

Answers to questions in

Lab 2: Edge detection & Hough transform

Name: Arthur Caron & Côme Lassarat

Program: _____

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.

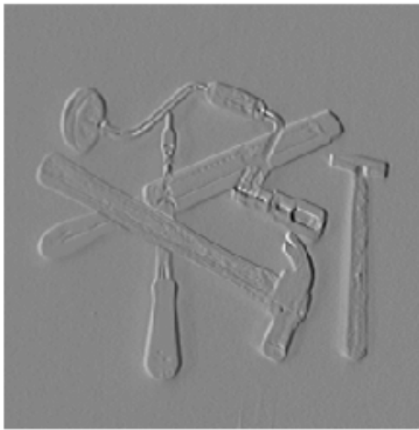
Good luck!

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

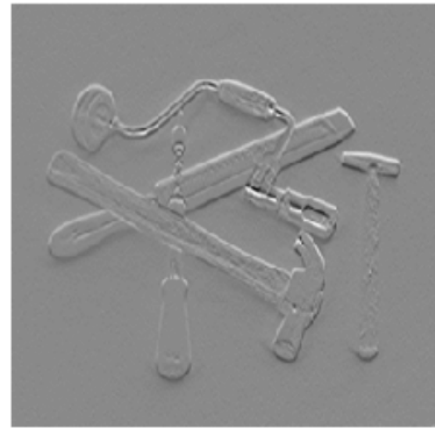
Answers:



Original image



dxtools



dytools

On '*dxtools*', we expect to see the edges especially in the x-direction whereas on '*dytools*', we expect to see the edges especially in the y-direction. The gradient operator measures the variation of pixel intensity in a given intensity.

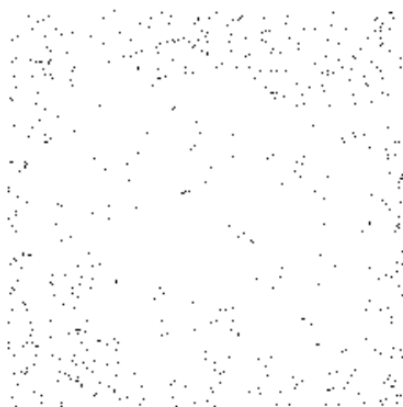
Furthermore, the size of '*tools*' is 256x256 pixels.

The size of '*dxtools*' is 256x254.

This is explainable because the convolution that gives '*dxtools*' applies neighbours-based operations: for a given pixel, *deltax()* computes the previous pixel and the following pixel on the x-axis. Given the two existing borders, that's why there is 2 pixels less on the x-axis of '*dxtools*'.

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

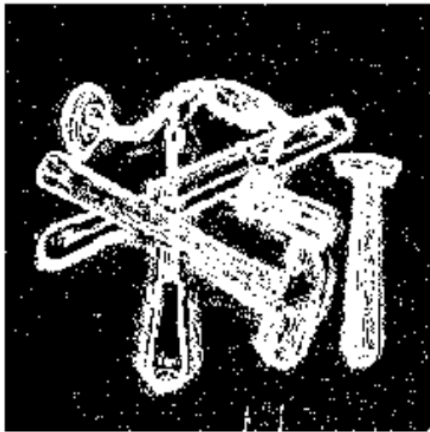
Answers:



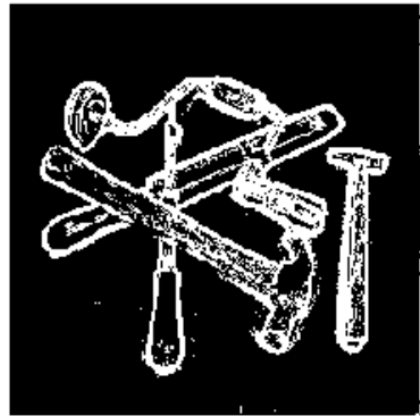
threshold = 0.1



threshold = 1



threshold = 5



threshold = 10



threshold = 20



threshold = 50



threshold = 100

It is not easy to find a threshold resulting in thin edges:



- as the threshold decreases, it is more likely to detect “false” edges and noise.
- as the threshold increases, it is more likely to not detect edges and real features of the image

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

Answers:



Original image

Threshold	Without smoothing	With smoothing (scale = 1)
5		

10



20



50



→ Smoothing helps to find edges (see threshold = 10). Indeed, smoothing through the Gaussian filter acts like a low-pass filter which allows to remove noise.

However, smoothing also has downsides: when smoothing is too important, it is more complicated to detect edges.

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

Answers:

Points such as $L_{vvtilde}() = 0 \rightarrow$ gradient magnitude reaches a local extremum



scale = 0.001



scale = 1



scale = 4



scale = 16



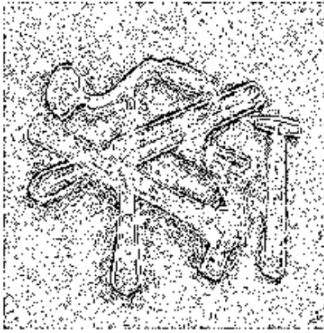
scale = 64

Zero-crossing of $L_{vvtilde}()$ is sensitive to noise (see picture with scale = 0.001): we use a Gaussian filter on the original image.

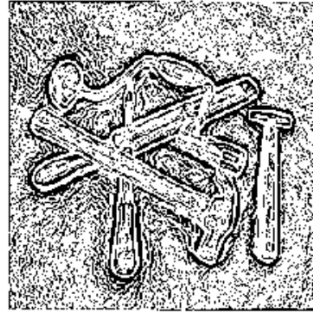
However, as the scale increases, fewer edges are detected (an edge occurs when $L_{vvtilde}() = 0$) and the ones detected aren't faithful to reality: the original image is too blurred by the Gaussian filter.

But checking only if $L_{vvtilde}() = 0$ doesn't give information about the intensity of the gradient: need to check if it is a local maximum by looking at the sign of $L_{vvvtilde}()$.

Points such as $L_{vv\tilde{t}} < 0$



scale = 0.0001



scale = 1



scale = 4



scale = 16



scale = 64

Edges can also be detected by looking at the sign of $L_{vv\tilde{t}}$: this allows us to check whether the gradient magnitude has its slope that increases or decreases.

Having the condition $L_{vv\tilde{t}} > 0$ instead of $L_{vv\tilde{t}} < 0$ only inverts the black and white.

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

Answers:

Please see above.

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

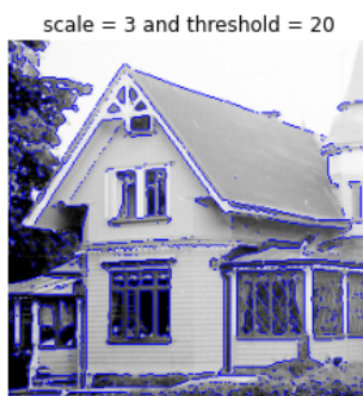
Answers:

Lvvtilde can be used to detect edges by plotting points where Lvvtilde = 0. Then, from these points, we can only select the ones where Lvvtilde < 0, which corresponds to a local maximum. This last step removes the points that are not gradient magnitude maxima.

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

Answers:

Best results:



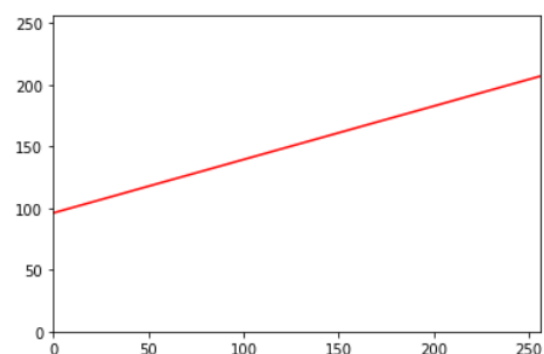
Question 8: Identify the correspondences between the strongest peaks in the accumulator and line segments in the output image. Doing so convince yourself that the implementation is correct. Summarize the results of in one or more figures.

Answers:

In the following picture, we call houghedline function with the following parameters :
`houghedline(testimage2,100,0.1,120,120,10)`

The most important one being $\text{nrho}=\text{ntheta}=120$ and picking the 1 biggest peaks and line segments. We choose only one peak to see if the implementation is correct.

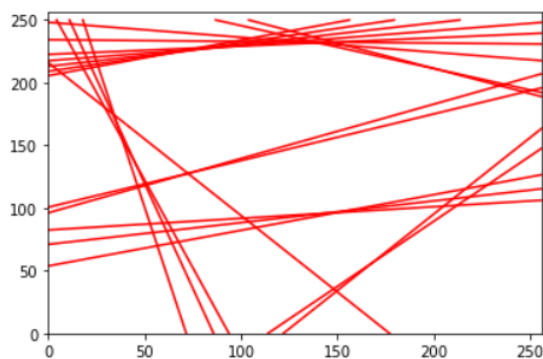
Theta varies between $-\pi/2$ and $\pi/2$. and rho goes up to 362 in the following case :



The line corresponds to the point with value $\theta = -0,41$ and $\rho = 88$



We then verify the implementation with more lines and by comparing it to the original



picture. (with 15 lines)

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

Answers:

The more cells there are, the longer computational time is and results are more difficult to interpret (resolution/samples tradeoff).

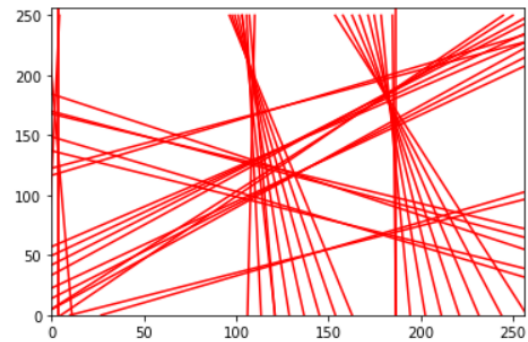
Question 10: How do you propose to do this? Try out a function that you would suggest and see if it improves the results. Does it?

Answers:

We can just add a function that uses the “magnitude” matrix in the arguments of the fonction houghline, then we can use the identity function for h.

We also tried to use the log function in order to not give too much weight for points with a magnitude that's too high compared to other points. In this case the log function would detect details edges more easily.

nrho=100 ntheta=400 lines=40 $h(x)=x$



nrho=100 ntheta=400 lines=40 $h(x)=\log(x)$

