ESA-VIS Assignment 1

Assignments for the Vision course using OpenCV.

Edge detection

To do some edge detection on the provided image we used different methods. We first converted the image to grayscale. The we used three different filters, two Laplacian filters and one gradient filter.

For the Laplacian filters we used the following kernels:

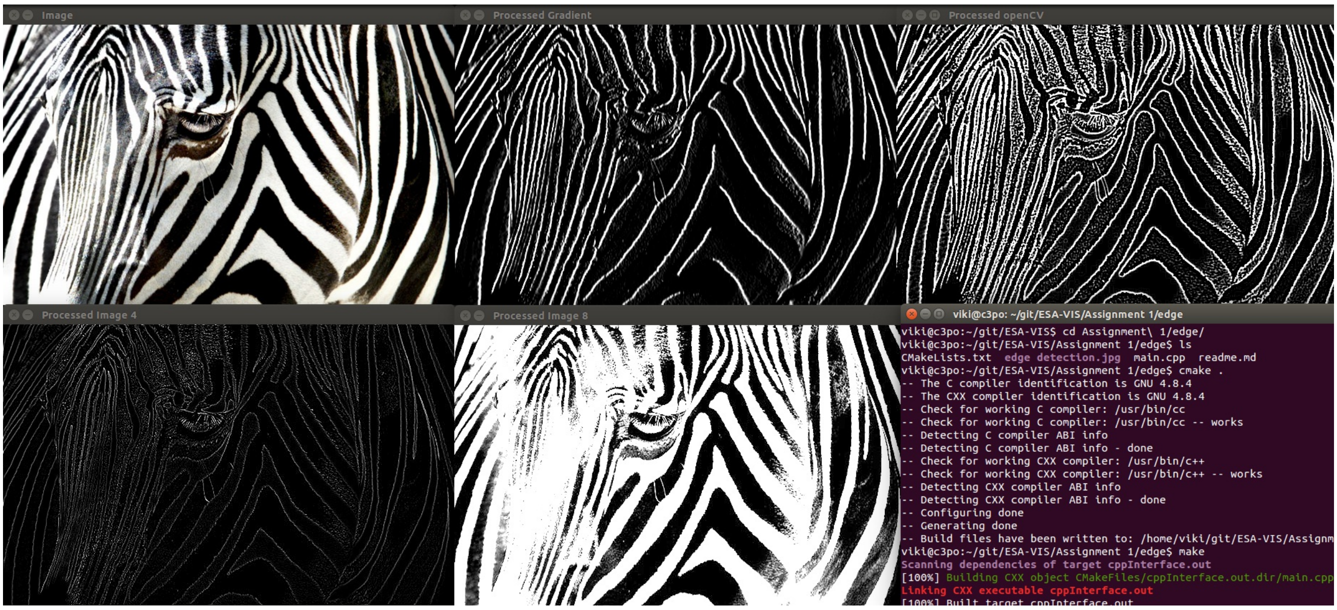
0	-1	0
-1	4	-1
0	-1	0

0	-1	0
-1	8	-1
0	-1	0

The gradient filter used the following kernel:

-1	0	1
-1	0	1
-1	0	1

As extra we processed the image with the OpenCV Laplacian filter, just to see the differences with the custom kernels we made and to do some experiments with the parameters.



```
viki@c3po:~/git/ESA-VIS$ cd Assignment\ 1/edge/
viki@c3po:~/git/ESA-VIS/Assignment 1/edge$ ls
CMakeLists.txt  edge detection.jpg  main.cpp  readme.md
viki@c3po:~/git/ESA-VIS/Assignment 1/edge$ cmake .
-- The C compiler identification is GNU 4.8.4
-- The CXX compiler identification is GNU 4.8.4
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/viki/git/ESA-VIS/Assignm
viki@c3po:~/git/ESA-VIS/Assignment 1/edge$ make
Scanning dependencies of target cppInterface.out
[100%] Building CXX object CMakeFiles/cppInterface.out.dir/main.cpp
Linking CXX executable cppInterface.out
[100%] Built target cppInterface.out
```

Figure 1: Results of the edge detect

Blobs

Preparations

The input image was in jpg format. Although it appeared to be black and white, when zoomed in, colored JPEG artifacts were visible. The edge was also fuzzy and not pixel-sharp. We checked the *brightness* of the pixel and when the average was brighter than half, it was determined to be white. In this way we created a purely black and white image to work with.

Algorithm

For the blob assignment, we first tried doing it per line pixel and working line by line. This proved to not work very well, as it was impossible to determine the colors of the pixels starting on another line which are offset more than one pixel from the previous lines pixel.

We then tried to color the first found pixels and the pixels around that, with a recursive function moving to the next pixel each time. This proved to work well, with each blob appearing separately colored.

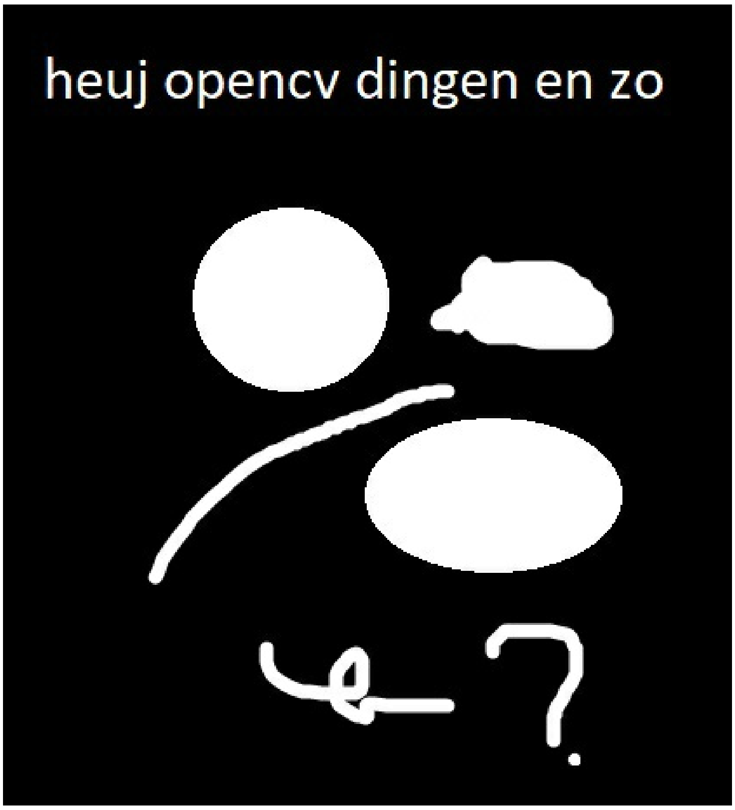


Figure 2: Input image with more blobs

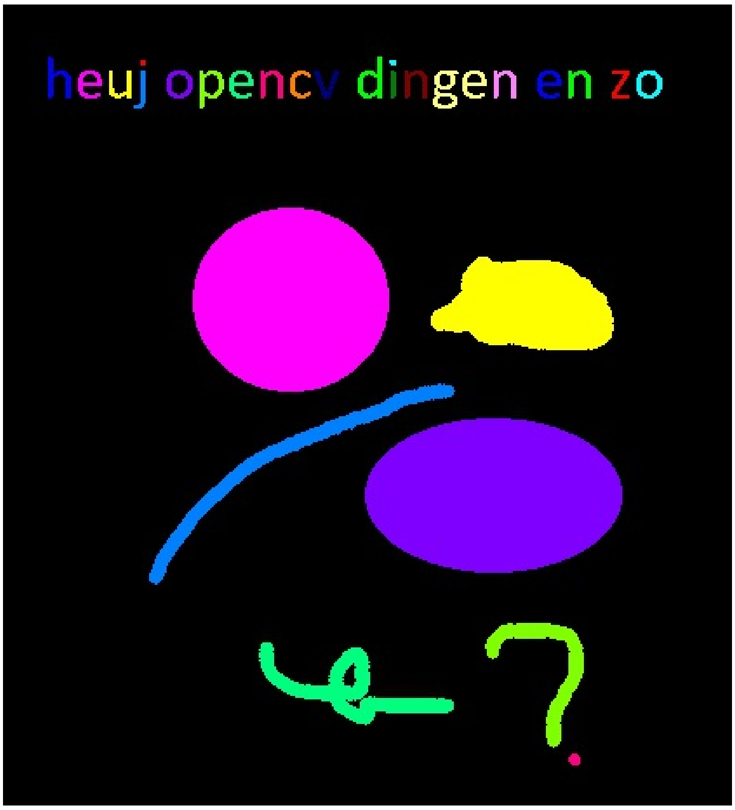


Figure 3: Results of the blob program