Lab 1, Introduction to OpenCV

João Santos, *MRSI, 76912*

***Index Terms*—OpenCV, Computer Vision, MRSI, UA, DETI, LaTeX.**

I. Introduction

THIS report is intended to be used alongside the Python3 code developed for this Lab.

The Lab #02 is a introduction class to an IDE (in the case of the author, Visual Studio Code), OpenCV and Python 3.

This report was written using LaTeX .

II. Exercises

Lets analyse the resolution of the proposed exercises.

A. Ex. 1.3: Direct pixel manipulation

For this exercise, the goal was to, put simply, create our own threshold function. For this, we would iterate over all pixels of a given image (passed as the first argument for the script) and if the value was bellow a given value (the second argument of the script), that pixels is set to 0.

Bellow, we show Fig. 1 which is the original image, followed by two outputs, Fig. 2 and Fig. 3 which are the original image thresholded, respectively, at 128 and 200.

Fig. 1. Original image.

Fig. 2. Original image thresholded at 128.

Fig. 3. Original image thresholded at 200.

B. Ex. 1.4: Mathematical operation - image subtraction

To practice our mathematical operation on images skills, we were asked to subtract two images.

Their base was the same, but have been saved to disk with different compression algorithms (.JPG vs .BMP). Bellow, by subtracting Fig. 4 to itself but compressed in different ways, produce different results. It is also noticeable the OpenCV's *cv2.subtract()* works in a different way of NumPy's *numpy.subtract()*.

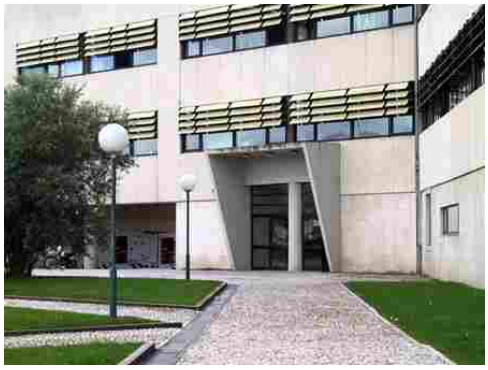

Fig. 4. Original image.

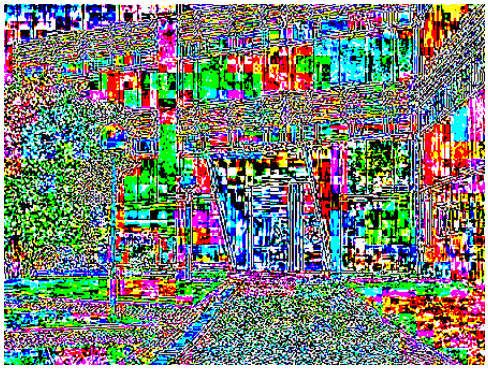

Fig. 5. Subtraction result using *numpy.subtract()* method.

Fig. 6. Subtraction result using *cv2.subtract()* method.

C. Ex. 1.5: Interaction - selecting a pixel and drawing a circle

OpenCV also provides a set of methods that provide GUI functionality. One of those provides a easy way to draw circles in images. Also, when creating a new Window, a callback can be associated in order to perform a given task and a defined event occurs.

In this case, when mouse left button click event occurs, a circle must de drawn on the image, with it's center where the mouse click occurred.

The code execution result can be seen on Fig. 7.

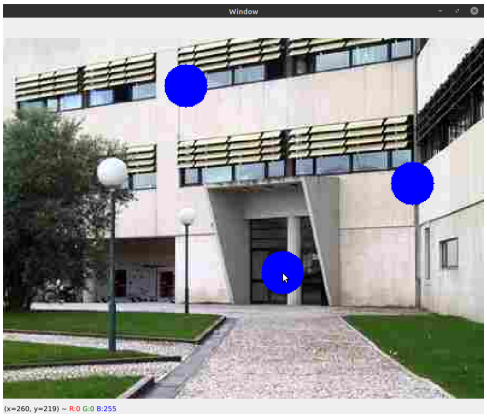

Fig. 7. OpenCV Window with a callback to draw circles on top of the image.

D. Ex. 1.6: Conversion between color spaces

Many operations on images can only be performed if it is a gray scale one.

If it is not, OpenCV provided the *cv2.cvtColor()* method to convert between different color spaces.

Fig. 8 displays the same image as on Fig. 4 converted to gray scale.

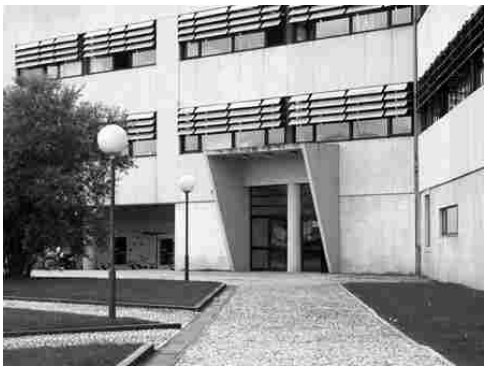

Fig. 8. Gray scale image.