**Image Processing Worms Assignment Report**

To start I read in both image channels as grayscale, normalized them both to the 0-255 range so they were visible as images were in the range 0-1 given to us and therefore black. I then added the two normalized images together with equal weighting. Also read in the ‘w2’ channel as ‘unchanged’ for later use as well as reading in the ground truth image for the respective images read in.

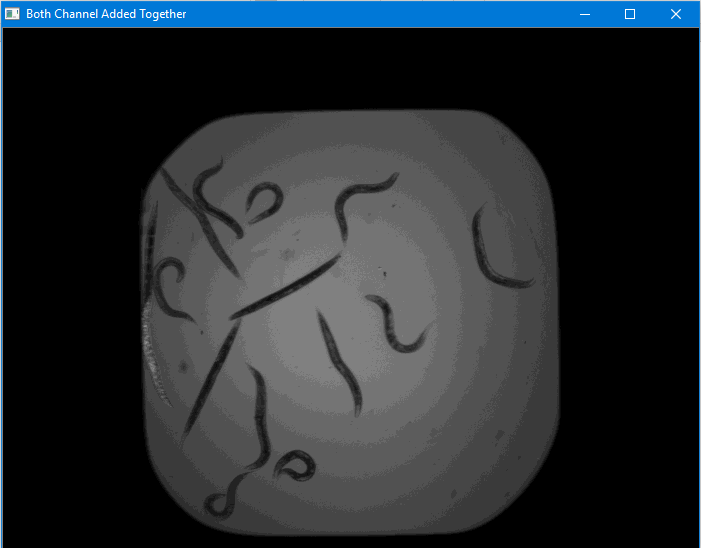


Image shows both channels normalized and

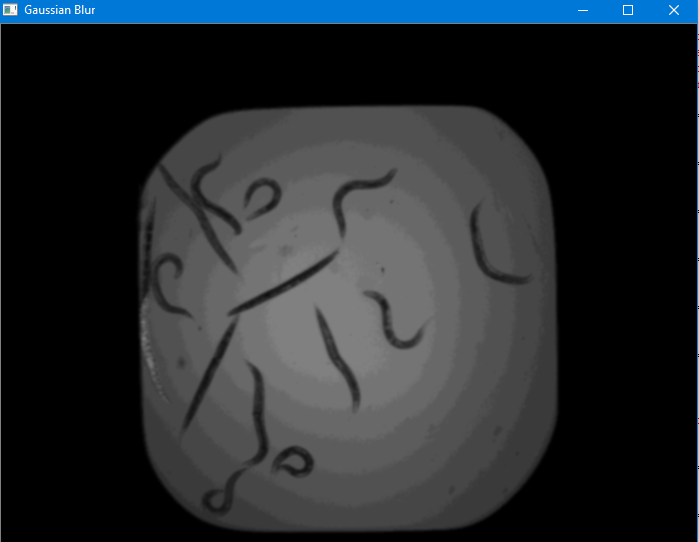
then added together.

This image is a good start but some flaws is that the worm on the far left is a similar shade to the background and that the background is not all one colour. As a result, I decided to apply various threshold based segmentations.

As the background is much lighter than the worms I could extract the worms from the background.

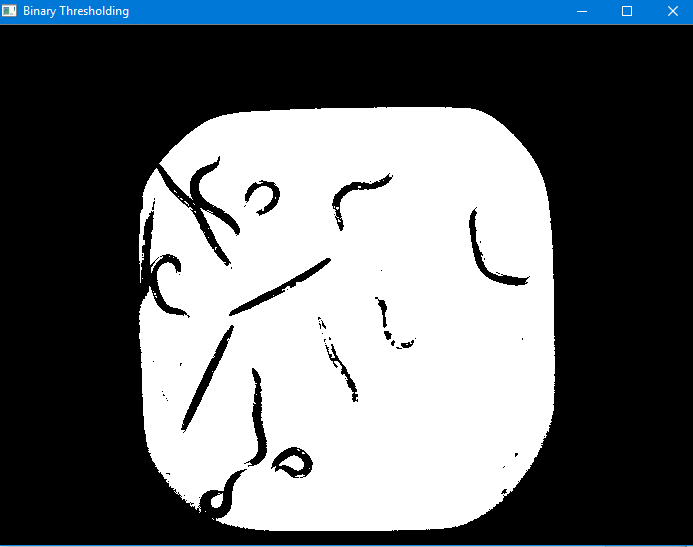
Noise in an image can cause small errors due to thresholding so it is best to first do some various forms of image filtering to remove this noise.

I started both of these segmentation techniques with a Gaussian Blur which helped to remove Gaussian noise from the image as well as median blur which helped to remove Salt-and-Pepper noise from the image.



I then implemented different segmentation methods depending on what image thresholding methods I was using

1. Simple Binary thresholding



I found a threshold value of 54 gave me the best results

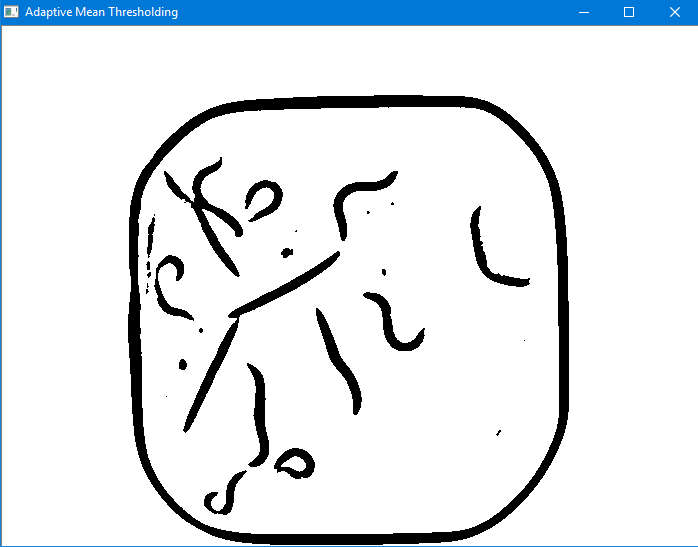
I then inverted the image so I could effectively use morphological transformations. I.e. all worms now white with a black background.

For these original thresholding methods, I found a kernel size of 3 gave me good results.

I applied morphological opening to remove the noise in the rest of the image and morphological closing which removed the small holes due to noise inside the worm object.



Results of morphological transforms

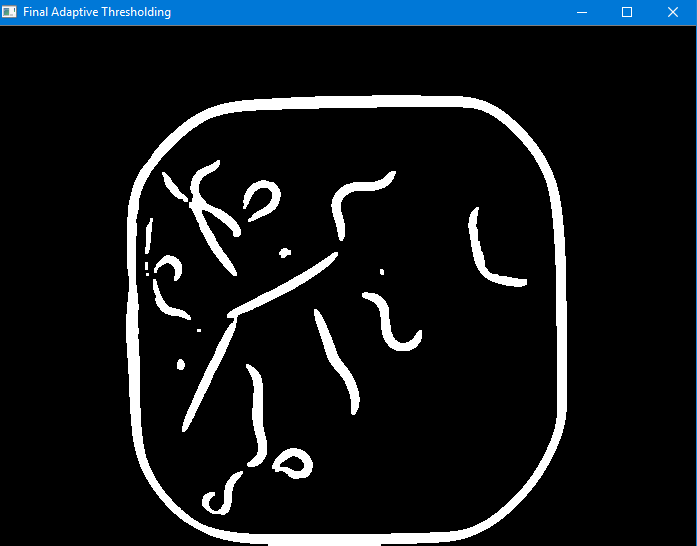
1. Adaptive Mean Thresholding

I repeated this process but in exchange of binary

thresholding I used adaptive thresholding.

Image on right is pre-morphological transforms.

Adaptive thresholding gives a much better output as the algorithm calculates the thresholding for small regions in the image. This gave much better results in terms of the quality of the worms but also left a border. I found a block size of 33 and a constant of 10 gave me the best results regarding the quality of worms.

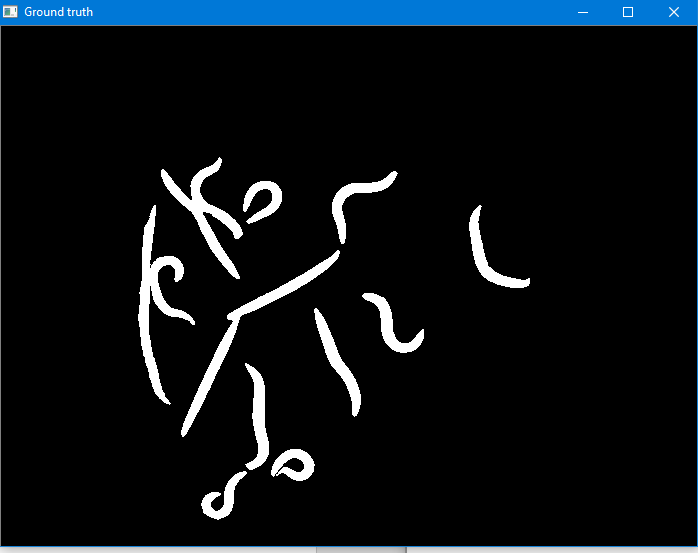


Post-morphological transforms

I then compared both methods (i) and (ii) to the ground truth data.

By visual inspection they are a good start.

Ground Truth Image

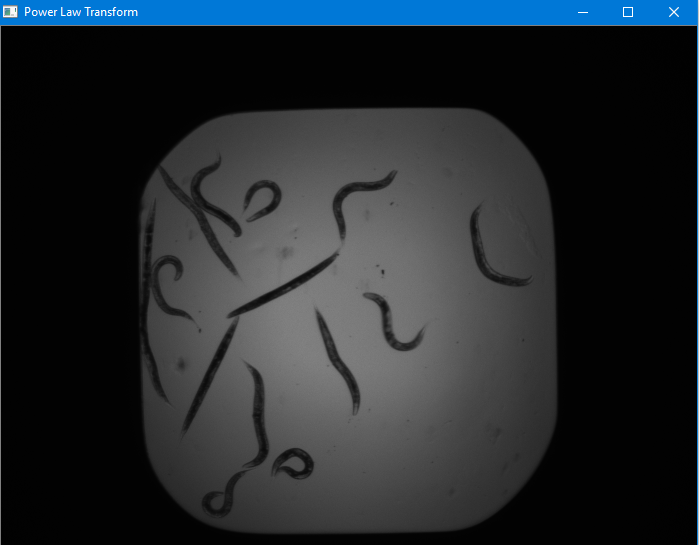


Comparison via taking the difference between the 2 images.

Segmentation method 1 comparison Segmentation method 2 comparison



To get a better image to compare to ground truth data, I read in the ‘w2’ band image only and started with a power law transform where I used a gamma of 1.3 to brighten the image.



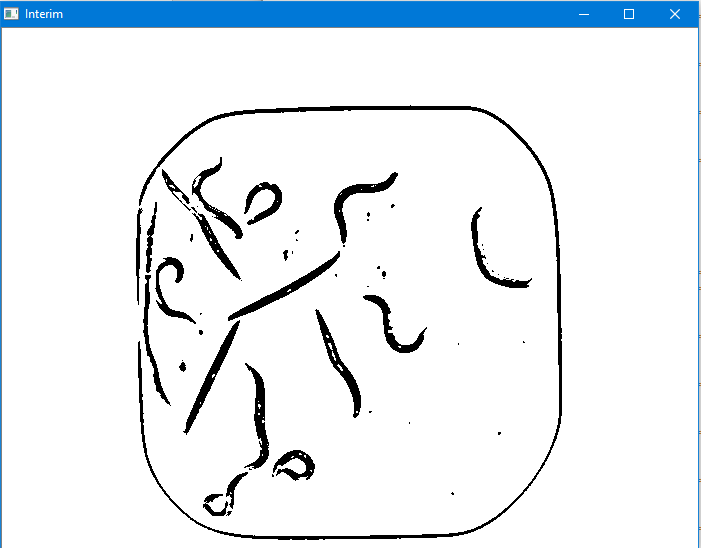
Power Law Transform

I then got a clear white frame with well-defined edges and an image with well-defined worms before adding them together.

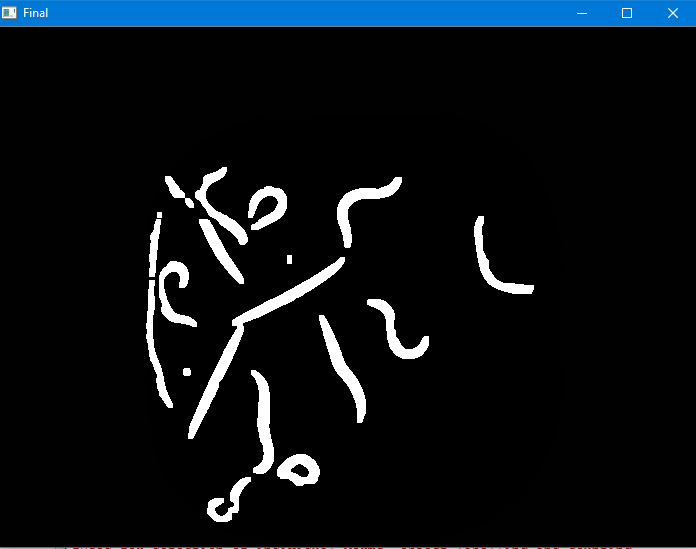
I did this by converting the image to 8-bit before using a median blur and an adaptive histogram to improve the contrast of the image and then doing Otsu’s binarization thresholding.

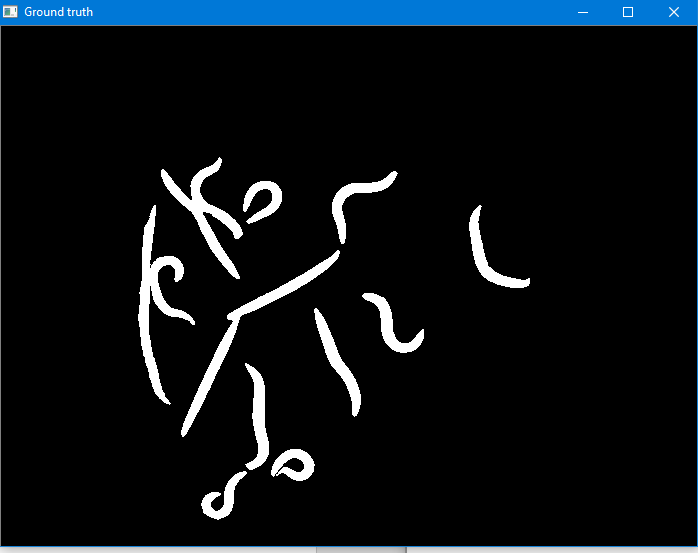
I then used adaptive thresholding, a median blur to remove noise and bilateral filtering to further remove noise without ruining the edges.

Interim Image of segmentation and background separation.



I then inverted the image before applying various morphological transforms however this time with a kernel size 2.

 Final Segmented Image Ground Truth



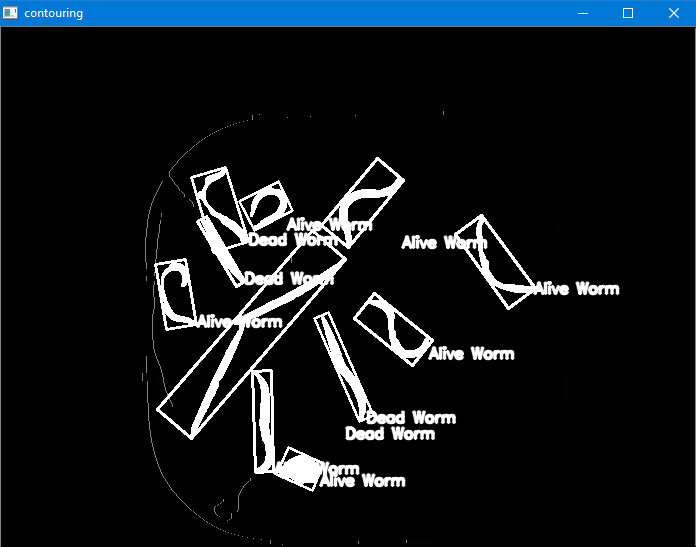
Very good segmentation in comparison to ground truth.

I then found the contours of the image. I looped through these contours and if their area was greater than 250 and less than 10000 I plotted a minimum area bounding rectangle. I then drew the contours on the image.

I calculated a rough length of each worm/contour using perimeter of contour/2. If this length was greater than the length of the diagonal of the bounding rectangle, I classed it as dead, otherwise alive. I labelled each worm dead or alive.

I counted the number of contours to get the number of worms.

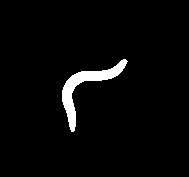
I printed each worm individually onto a black background and wrote to individual files.



Clearly shows evidence of success of system. Straight worms classified

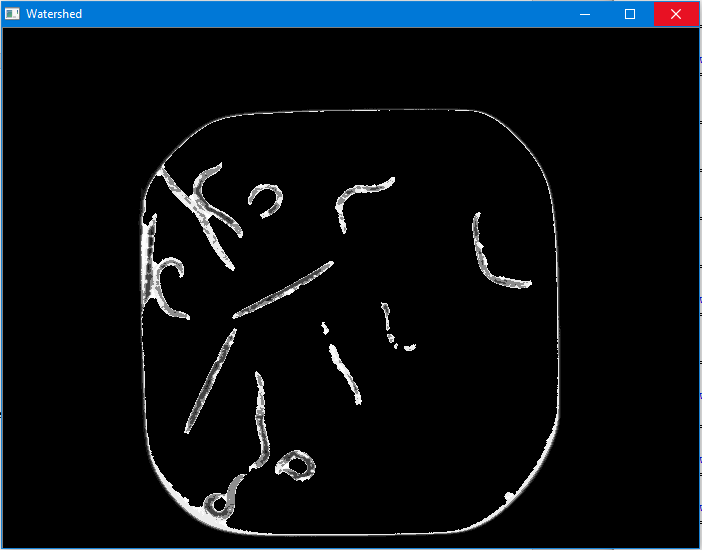
As dead and curly as alive.

Counts the number of worms as 11 which is a good level of accuracy.

Worm Written to file Individual Worm Ground Truth

Very clear evidence of system performing the specific task of separation of individual worms.

Watershed Algorithm



**Sources**

-               <http://stackoverflow.com/questions/11294859/how-to-define-the-markers-for-watershed-in-opencv>

-               <https://stackoverflow.com/questions/41555031/identifying-curved-and-straight-objects-in-opencv>

-               <http://blog.christianperone.com/2014/06/simple-and-effective-coin-segmentation-using-python-and-opencv/>

-               Lecture demos

-               <http://opencvpython.blogspot.co.uk/2012/05/skeletonization-using-opencv-python.html>

-               <http://stackoverflow.com/questions/34834523/an-alternative-way-to-skeletonize-in-opencv-python>

-               <http://stackoverflow.com/questions/15135676/problems-during-skeletonization-image-for-extracting-contours>

-