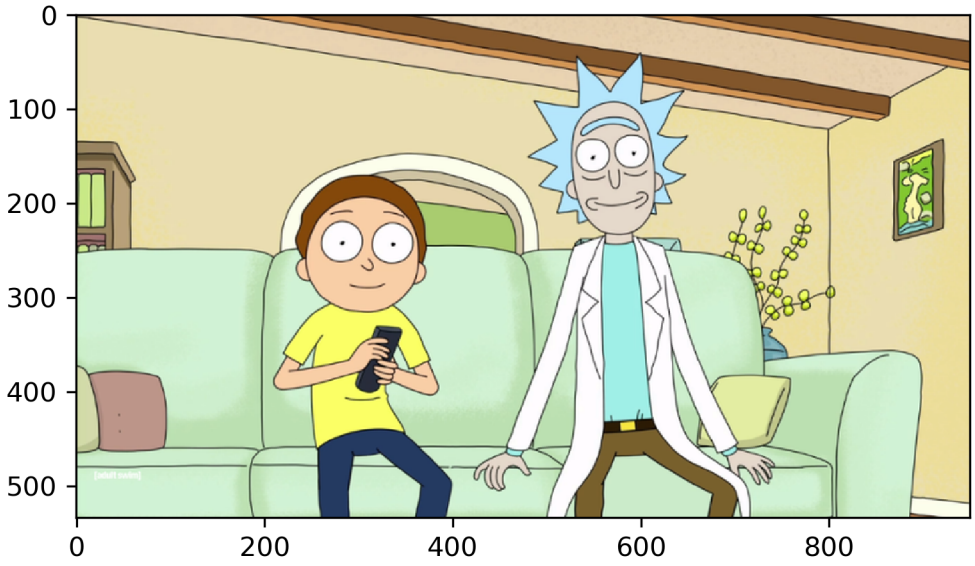
Edge detection with sobel filter

Given a RGB .pgn image, I am gonna create a grayscale image emphasizing the edges in the image.

1. Loading the RGB image:

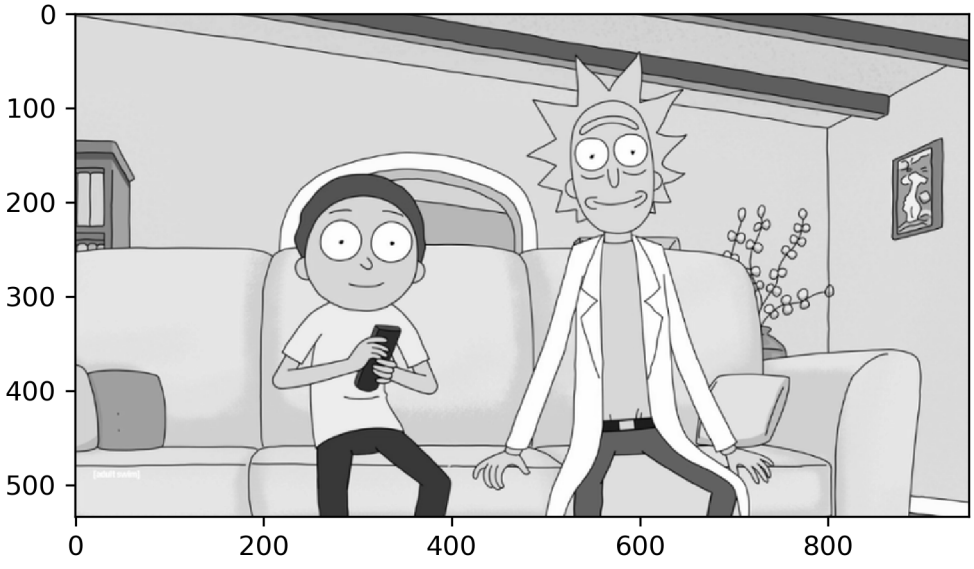
Loading the image from url, and putting it in a numpy array with shape = (534, 950, 3)



2. Converting the image to grayscale:

Each pixel get the greyscale value of 0.299*Red + 0.587*Green + 0.114*Blue.

Converted from shape (534, 950, 3) to (534, 950)



3. Padding with zeros:

Adding a padding of zeros around the image, to be able to apply the sobel filter to the pixels on the edges of the picture.

From shape (534, 950) to (536, 952)

4. Defining sobel kernels:

sobel kernel (x-direction) = [[-1, 0, 1],
[-2, 0, 2],
[-1, 0, 1]]

sobel kernel (y-direction) = [[-1, -2, -1],
[0, 0, 0],
[1, 2, 1]]

4. Applying sobel kernels:

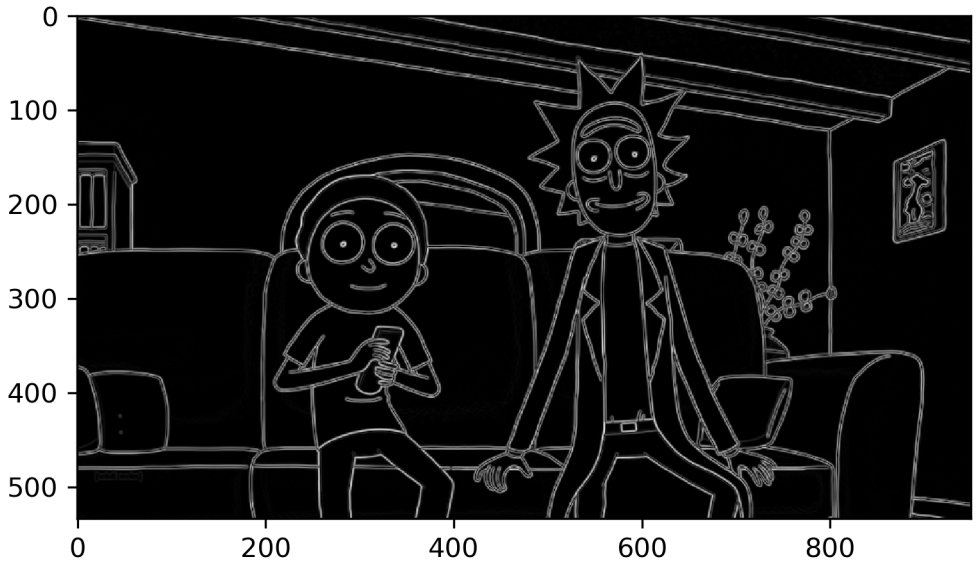
In this stage I convolve the kernels on the padded image.

For each pixel inside the padding find the values from the two kernels.

result[i][j]=(sobel[0][0]*image[i-1][j-1]+sobel[0][1]*image[i-1][j]…+sobel[2, 2]*image[i+1][j+1])

And then combine the absolute magnitude of the results in x- and y-direction: result = sqrt(result-x^2+result-y^2), and set the pixel value to this value.

This way one would find the difference in intensity of the pixel evaluated and the neighbour pixels. This will result in an approximation of the gradient of the image intensity. High intensity pixels of the result image represent edges in the original picture.



5. Highlighting the edges:

Thresholding the image with a value of 70, setting all pixels with intensity over 70 to 255 and all from 70 and below to 0. This way the weakest edge pixels get set to 0, and the stronger edges gets highlighted.