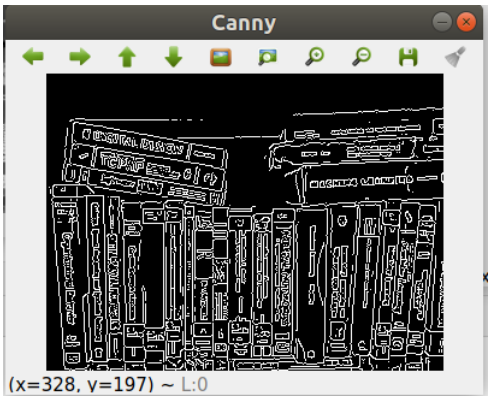
Assignment 1

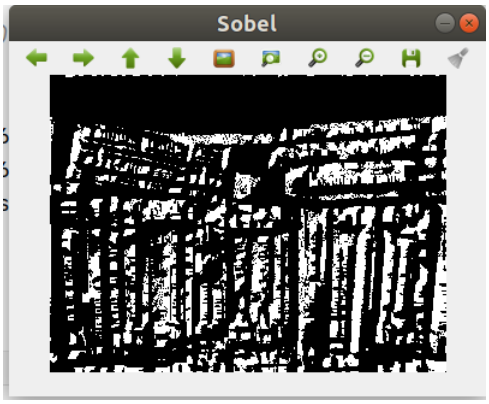
Shourabh Payal (MT2020054)

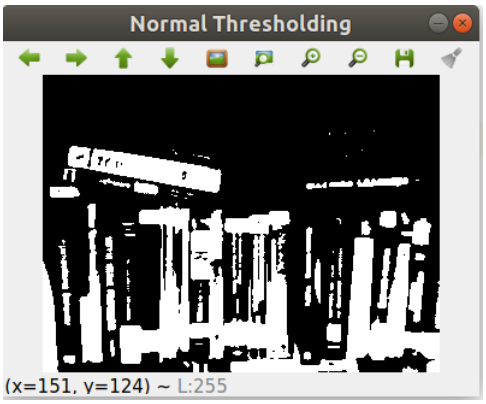
1. ***Book Counting:*** *Write a rule based algorithm to count the number of books in my shelf*

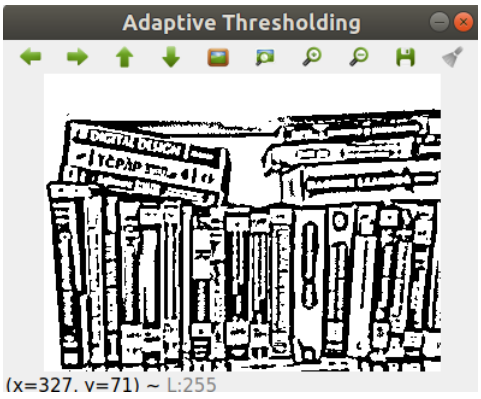
   (a) **Initial thoughts**

   First thoughts were to apply different transformations to the book images and look for a certain pattern which i can exploit to count the books.
   Below are some prominent sample transformations of images which guided me to the ideas discussed in subsequent sections.









   (b) **Hypothesis 1 & 2**

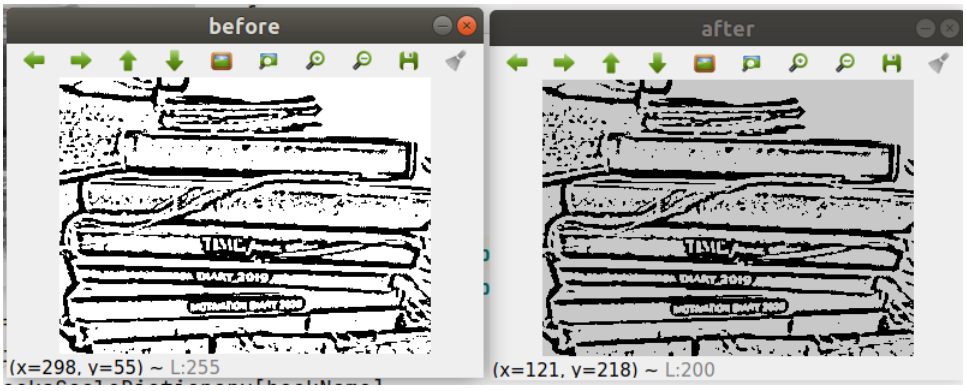
   Based on the above pattern I formulated a hypothesis which focused on the idea of doing a flood fill or dfs using the white or black pixels obtained from pre processing the images as can be seen above.

The algorithm will try to identify the regions of white area via standard dfs algorithm. We count the number of regions found and report them as book count. This is quite naive algorithm. To improve it we introduce the concept of area of the region to eliminate unnecessary noise being counted as book. The regions satisfying a threshold will be reported as books.

How to decide the threshold? At first thought I wanted to go with mean area but there were very large number of small noisy region which were pulling the threshold low (Standard pre processing steps are already applied like blurring etc to reduce the effect of noise). I decided to use median of the areas to be more accurate.

(c) **Experiments & Results**

This hypothesis reported book counts to be 60 before median area filtering and 30 after. While the actual number of books is around 23. (These best results were obtained for adaptive thresholding out of the above 4 samples)
On trying this hypothesis on other images results were close like the stock image but not quite accurate.



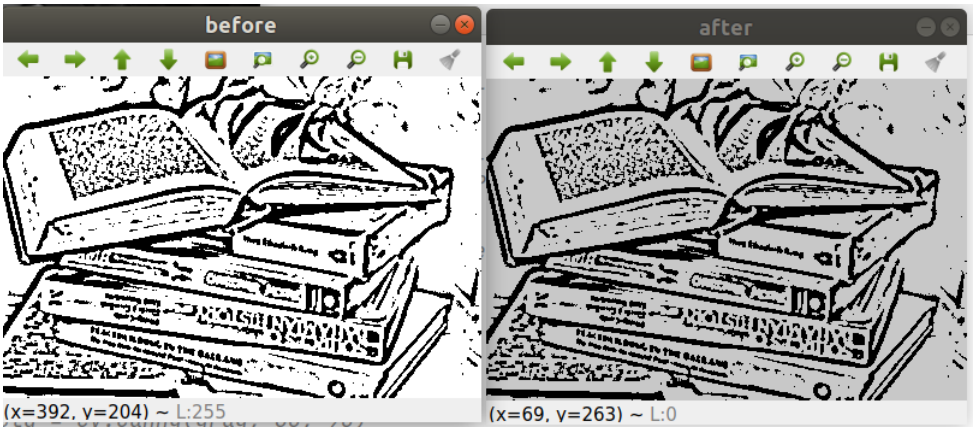
For example this image reported the book count to be 10 while the actual book count is 6.
Again adaptive thresholding was the best approximation. Upon trying it on more images adaptive thresholding consistently outperformed other preprocessing methods. The results based on this hypothesis were good but we want to improve more and build in similar directions.

(d) **Hypothesis 3**

Carrying the earlier experience and pattern we can notice that the books if arranged in the shelf can be viewed horizontally or vertically.
We can notice that either height or width of the book is significantly more than the other dimension depending on the orientation in which the book is kept. We can perhaps exploit this to obtain more accurate results.

We modify our dfs to report the maximum width and height of the region under consideration. We set some small cutoff threshold for the height and width of the books as well depending on the scale.

(e) **Experiments & Results**



The result obtained were more accurate than earlier hypothesis. In the above image we have 5 books in reality and number of books reported were 7. Similar results were observed for other book images. Once again the adaptive thresholding is destroying the competition consistently.

It was also noticed that sometimes the area bounded by the black region was not fully covered up for a book and as a result the number of books came out to be less than actual. While other times they came out to be slightly more when image was able to close edges accurately due to image quality.

(f) **Hypothesis 4**

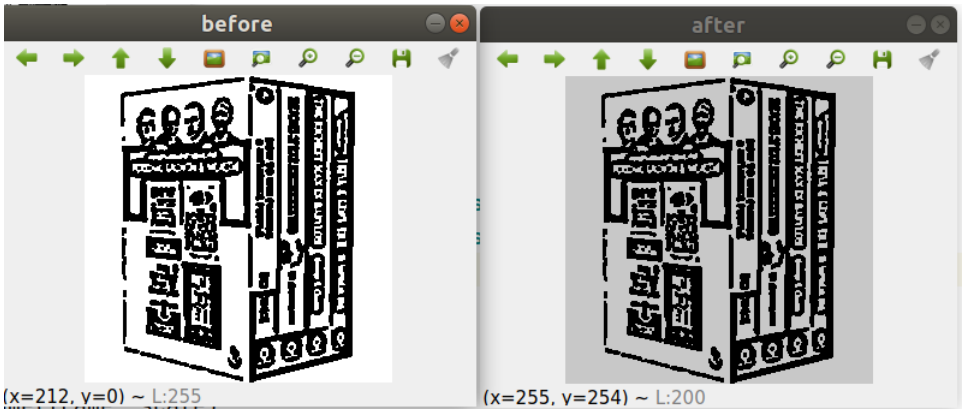
To account for the irregular fluctuations in number of books, sometimes more sometimes less I decided to apply a combination of a stronger blurring plus eroding function from opencv. This will compensate for the open edges and will also take care of to open up stronger edges. Below is a sample from opencv documentation on how eroding affects the image.

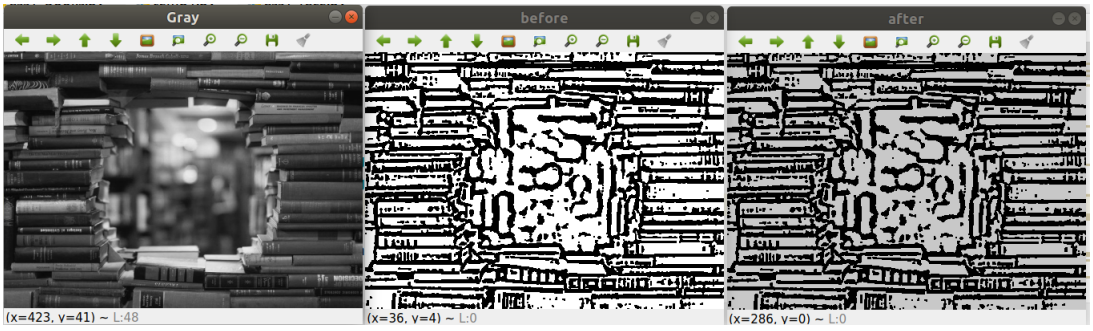
(a) Before

(b) After applying erode function

Tried other operations as well like morphologyEx from opencv with MORPH_CLOSE flag. But erode function worked really well.

(g) **Experiments & Results**





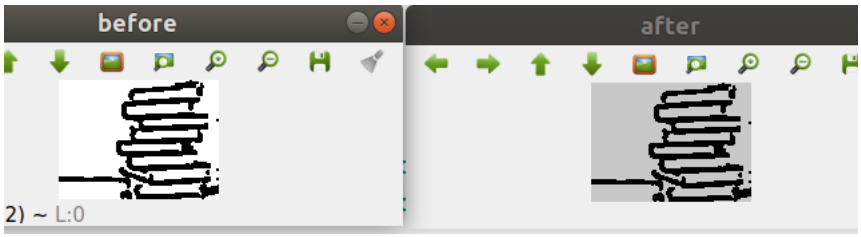
The results of this hypothesis were consistent and quite accurate. Number of books being reported were very close to the original count. It even worked on a more sophisticated images such as the ones shown above.

(h) **Limitations & Assumptions**

The algorithm performs good when books are kept in the shelf. The performance starts to deteriorate sometimes when book is kept flat on table and image is of top view of the books.

If we have an image such as the one shown below the performance can deteriorate.

The reason for this is the fact that the white pages of the image tend to blend in with white background which makes it difficult for the thresholding to form good separating regions for detection.



Still we are able to detect and report book count as 4.

Conforming to these limitations the algorithm fairs well in producing accurate results.

2. ***Pixels on your face:*** *Write an algorithm to mark out pixels of your face*

(a) **Initial thoughts**

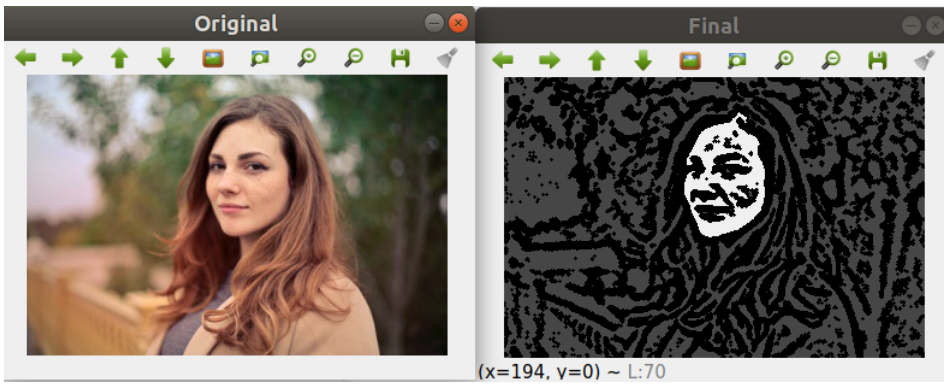
Riding on the success of previous algorithm, I decided to for a modified version of the algorithm used for book counting.
This would set a baseline and allow me to proceed in other directions if necessary. We can compare our improved hypothesis from the future and look for what will work best.

Regardless of this hypothesis working we can always have a baseline, so its a win win situation.

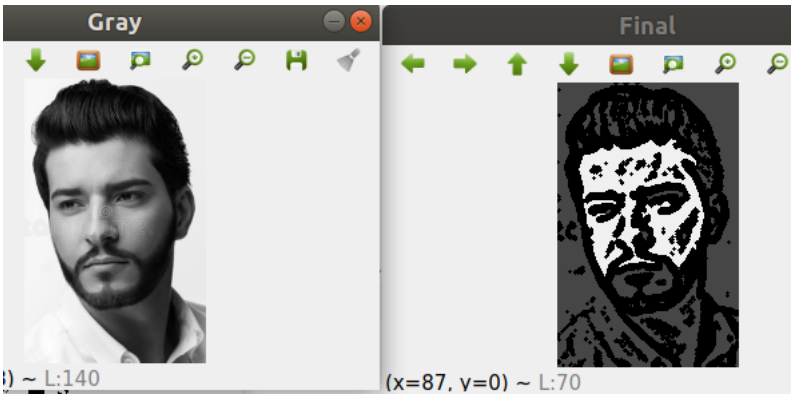
(b) **Hypothesis 1**

The idea is to modify the algorithm from book counting. We can perform the dfs and flood fill as before plus eroding logic. But for face identification we will require the aspect ratio of the discovered region to be comparable to the aspect ratio of a human face.

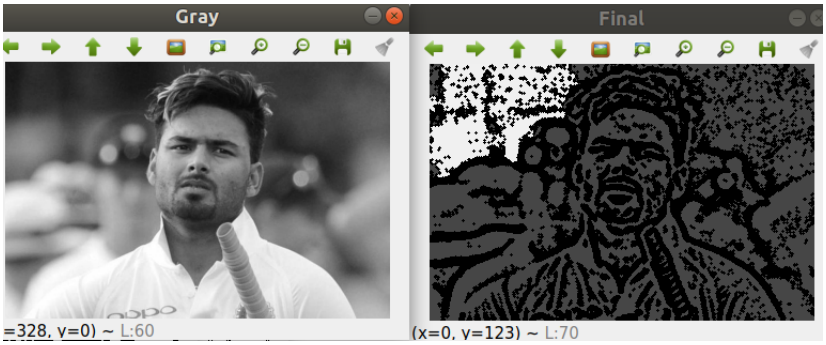
(c) **Experiments & Results**



This works very well for images having no facial hair as can be seen in the above images.



But soon it starts to deteriorate for images which can include moustache as can be seen above.



The baseline algorithm starts to fall apart when the image is a complex image where closing of edges is using erosion is very difficult as can be seen in above image.

(d) **Hypothesis 2**

I decided to apply a range threshold depending on the color tone of the character in image. We mark pixels within the range of skin colors with white and others as black.
We can then use our hypothesis 1 algorithm to detect a face.
This turned out to be a hit or a miss. Sometimes when the image had similar kind of color as skin color in background objects, this tend to fall short.

Also sometimes a part of the face was detected instead of whole face as can be seen above. We will try to improve this with hypothesis 3.

(e) **Hypothesis 3**

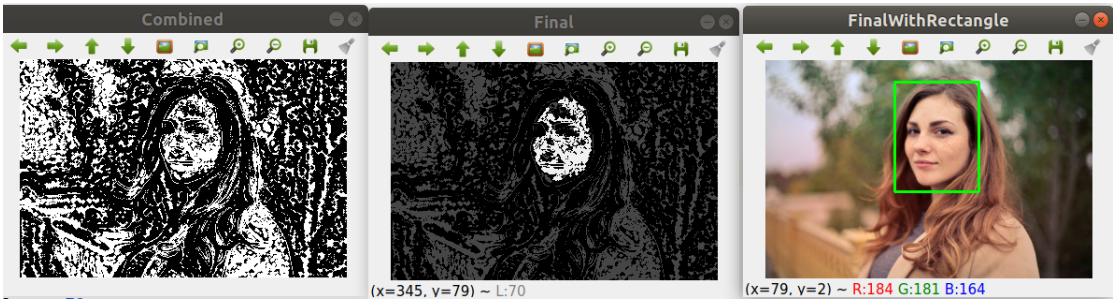
We will combine hypothesis 1 and 2 from earlier discussions. We will produce 2 images 1 from hypothesis 1 and another from hypothesis 2. We can then apply some sort of combining operation like bitwise or or nand to combine the two images into a single image.

This should do 2 things. Hypothesis 1 would help in retaining image boundaries accurately and at the same time hypothesis 2 image can discover more of the face like pixels. When we combine these 2 images we expose more of the face while keeping the boundaries intact. Then we can use our book counting algorithm to find a perfect face.

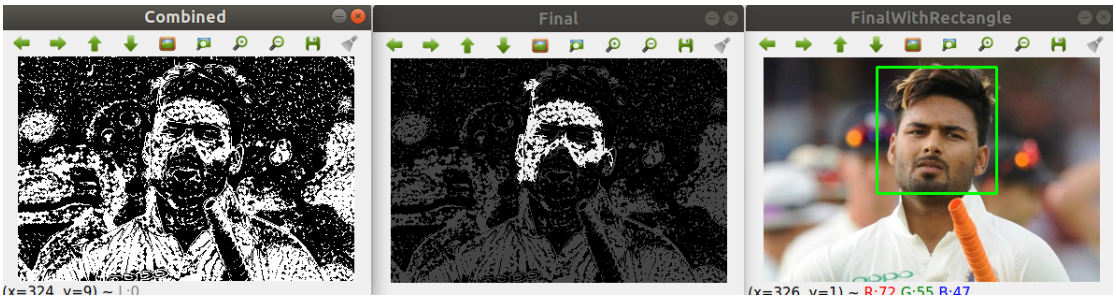
Also since we have noticed that it is difficult to mark out all face pixels individually, we can try to make a rectangle around the detect area which can solve problems for images such as shown above in case of hypothesis 2.

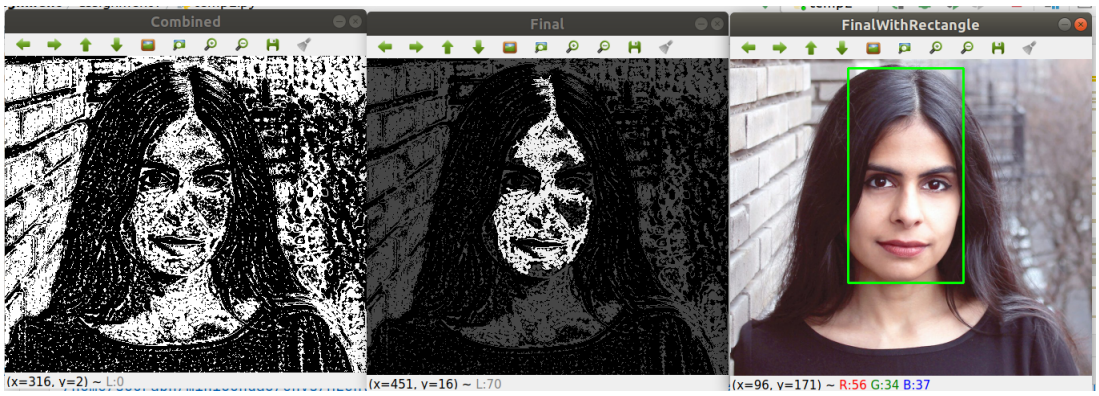
Additionally the hypothesis 1 and 2 combining using bitwise NOR worked the best for multiple images.

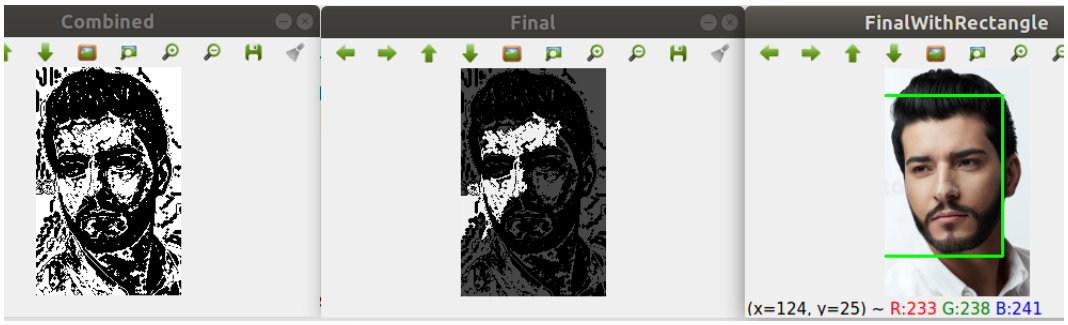
(f) **Experiments & Results**



Results seem to be quite accurate and consistent subject to some limitations discussed in further sections.
Notice how the image of woman above has a lot of skin toned objects in the background but we are able to capture the face quite accurately.
Below are results for some more complex images including Rishab Pant image.



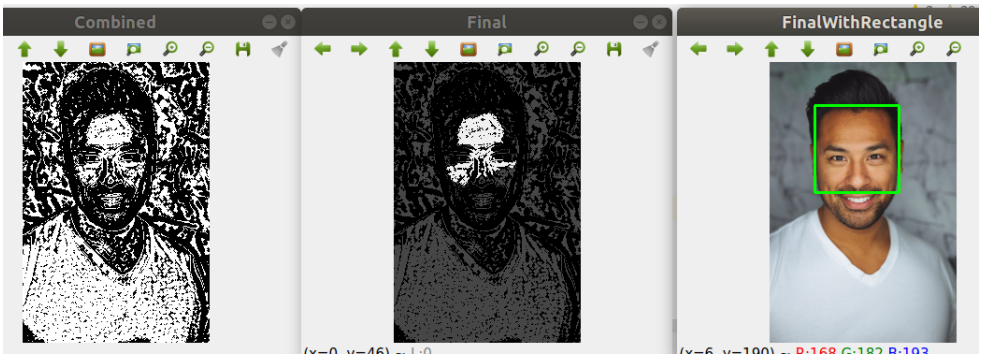


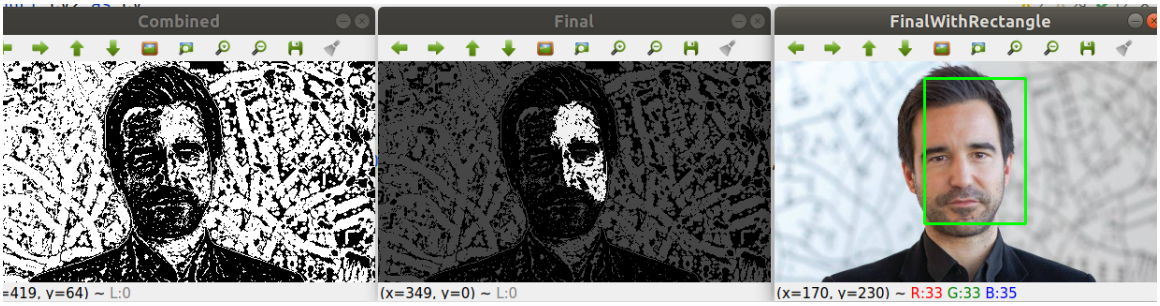


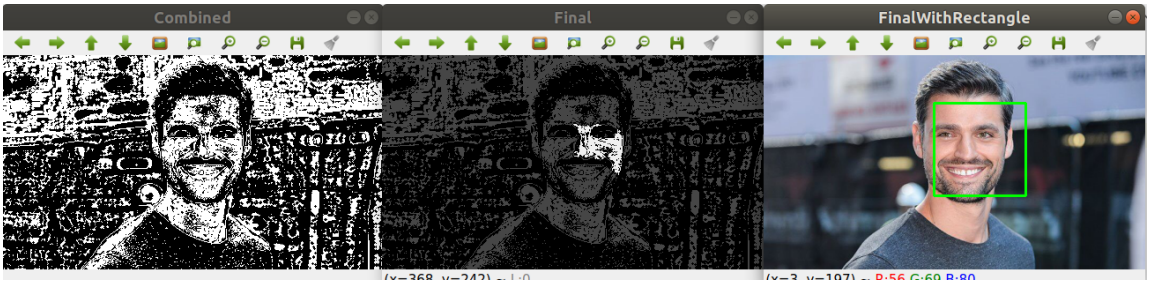
(g) Limitations & Assumptions

The hypothesis works under the assumption that there is only one face to be detected in the image.
The range thresholding done for skin color should be sometimes done manually depending upon the skin tone of the character, which is the main draw back of this hypothesis. A sharp image is preferred for best results.

Below are some more results produced by the same hypothesis.







As we can notice the results are not always accurate for the pixel marking but the rectangle provides a good approximation.