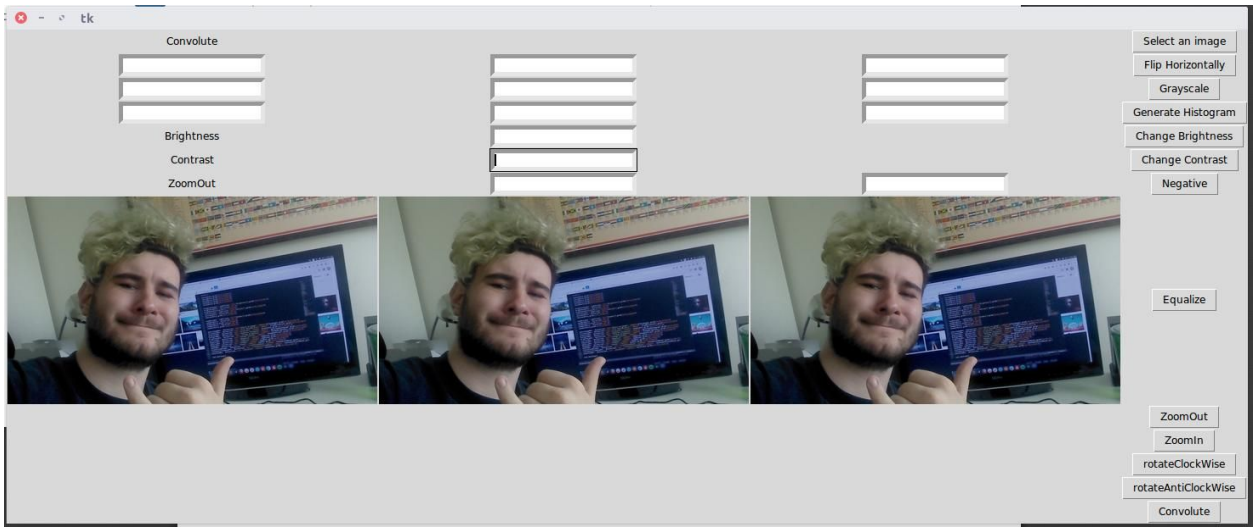
Rodrigo Ronconi Richter, 217445
Fundamentals of Image Processing, Prof. Manuel M. Oliveira
Implementation Assignment #2

Part I

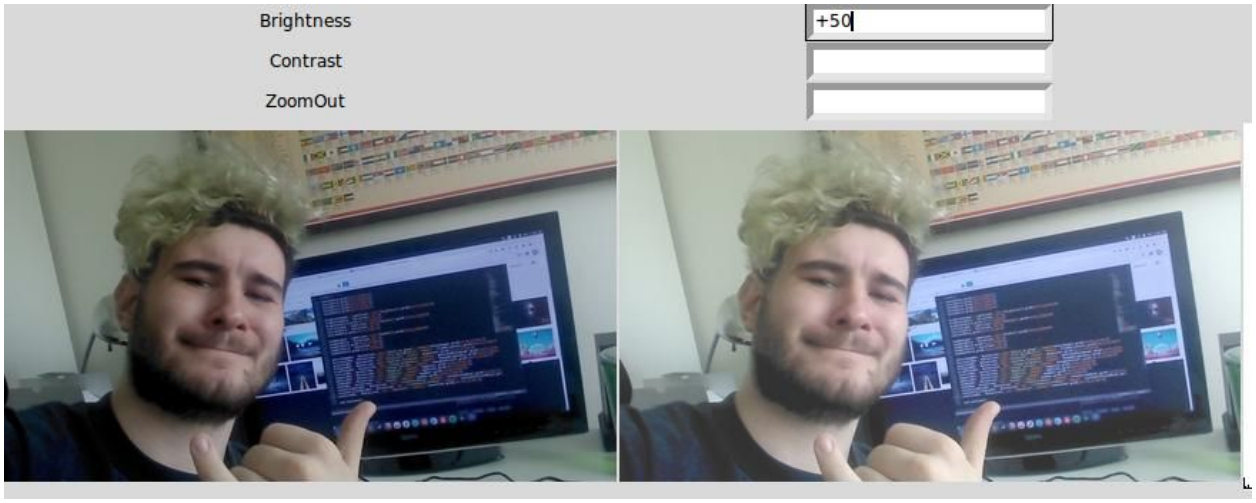
The original program was extended to include a whole new GUI system: Tkinter. This way, my program uses OpenCV to handle loading of image files, and now uses Tkinter to display them. Also, it now can read text input and has buttons. It has 3 images, the original, the one after modifications, and a new one to display histograms.

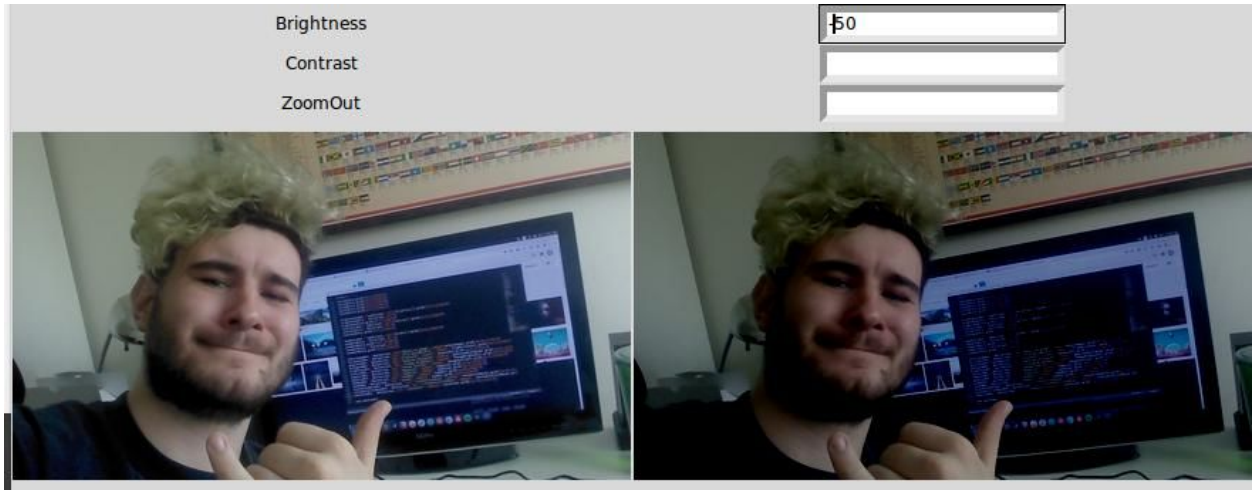


1) Generate Histogram

The histogram is normalized to the biggest frequency of a luminance value in the image. This way, it will be more likely to have a more appropriate representation.

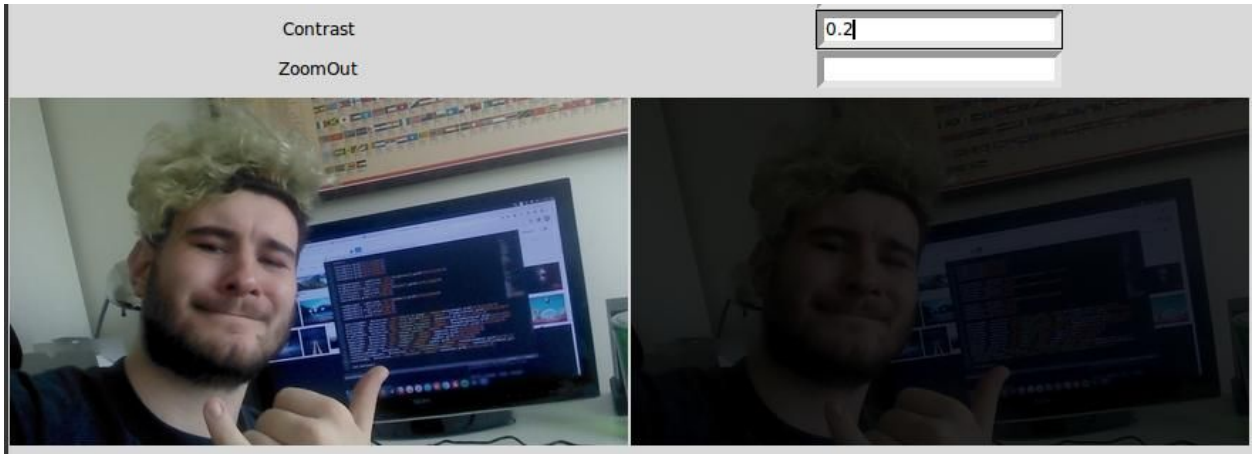
2) Change Brightness





The algorithm executes on each channel individually. The GUI reads the value and converts to an integer, negative or positive.

3) Change Contrast



4) Negative

5) Equalize

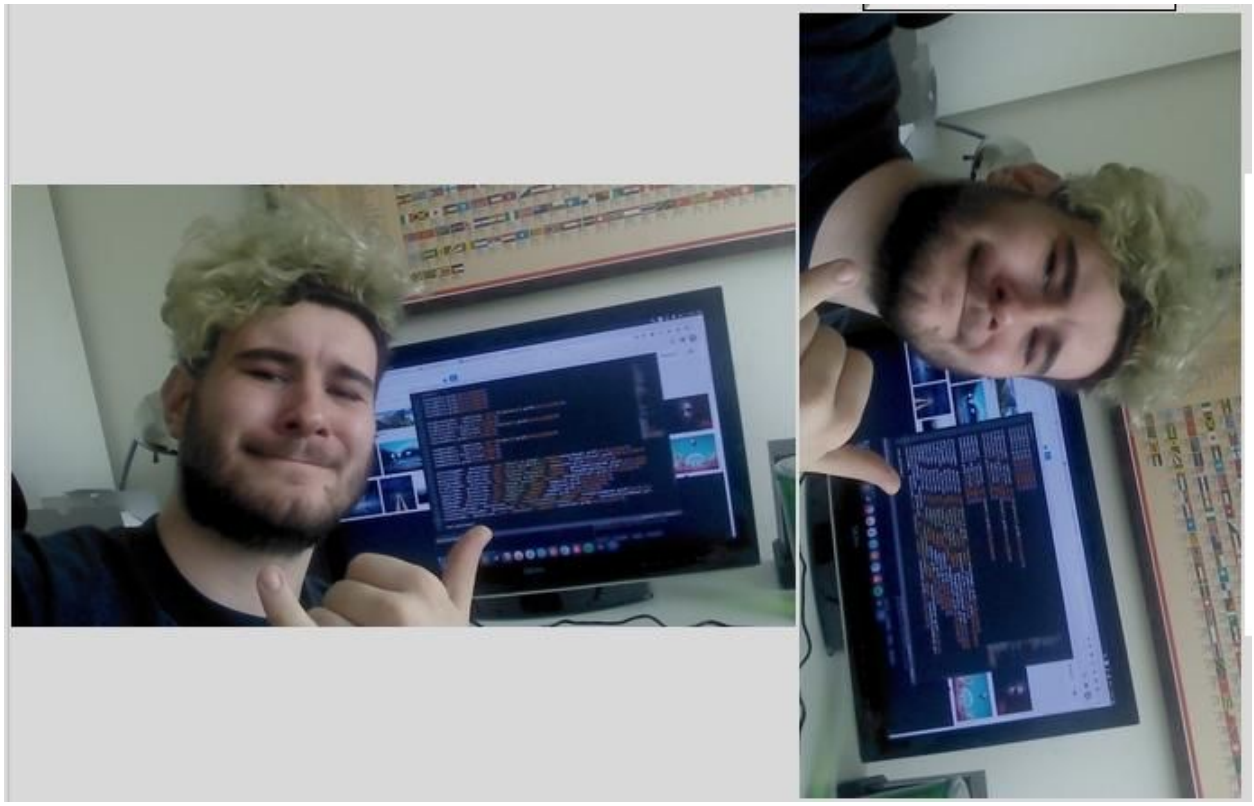
First, it is generated a cumulative histogram, from the starting image. The algorithm then uses it to generate the equalized image. The difference between the histogram of the equalized image and the original histogram are easily perceived.

Part II

6) ZoomOut

7) ZoomIn

8) Rotate

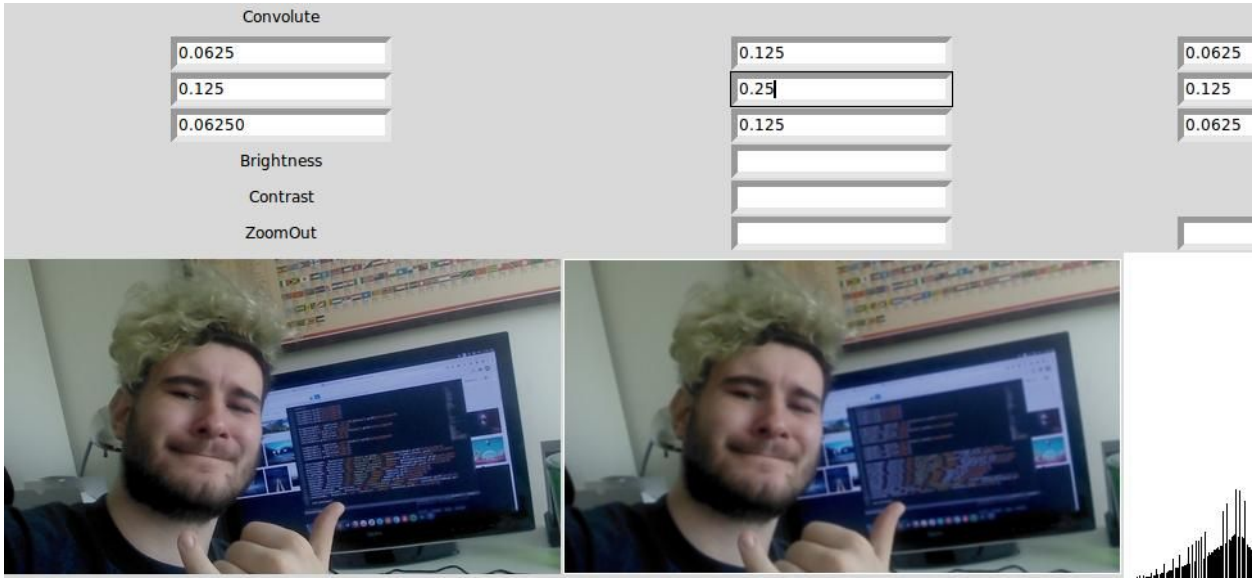


The program can rotate clock and anticlock-wise, and it can apply it multiple times.

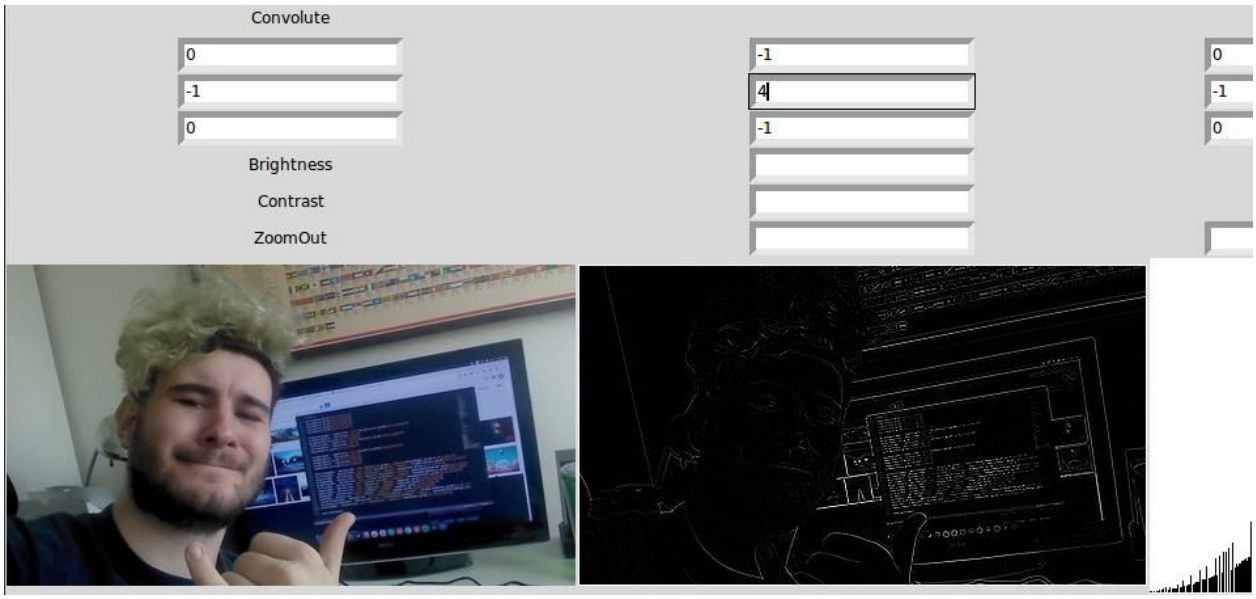
9) Convolute

The convolution filters were made to work on colored images as well. Some noise can be seen on the results of the convolution, and that is due to the fact that the image is a picture from a not so good smartphone camera.

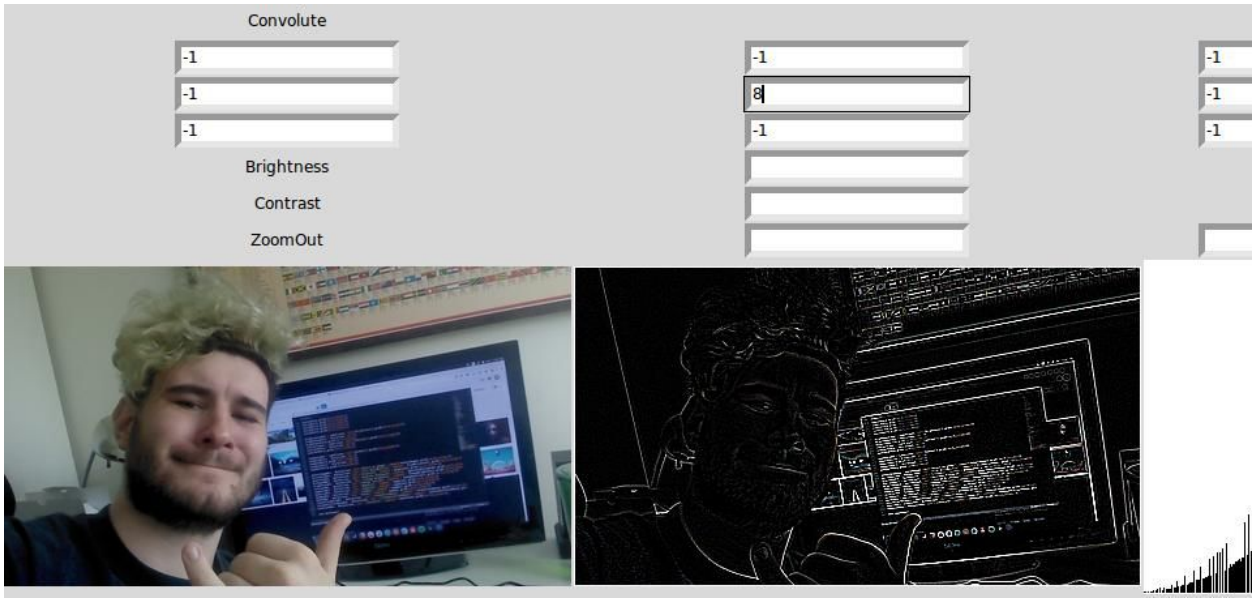
I) Gaussian



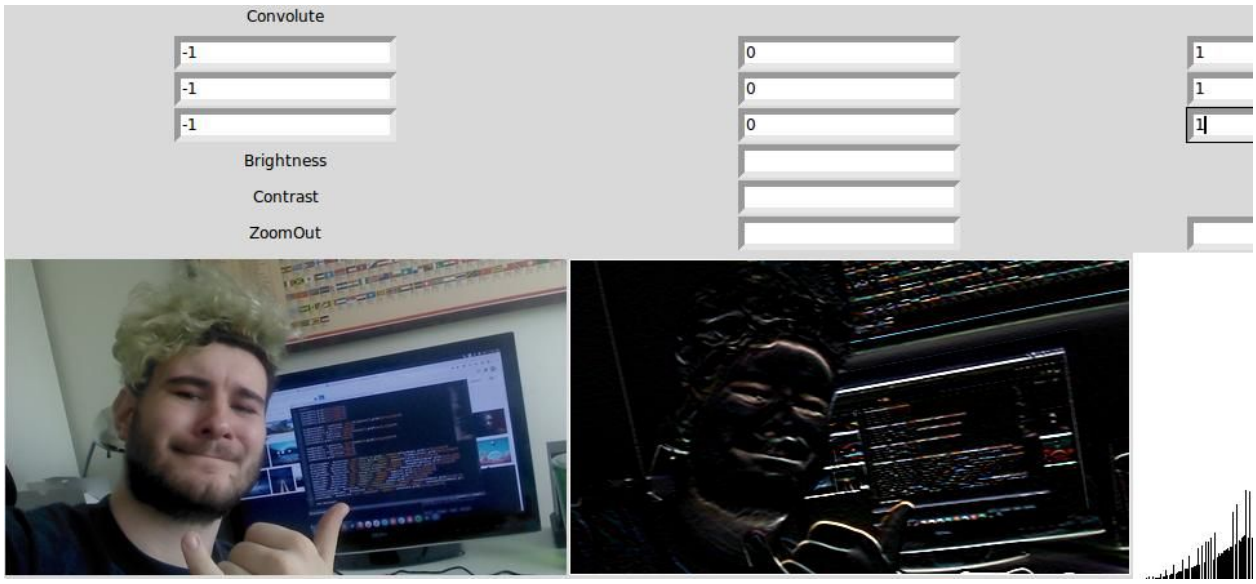
II) Laplacian



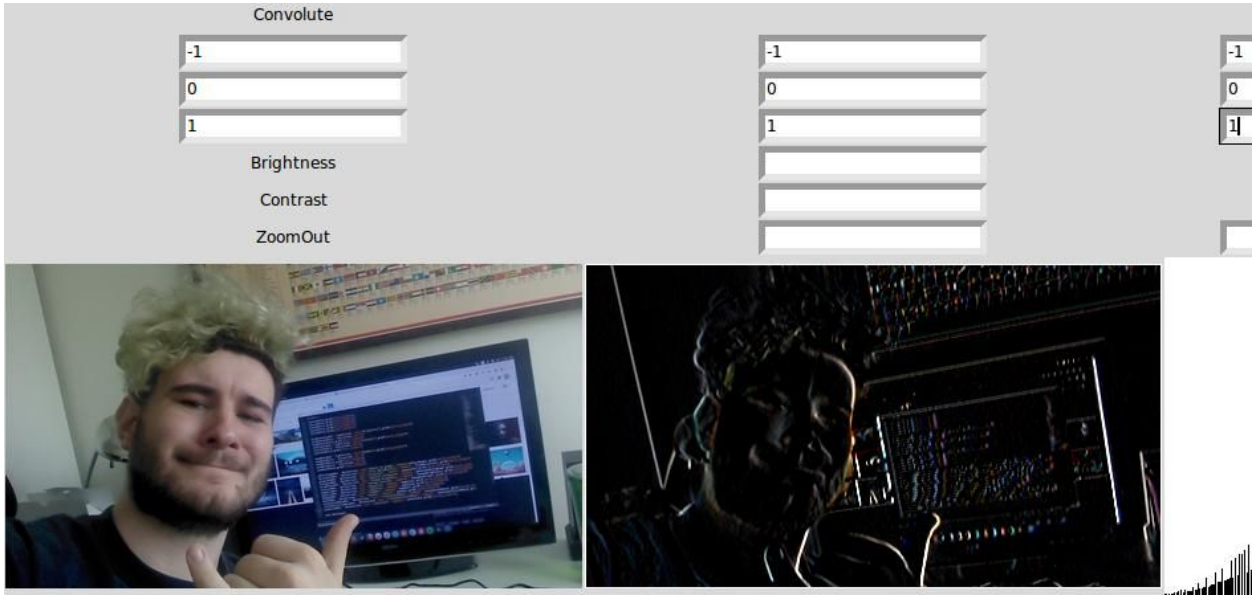
III) Generic High-Pass



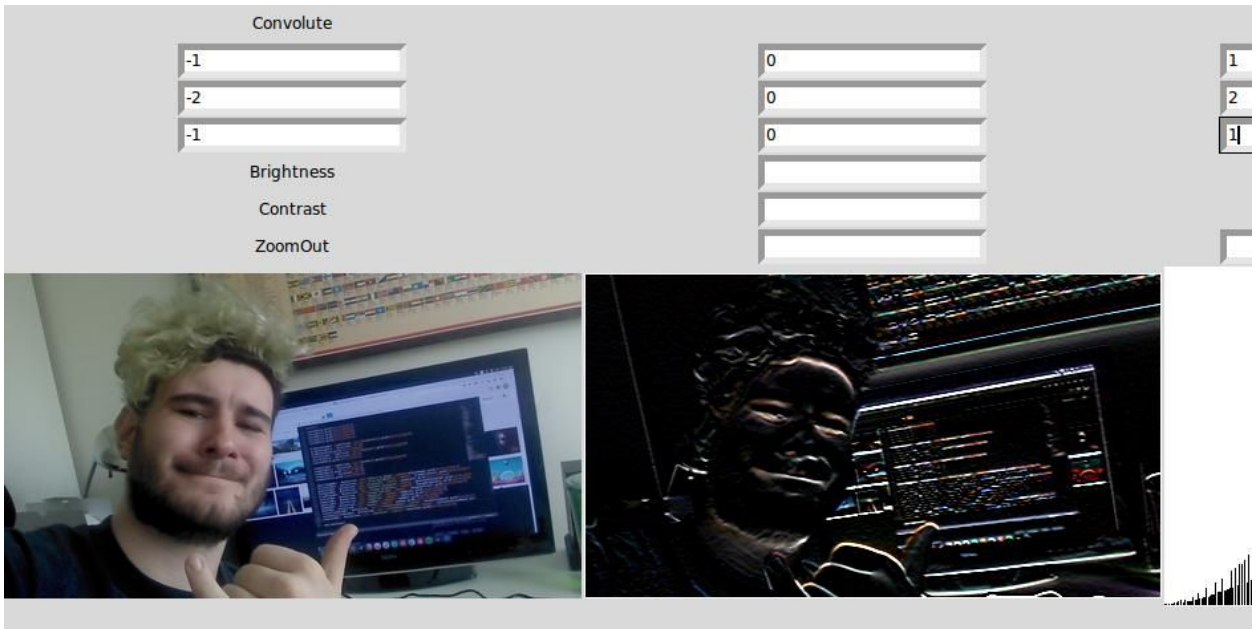
IV) Prewitt Hx



V) Prewitt Hy Hx



VI) Sobel Hx



VII) Sobel Hy