Lab 2: Edge detection and Hough Transform

DD2423 Image analysis and Computer Vision, 2019

Federico Favia
favia@kth.se

Martin De Pellegrini
martindp@kth.se

December 12, 2019

1 Difference operators

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?*

Results are shown in Fig. 1 with both a simple difference operator and the more used central difference operator: `dxtools` emphasizes differences in x-direction (stronger responses for vertical lines), while for `dytools` it is the contrary. They will detect a large gradient magnitude near edges (positive or negative crease at edges) hence the grey-level will be close to zero for most part because it has flat background. `dxtools` is smaller than `tools` because of the 2-D convolution with 'valid' flag, therefore `tools` is not-zero padded at edges (the simple difference operator is a 1×3 matrix).

2 Point-wise thresholding of gradient magnitudes

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

As shown in Fig. 2, it is hard to find a perfect threshold but it is feasible

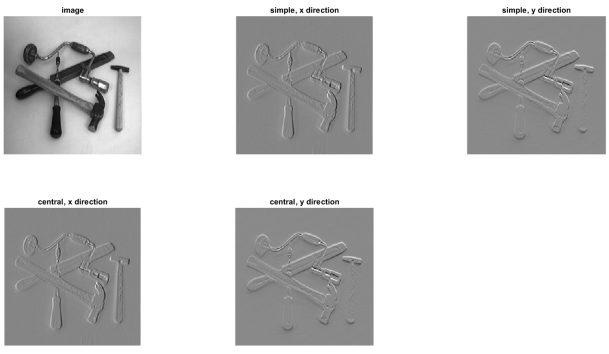

Figure 1: 1^{st} ord. partial derivatives applied in two orthogonal directions, implemented as simple difference operator and central difference operator.

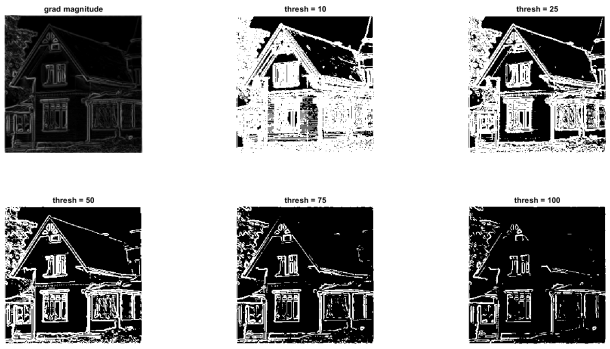

Figure 2: Result for thresholds = {10, 25, 50, 75, 100} and variance of the Gaussian smoothing $\sigma^2 = 0.6$.

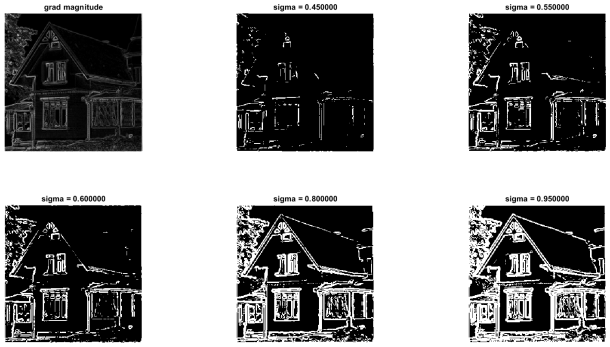

Figure 3: Result for Gaussian smoothing $\sigma^2 = \{0.45, 0.55, 0.6, 0.8, 0.95\}$ and threshold = 90.0.

to find an acceptable one, because edges are not always sharp. With an high threshold, weak edges can fade while with a lower one sharp edges become wide and results seems worse. Experiments have been conducted performing a Gaussian smoothing before computing the magnitude, with a $\sigma^2 = 0.6$, and threshold = {10, 25, 50, 75, 100}.

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

Smoothing (Fig. 3) helps a little because it gets rid of high frequencies noise, but choosing the threshold it is not so easy. Indeed smoothing the image with a Gaussian of $\sigma^2 = 0.3$ we could lose edge information or distort it, therefore a compromise with the degree of smoothing must be reached.

3 Computing differential geometry descriptors

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

In Fig. 4 we can see that with a small variance (σ^2) of the Gaussian smoothing

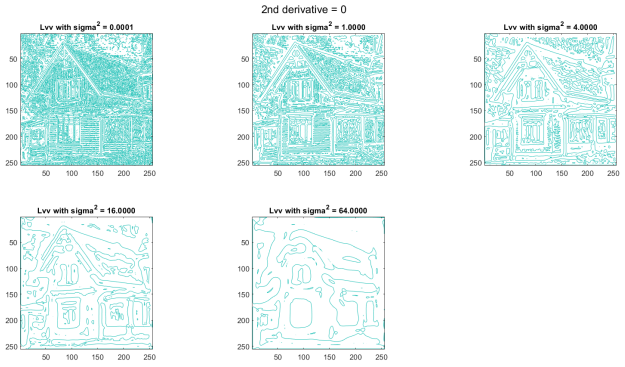

Figure 4: 2^{nd} ord. derivative zero-crossings with Gaussian smoothing $\sigma^2 = \{0.0001, 1.0, 4.0, 16.0, 64.0\}$.

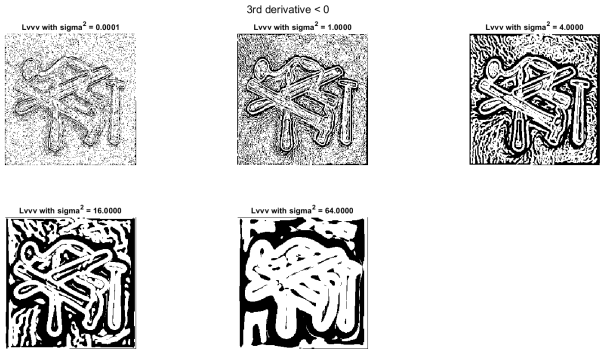

Figure 5: 3^{rd} ord. derivative smaller than zero with Gaussian smoothing $\sigma^2 = \{0.0001, 1.0, 4.0, 16.0, 64.0\}$.

before computing the 2^{nd} order derivative equal to zero, we have a noisy image, with every tiny detail (edge) shown by many curves. With a large variance, edge curves are clearer and more sparse. With a too large variance some curves vanish and distort. This happens because too much smoothing distort edges while with very few smoothing we have many zero-crossings due to minute high frequencies (such as texture details and noise). A normal smoothing instead allows to have fewer noises in 2^{nd} order zero-crossing contour.

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

As shown in Fig. 5, we cannot observe a good result since we lose a clear vision of edges: in the white area we have edges and pixels with negative 3^{rd} order derivatives. If the image is smoothed, the edges are weaker and the white region becomes wider. Smoothing at certain level can help to emphasize edges (get rid of noise), but doing it too much erode other parts of the object.

Question 6: *How can you use the response from* $L_{vv}{}^{tilde}$ *to detect edges, and how can you improve the result by using* $L_{vvv}{}^{tilde}$*?*

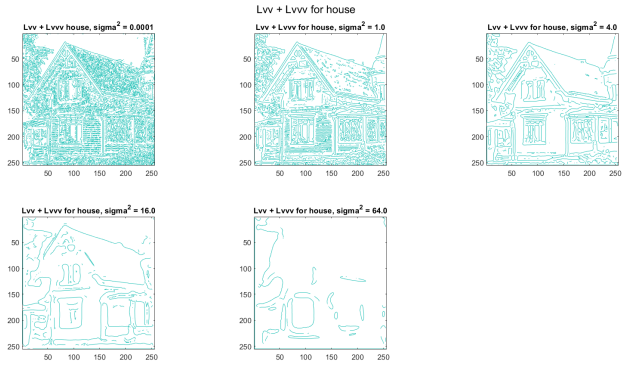

Figure 6: Combination of L_{vv} and L_{vvv} with Gaussian smoothing $\sigma^2 = \{0.0001, 1.0, 4.0, 16.0, 64.0\}$.

Figure 7: Best extracted edges for house and tools with respective parameters.

As shown in Fig. 6, theoretically we could improve the results adding a sort of threshold for L_v to get rid of little edges, namely adding the constraint of negative 3^{rd} order derivative to avoid rough detection after picking points with 2^{nd} order derivative equal to zero. Since the noise and tiny details are a problem for complex images, we do not have a substantial improvement, apart from $\sigma^2 = 1.0$ and $\sigma^2 = 4.0$.

4 Extraction of edge segments

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

We can see in Fig. 7 our best results for extracting edges on images `house` and `tools`. In particular for the first one $\sigma^2 = 4.0$, `threshold` $= 3.5$ have been used, while for the second one $\sigma^2 = 4.0$, `threshold` $= 8.0$.

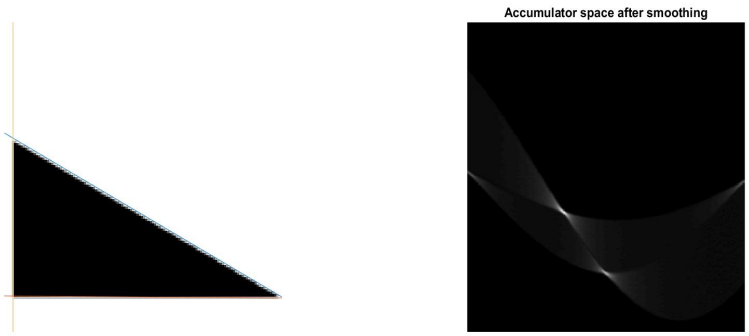

Figure 8: Results of the function houghedgelines.m applied on the test image triangle128.

5 Hough Transform

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.*

In Fig. 8 are presented the output of the function `houghedgeline(...)` applied to the test image `triangle128` specifying the following parameters:

- `scale=4;`
- `threshold=5;`
- `nrho=size(testimage1,1)*2;`
- `ntheta=size(testimage1,2)*2;`
- `nlines=3;`

As it is possible to notice that in the accumulator space there are three peaks (one is split on the borders due to the vertical line). These peaks correspond to all the points laying on the edges of the triangle.

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

Increasing the number of cells leads to a longer computational time, with local *maxima* near the edge line and more lines gathered at sharper edges, a good practice could be increasing the number of lines that have to be displayed. While if we do the contrary we cannot detect edges accurately but we will obtain a shorter computational time. As we can see in Fig. 9, for large values of $\Delta\rho$ and $\Delta\theta$ the lines will be of low accuracy, while small values may may result in multiple responses for the same line.

5.1 Choice of accumulator incrementation function

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?*

Figure 9: Results of changing resolution of the accumulator space.
ntheta = {size(testimage1,2), size(testimage1,2)*4, size(testimage1,2)*9};
nrho = {size(testimage1,1), size(testimage1,1)*4, size(testimage1,1)*9}

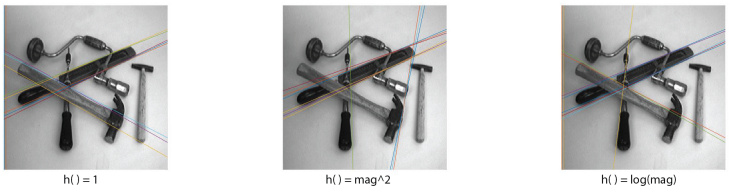

Figure 10: Result with different increments in the accumulator space.

An increment with gradient magnitude reduces critical dependency on threshold. With $h(\cdot) = 1$ we treat noise in the same manner as other points, but some parts of the image have large noise. If $h(\cdot) = log(mag)$ we increase the possibility of edges with lower values of gradient magnitudes. In this way, points may not be from the same edge curves (and some incorrect lines can appear). The results are shown in Fig. 10.

References

[1] Rafael C. Gonzalez and Richard E. Woods, *Digital Image Processing*, Prentice Hall, 2nd ed., 2002

[2] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2010