Dpto. Ingeniería de Sistemas y Automática
Universidad de Málaga

COMPUTER VISION

EXERCISE 2a: Edges Detection with mVision

Concepts: Gradient operator: Sobel. Drog, Canny

1. Load the *edges* GUI from the *mVisionGUI* menu.
2. Load the *piezas.bmp* image.
3. Detect edges using Sobel3 (the Sobel operator with a 3x3 kernel) and Sobel5 (with a 5x5 kernel). You can play a bit with the gradient image in Sobel3 since it shows the gradient direction clicking on it.
    a. Which is the difference between them (in the result image) with the default threshold (20, pixel intensity in the gradient image for being considered as an edge)?

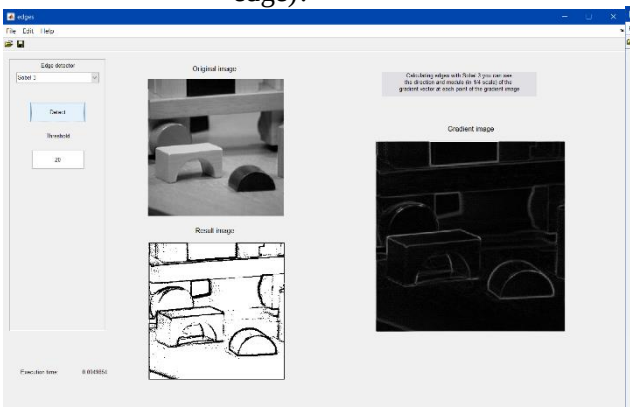

Figure 1

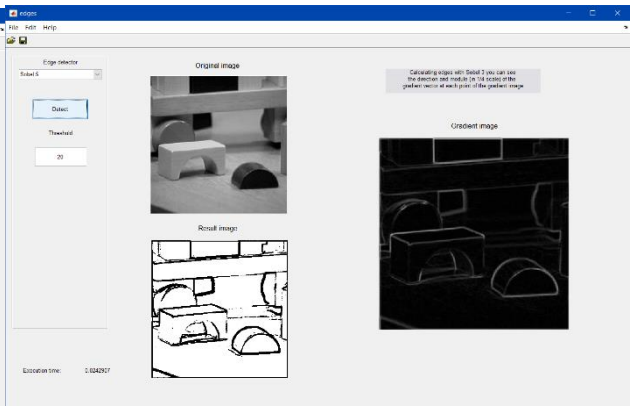

Figure 2

The difference is the Sobel-5 detects less edges than Sobel-3. However, Sobel-3 captures more false edges than the other.

b. Try both kernels with different thresholds: 20, 40 and 60. What is happening?

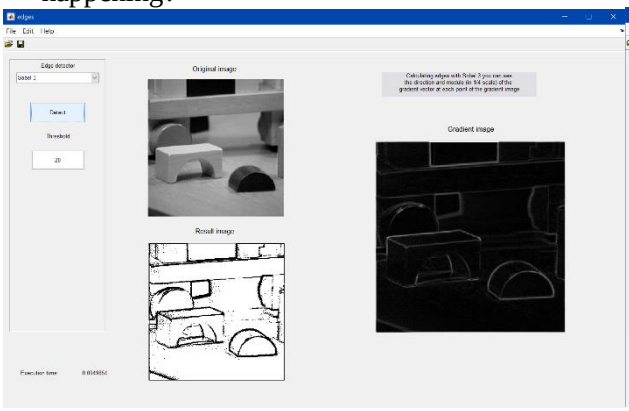

Figure 3

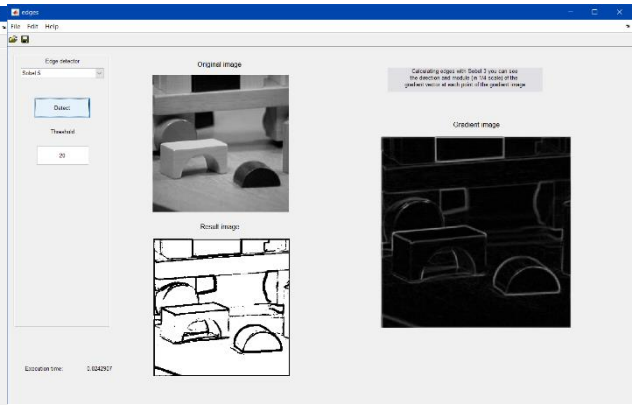

Figure 4

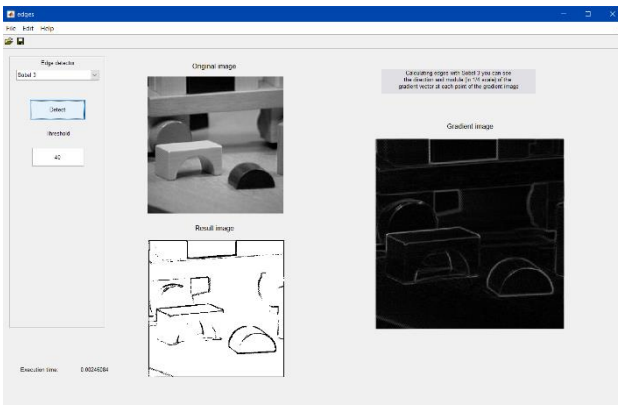

Figure 5

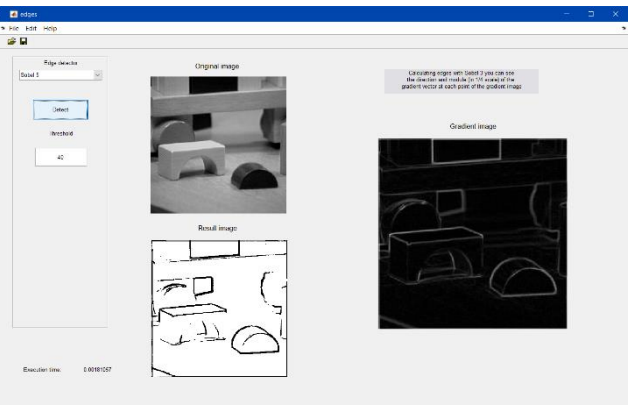

Figure 6

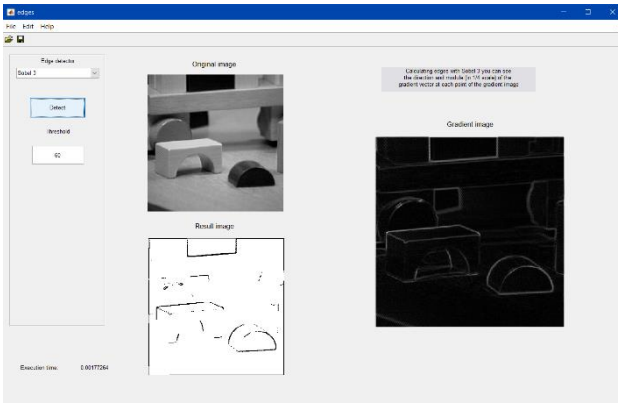

Figure 7

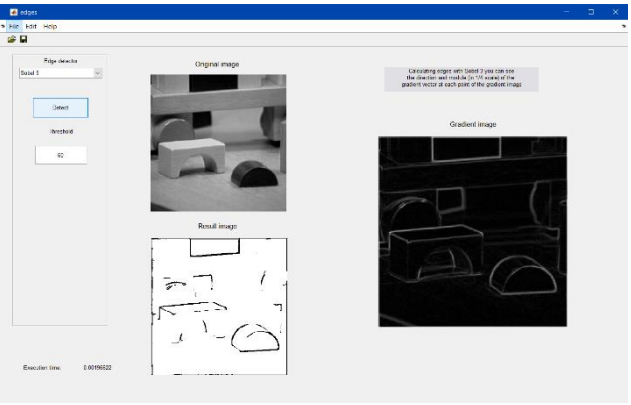

Figure 8

What is happening is there are less edges detected when we increase the thresholding for the only reason that those values do not reach the threshold.

4. Now use the DroG detector with the default parameters.

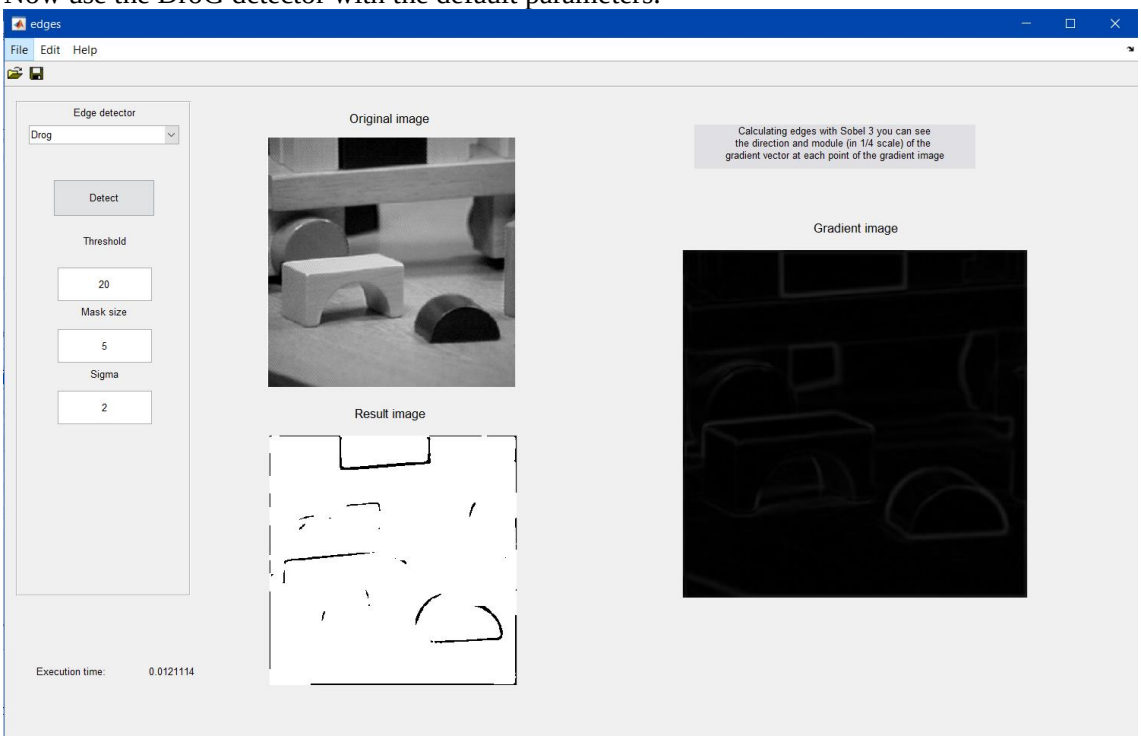

Figure 9

a. Why are detected less borders?
DroG is the combination of smoothing the image and detect the edges. It detects less because of the smoothing which make the border of the image more blur. This make it will more difficult of identifying the edges in the blurred zone.

b. Try with a different threshold and explain the result.

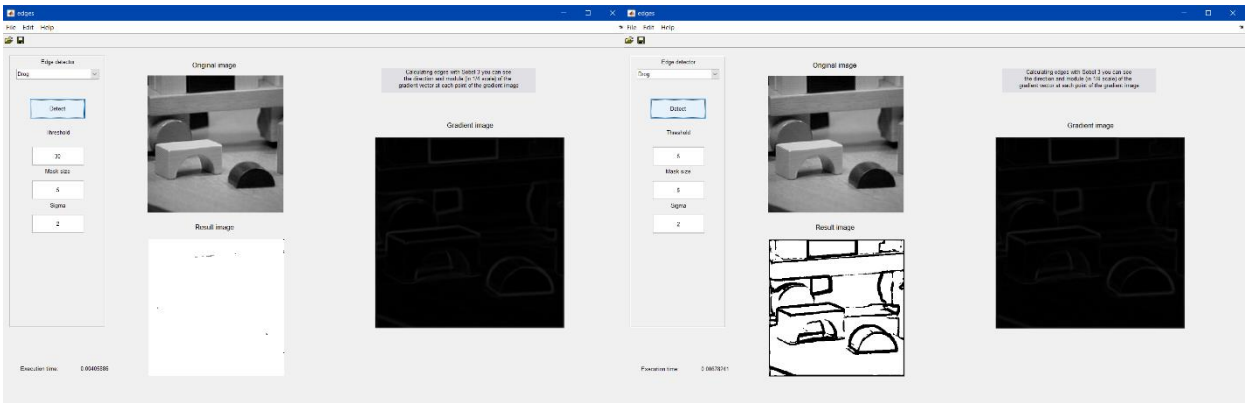

Figure 10

Figure 11

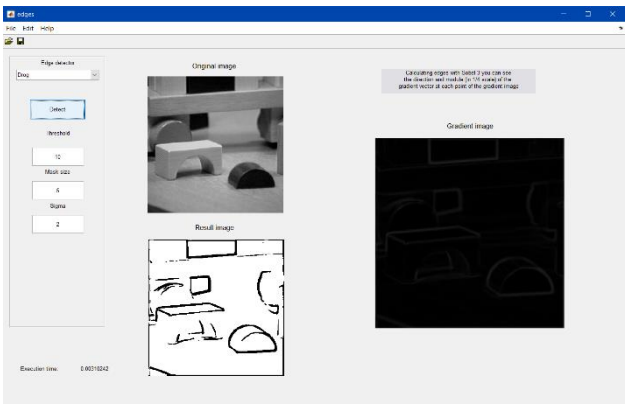

Figure 12

This have the same result as the exercise 3b where if we lower the threshold, there will be more edges detected, whereas if higher it, less edges will be detected.

c. With a threshold of 10, use different values for sigma (smoothing) and report which one provides the best output.

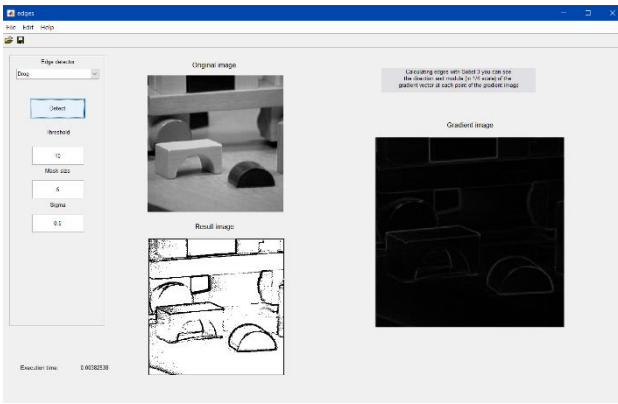

Figure 13

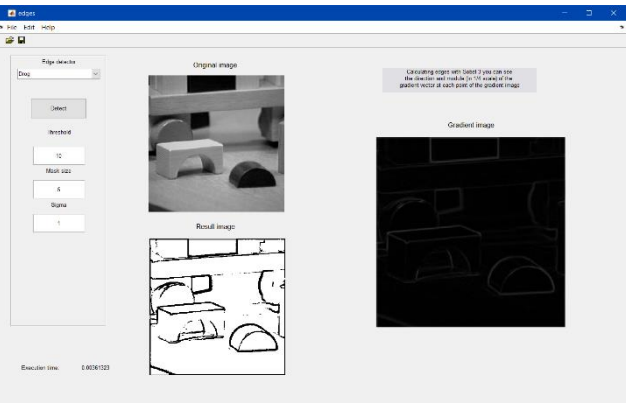

Figure 14

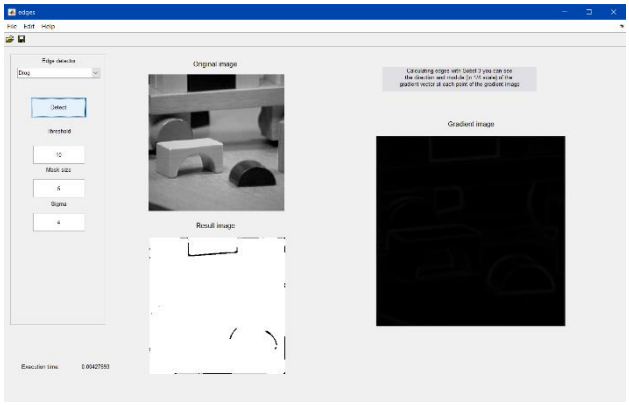

Figure 15

In this case, the gradient image change depending of the sigma. If it has a lower value, the gradients will be more noticeable, and thus more edges detected. Whereas, if it has a higher value, the gradients will be do blur that it will be more difficult to identify an edge, like happens in Figure 15.

5. Finally, let's try the Canny detector with the default values.

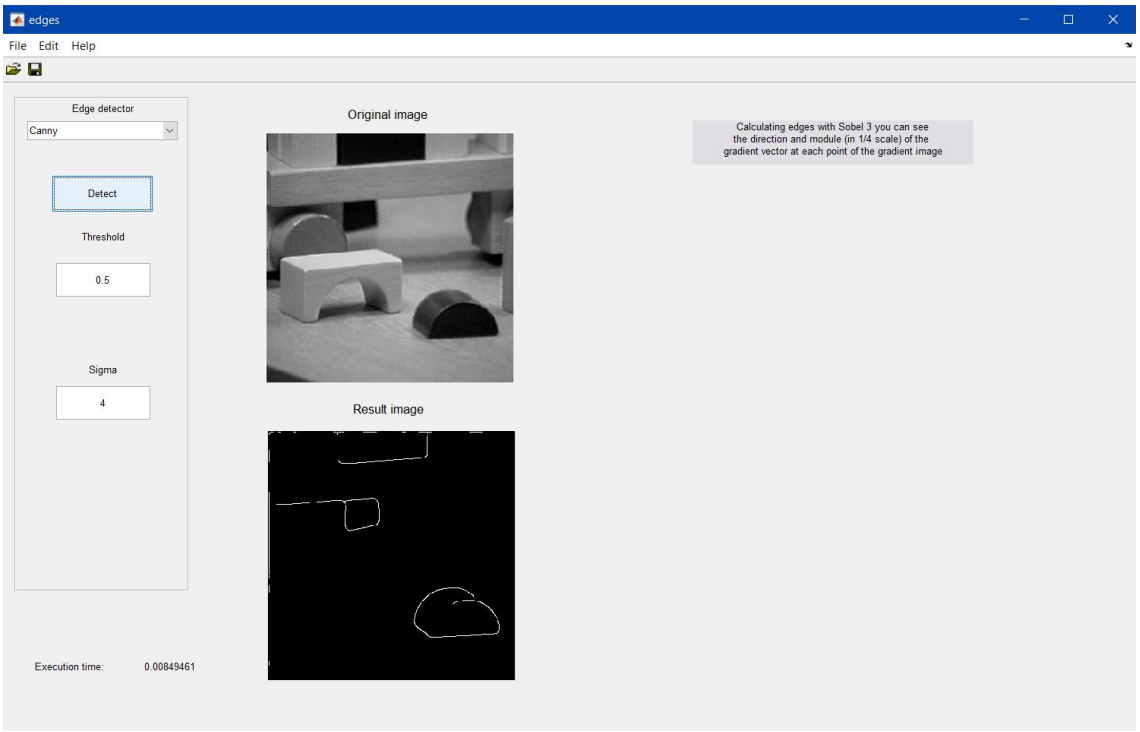

Figure 16

a. Again, we are getting a poor edges image. What should we do?
We must or lower the threshold or lower the sigma for identify more edges.

b. Matlab provides a function that detects edges in an image using different techniques, called `edge ()`. Take a look at its syntax with `help edge`. Since through the GUI we are specifying just a threshold and a value for sigma, which are the values for the *large threshold* and the *small threshold* used by Canny?
"If you specify a scalar for THRESH, this value is used for the high threshold and 0.4*THRESH is used for the low threshold."

c. Modify the parameters for getting the best possible edges image. Would be that parameters valid for any image?

Figure 17

No, because, depending of the image, we need other values to get a correct output on identifying edges.

6. In your opinion, which is the best edge detector in terms of lower detection error, localization error, and multiple response?
The best edge detector is the Canny, although it is computational expensive, with the right values identifies almost all the edges without false positives or true negatives.

EXERCISE 2b: Implementing edges detection

Concepts: Gradient operator: Sobel.

1. Design a Matlab function (not a script) that detects edges by means of the Sobel operator. The function must show a figure with 4 sub-plots (2x2), including:
   a. The initial image,
   b. an image with the vertical edges,
   c. an image with the horizontal edges,
   d. and a binary image with edges in green overlapping the initial one.

   Input parameters of the function: image name, threshold for the edges image. It has to return a matrix with the edge angle in each pixel, in the range [-π…π].

Important code

```
mask=1/4.*fspecial('sobel');
im2=conv2(double(im), mask, 'same');
im3=conv2(double(im), mask', 'same');
grad= atan(im3/im2);
im_modulo=abs(im2) + abs(im3);
pintar_verde; %Script which change the colour of the edges to
green and overlaps the original image
```

Result

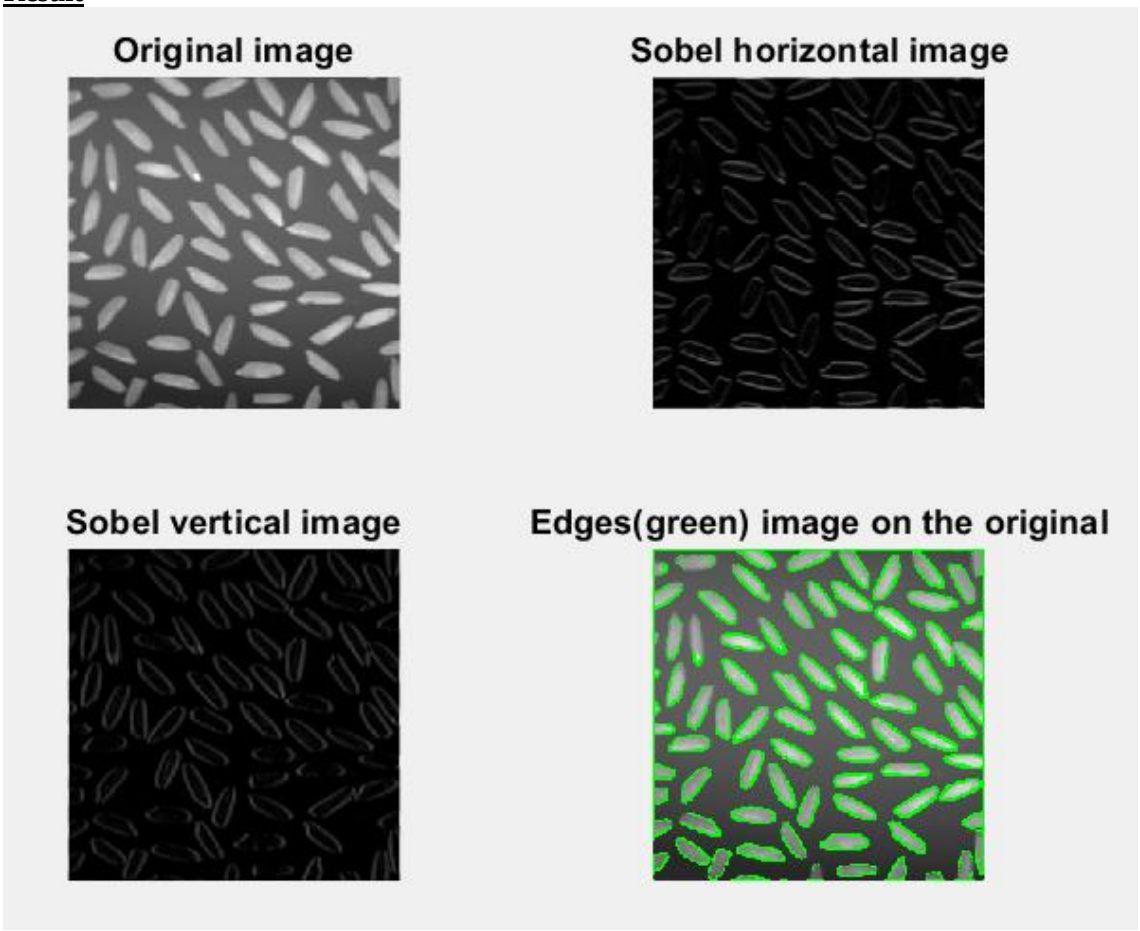

Figure 18. Edge detection with threshold=50