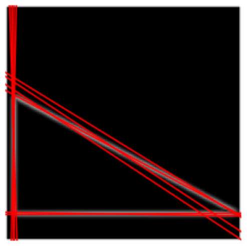
Figure 7: Lines detected in the space domain using a simple image

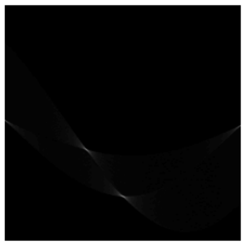
Figure 8: Hough transform of the image

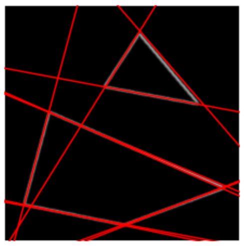

Figure 9: Lines detected in the space domain using a more complicated image

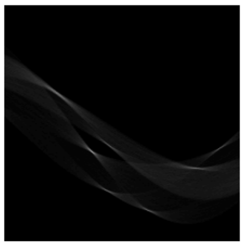

Figure 10: Hough transform of the image