COMP9517 Assignment 1

z5169780

Task 1

Otsu's Method

My implementation of Otsu's method follows the algorithm described by the specifications. I first convert the image to grayscale which allows for its intensities to be easily identified. I use *numpy* to count the frequency of each intensity and therefore creating a histogram (representative of *weights*), I next create an array of *means* to use later when calculating variance. These two variables have been created outside the threshold loop as they will be used multiple times and their values will not change.

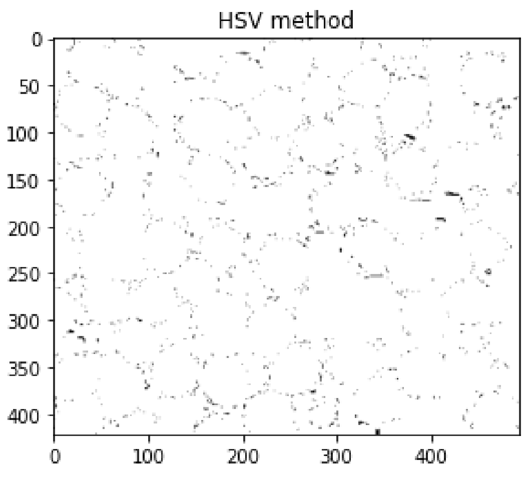
I initialize the *max_variance* and *desired_thresholds* to zero as any valid variance will render the initial values void. Using a loop, I iterate through every threshold to find the desired threshold such that a maximum interclass variance is produced. Note that in finding the maximum interclass variance, we subsequently find minimum intraclass variance.

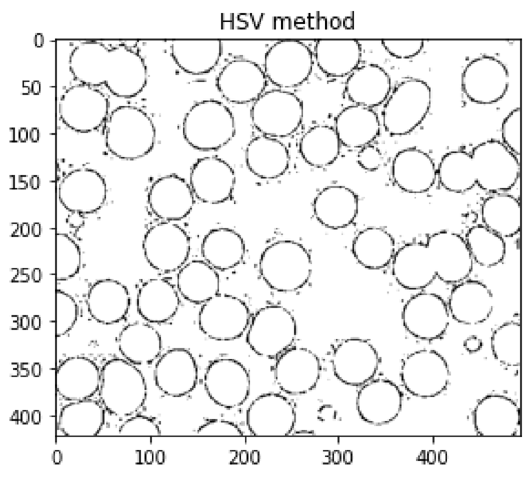
Within my loop, I calculate each threshold's interclass variance by using the aforementioned *weights* and *means*. To improve clarity, I moved this calculation into its own function but in doing so I had to use *sum(weights)* instead of *len(img.ravel())* which was something I wanted to note as it has a small impact on runtime. However, the calculation of interclass variance is heavily simplified though the variables I created as I can quickly calculate the product of weights and the difference in means with their existence. Once I have the product of weights and difference in means, it is matter of simple multiplication and squaring to find our interclass variance.

HSV Thresholding Method

My implementation of the HSV method was reached through a series of trial and errors. Essentially, I set up variable threshold parameters for the hue, saturation, and value, then adjusted them until a desired output image was produced.

I found that when the parameters are not adjusted appropriately the most common outcome would be the image only producing outlines of the blood with only the edges being identified. In my trials, I noticed that the hue needed to be relatively high (>195), saturation could not be too low (>42), and value could not be too high (<228). Some interesting trends included saturation having the biggest effect upon the blurriness of outlines (left image), and value having the biggest effect on the distinctness of outlines (right image).

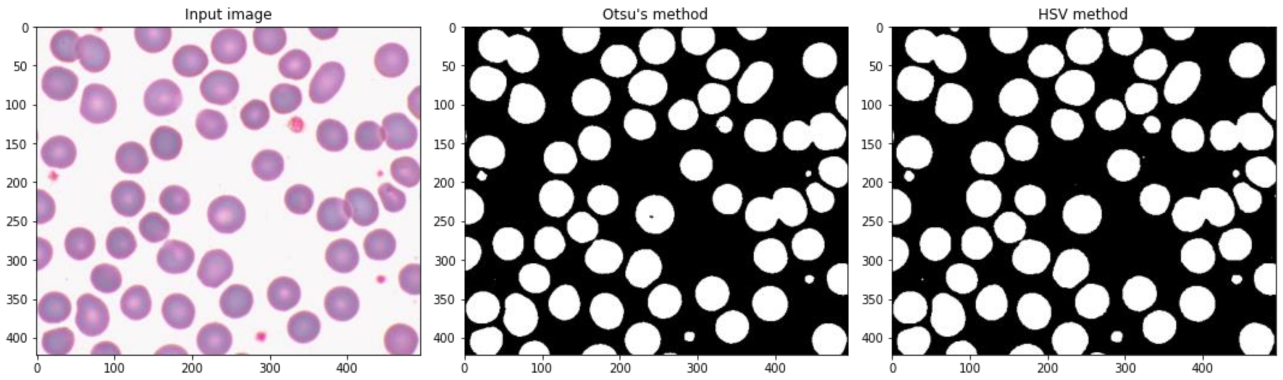




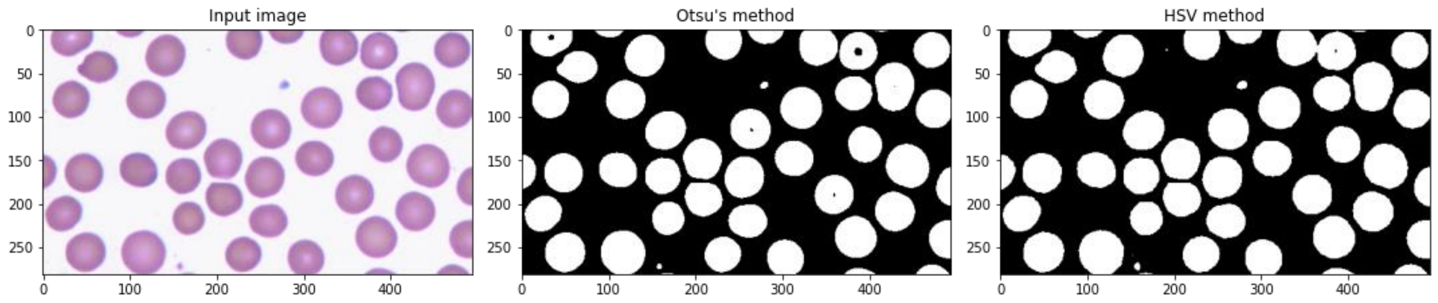
Results

All images were found to have used an Otsu threshold of *202* and HSV thresholds of *192*, *42*, and *228* respectively.

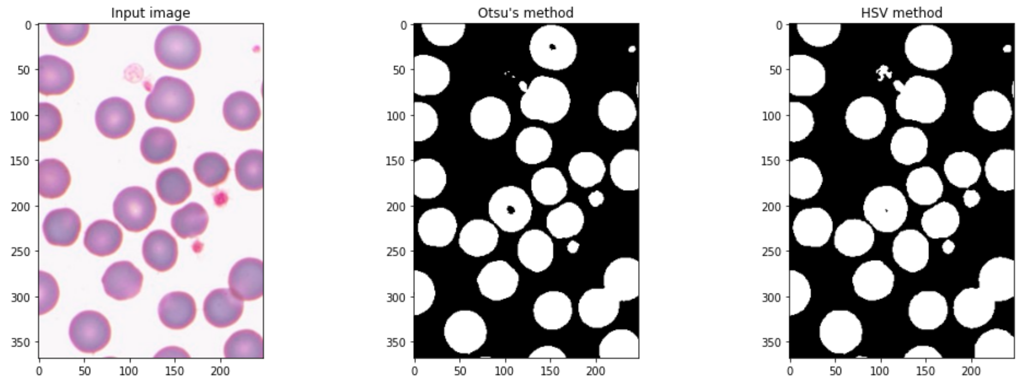
c1.jpg



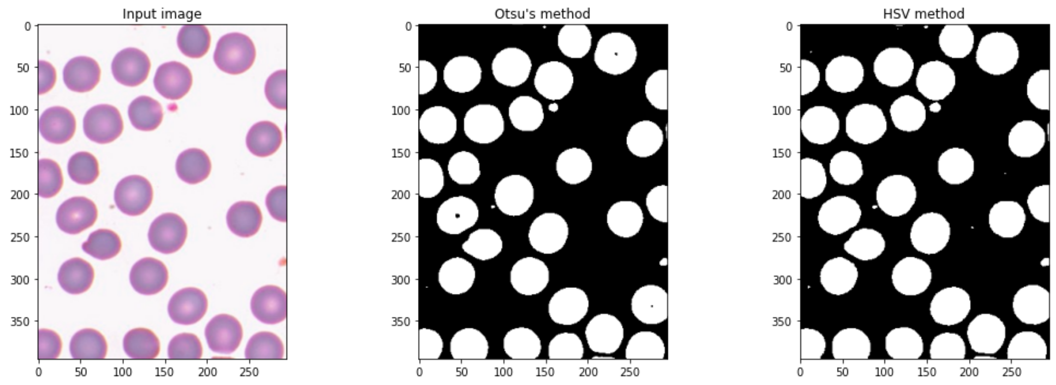
c2.jpg



c3.jpg



c4.jpg



Task 2

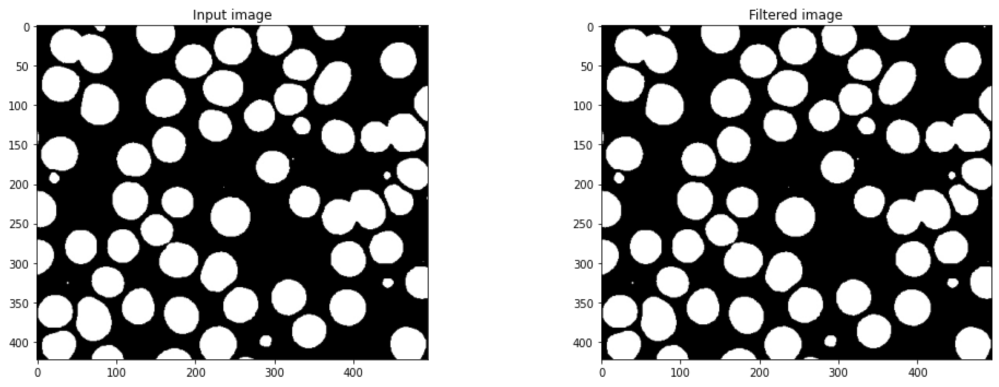
Connected Components Algorithm

The connected components algorithm involves analyzing data and attempting to detect and identity connected regions by generalizing them into characteristics. There are a few different types of algorithms, however all involve one common trend where each component is compared to its neighbors and a conclusion is reached on whether the two are connected. Once these connections are formed, clusters emerge and an idea upon what components are connected can be made.

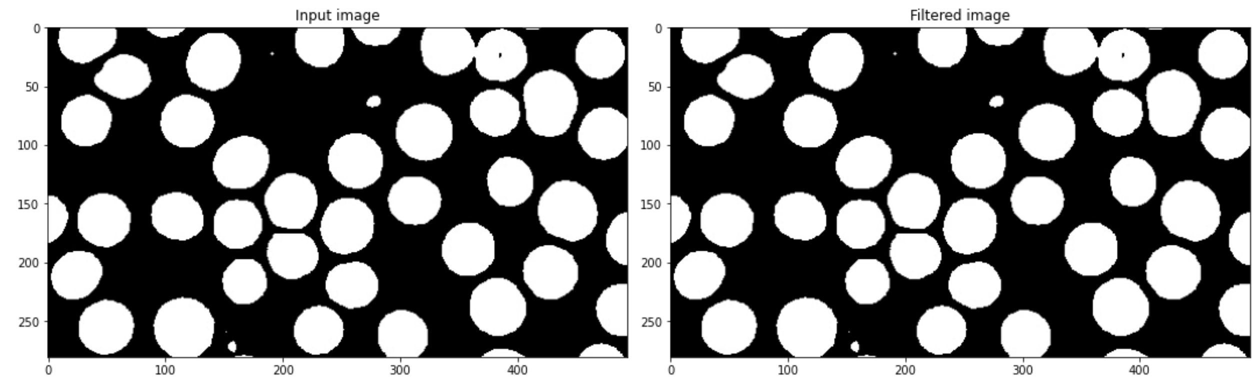
As for the two-pass algorithm that is described in the specifications, the method involves “two passes” which each serve their own purpose in identifying clusters. The first pass identifies which pixels are of relevance, that is not the background, and when these are found, label them accordingly such that they adopt the lowest label number from their neighbors. Once the first pass is complete, a general map is formed, and large clusters should be easily identified. However, there may be discrepancies such as two clusters being connected, this is where the second-pass will make final connections and modify any discrepancies. Finally, the two-pass algorithm will produce a map of all connected components or clusters for the given data which in this case is an image of blood cells.

Results

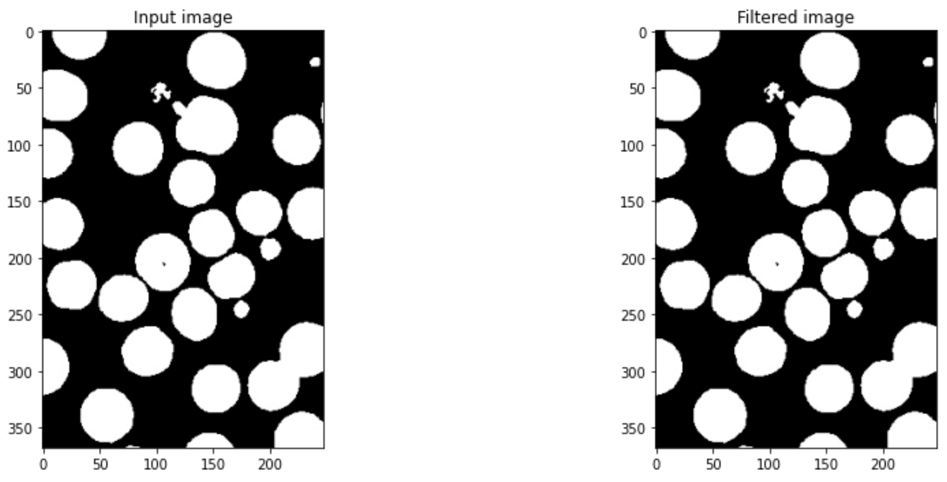
hsv_c1.jpg – *56 cells found*



hsv_c2.jpg – *37 cells found*



hsv_c3.jpg – *27 cells found*



hsv_c4.jpg – *30 cells found*