Visual Recognition

Udith Sai .M
IMT2018081

Assignment-1

February 08, 2021

Introduction

This assignment is about getting started with OpenCV and some algorithms we use on images to get desired output like edge detection,smoothing,blurring. OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products. We also learnt some basic algorithm functions available in OpenCV like Gaussian Blur,canny edge detection, convolution, Hough Lines, Contour figures. There are three parts in this assignment. One is Finding number of books and the other is Finding face pixels in an image.

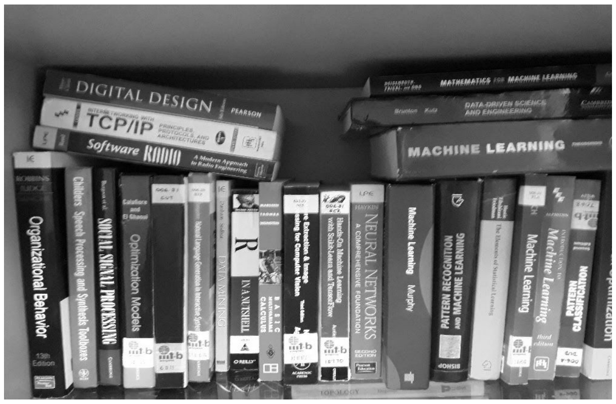
Parts

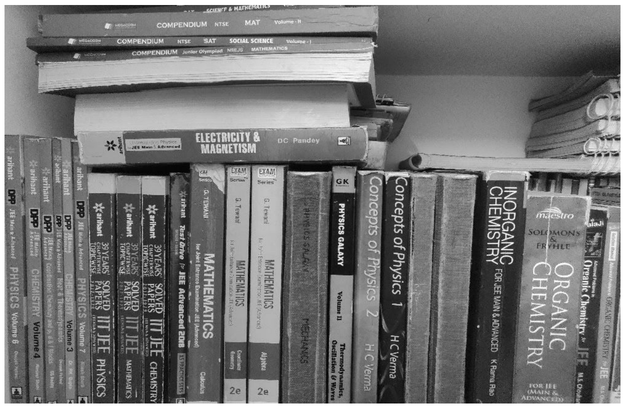
1. **Book Counting:** In this part an image of a shelf containing books is given and when we run the program giving the name of the image in the code book.py, it shows the approximate number of books present on the shelf.

2. **Pixels on your face:** In this part an image of a person is given and when we give the name of image as input in code face.py, it only takes the area of face and makes the other space black.

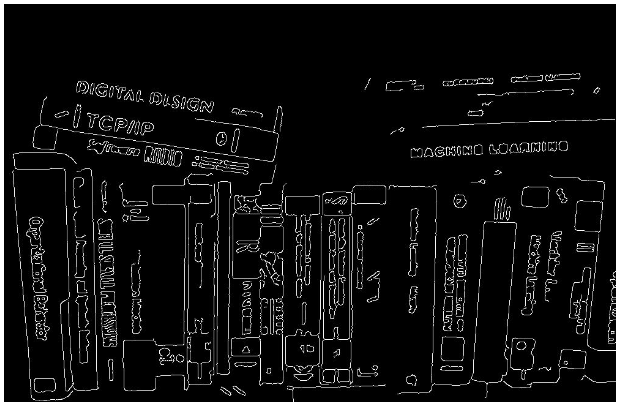
Book Counting

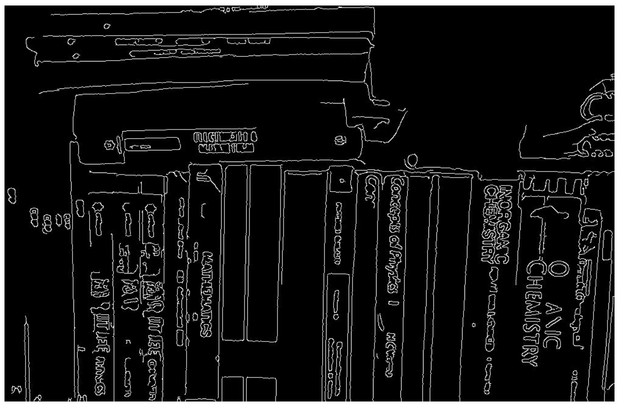
To find the number of books on a shelf in the image we need to do a lot of pre-processing on the image before we apply the actual function. So initially we resized and converted the BGR colour image to grayscale image so that we can apply any functions easily. And then to remove the noice like text on the books, we did smoothing and then blurring on image. For smoothing we chose kernel of 5x5 and blurred using GaussianBlur.



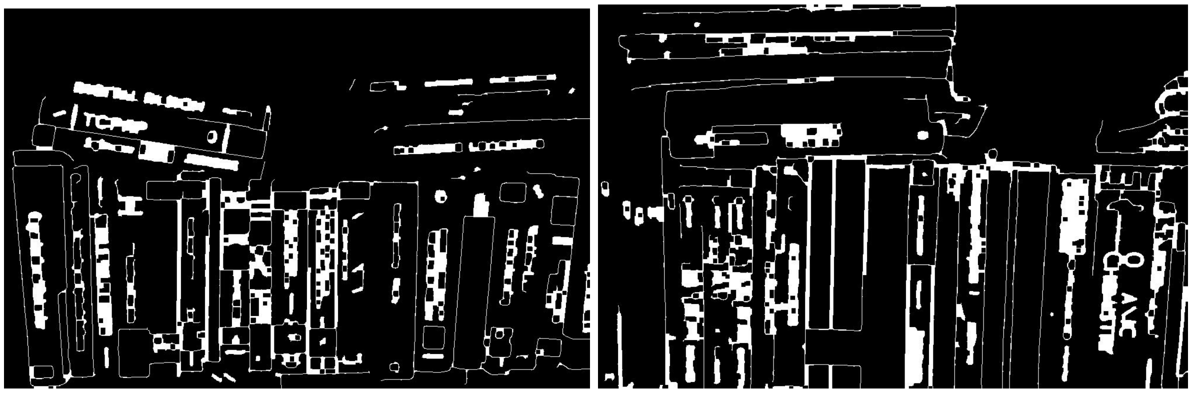


Then using this smooth and blurred image, we did canny edge detection so that we can find edges in the image. We took minimum length as 50, maximum length as 150 and apertureSize as 3.

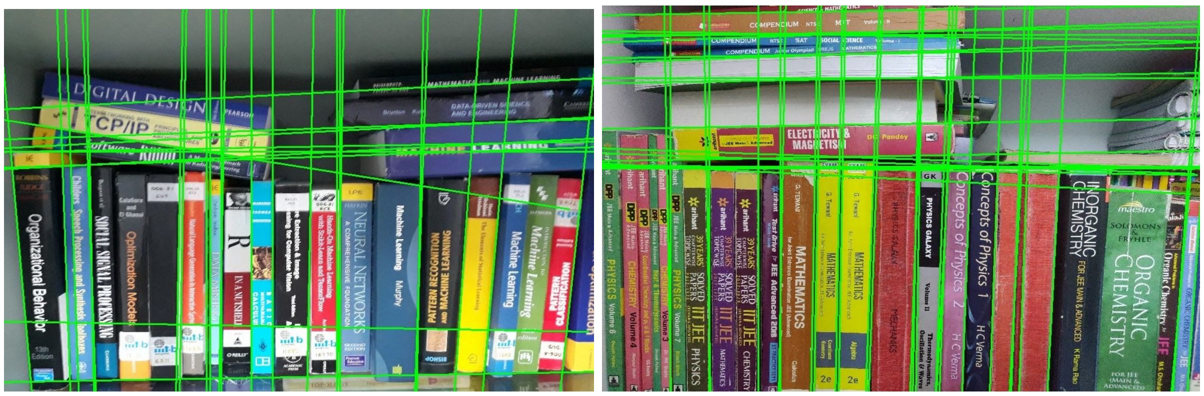




Then to fill the small gaps in the image after canny, we used close function to fill the gaps but later we found that it is not useful in this part.



Then we used Hough lines on canny edge detected image as books edes are straight lines, if we find number of lines available in image can be proportional to numer of books. So, we applied hough lines function with parameters as rho and theta as 1 and np.pi/180 and threshold as 130.



Then using angle of line, we divided into vertical and horizontal lines, the sum of (vertical line-2) and (horizontal lines -2).

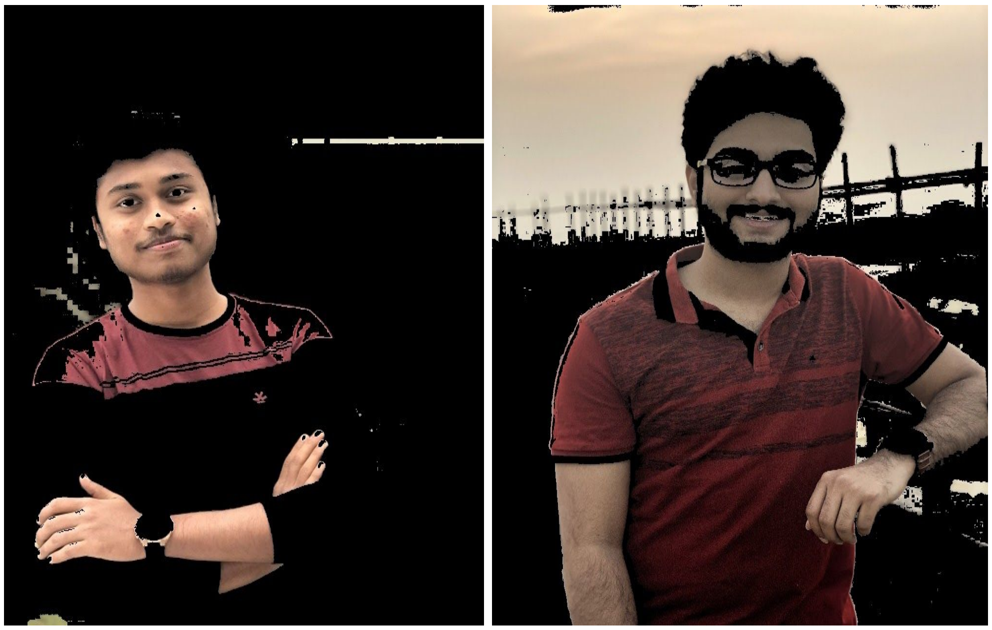
While doing this task, we did different methods of blurring to find better result like Blur, medianBlur, GaussianBlur and at last we found that using gaussianBlur we got better result.

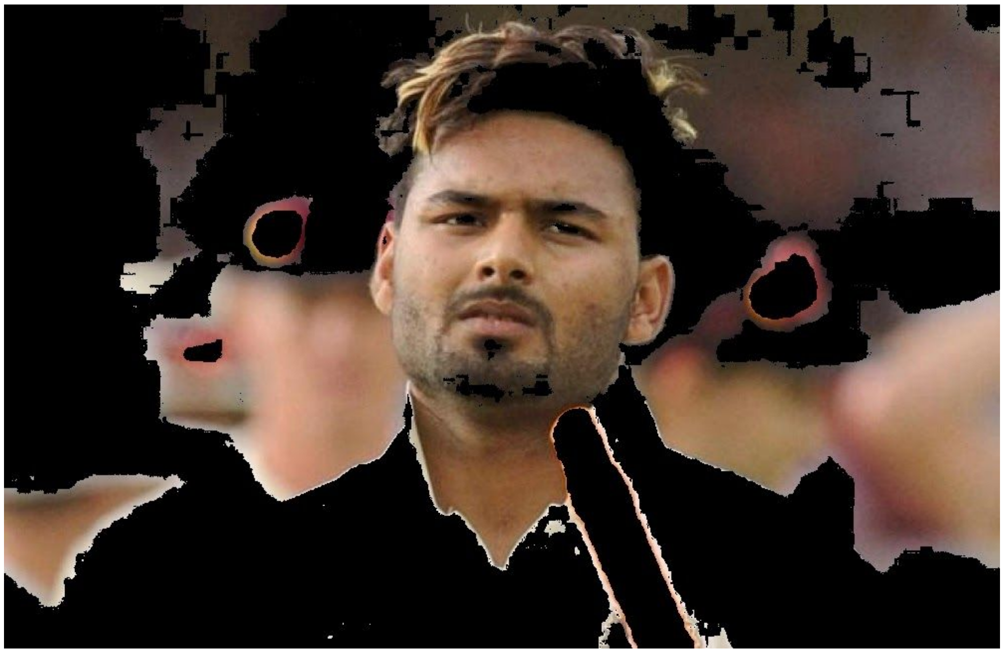
And after that I did hyperparameter tuning, for hough lines threshold, and kernels. For hough lines, initially 150 was threshold then number of lines were less than present and when threshold is 100, there are more lines than actual so at last 130 gave better result and size of kernel was 5x5. For Canny, min and max length are also tuned such that 50 and 150 gave better results.   Aperture size =3 only gave results.

When we run the code, images start to appear one by one, when one image is present, if you want to save, click ‘s’ or escape to go to the next image. And in the console, at last it prints the number of books. The mode is not 99% accurate there is some error so, actual number of books = showed number of books +- 6.

Pixels on your Face

To find the face pixels in an image, we first need to know the range of skin colours so that we can only segregate the scenery from the face. The range of skin colour is [0,133,77] to [235,173,127] in YCR colors. So, initially we converted image from BGR to YCR colors. Then using the range of colours we got skin area using inrange() function and then using bitwise_and() which detect the skin from the image on a new image passing three arguments src1, src2 and skinArea.





We even tried using contour figures but couldn’t get a proper output. Here, by giving more clear skin colours makes the code more efficient.When we run the code, images start to appear one by one, when one image is present, if you want to save, click ‘s’ or escape to go to the next image.

Conclusion

After completing this assignment, we got an idea about using the OpenCV library in python and using the library reading an image and applying different functions like grayscale, blurring, edge detection..etc. And we even learn many new things like what are the pre-processing steps to be done based on required tasks and also learn many functions like canny edge detection, hough lines..etc. At last but not least, i got to know that HyperParameter tuning is the important task in creating a model.

References

1. Books counting
    - https://yasoob.me/2015/03/11/a-guide-to-finding-books-in-images-using-python-and-opencv/
    - https://stackoverflow.com/questions/21487658/recognition-and-counting-of-books-from-side-using-opencv/21495568
    - https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_houghlines/py_houghlines.html
2. Pixels on your Face
    - https://www.codespeedy.com/skin-detection-using-opencv-in-python/
    - https://www.thepythoncode.com/article/kmeans-for-image-segmentation-opencv-python

Acknowledgements

We would like to thank our Professor Dinesh Babu for giving us a great headstart in this world of Visual Recognition.