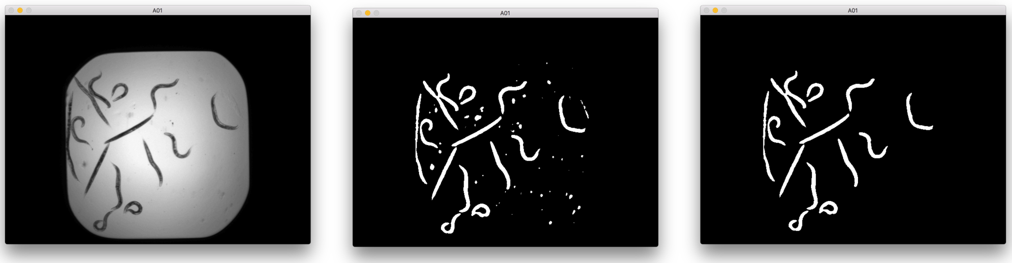
Reading Image Channels

Both channels are loaded as 16 bit-grayscale matrices. Images are normalised to preserve the data in the images. They are then divided to make values fit within an 8-bit matrix.

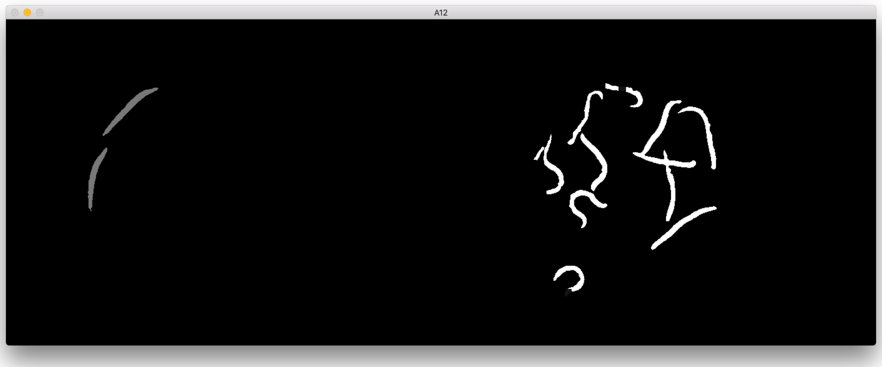
Image and Background segmentation

GFP (W2) - is segmented with a Gaussian blur, followed by Adaptive-Mean Thresholding. Non-worm objects are removed using contours. Where a border is present, A mask is created by segmenting the background, eroding it, and merged to the base image.

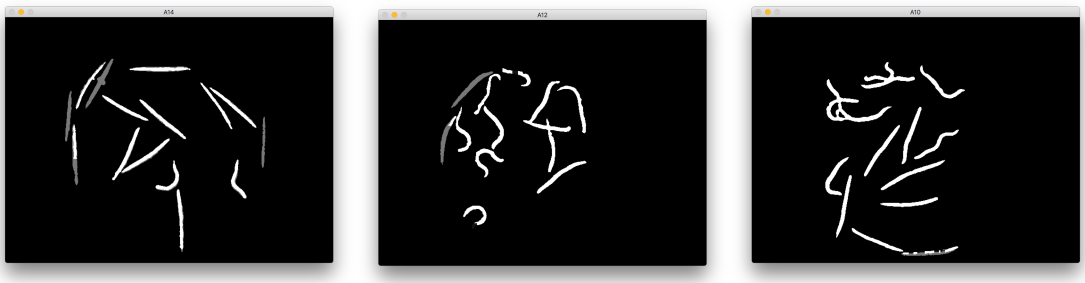


Images from left to right: Normalised, through thresholding, and Sanitised

Brightfield (W1) - Is subject to a Gaussian blur, Binary Thresholding and is checked for particles. There has been difficulty in extracting worms due to the image lacking sufficient worm data. W1 is merged with W2 to make a working file.

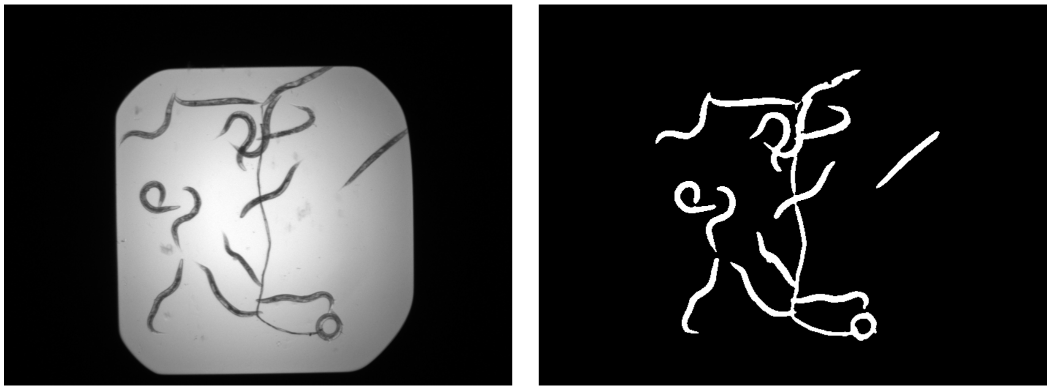


Some worms in W1 (Left) are absent, but worms found in w1 are absent in W2 (Right).



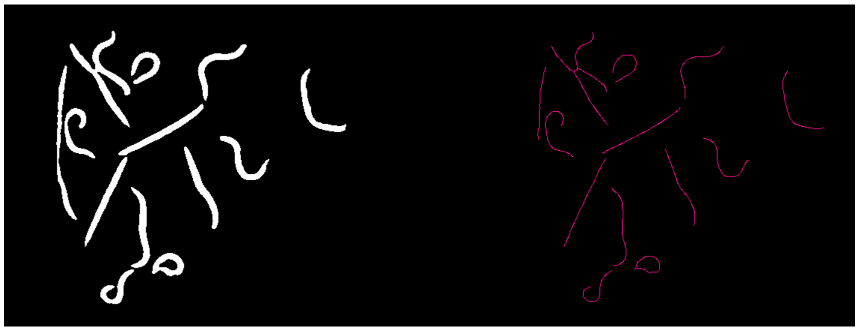
W1(Grey) and W2 (White) are merged together.

This method works well, but there have been exceptions, i.e: A large trail in `A03` was segmented.



Worms Detection

A skeleton of the workspace image is created using Zhang Suen's Algorithm[1][2]. This is cached locally.



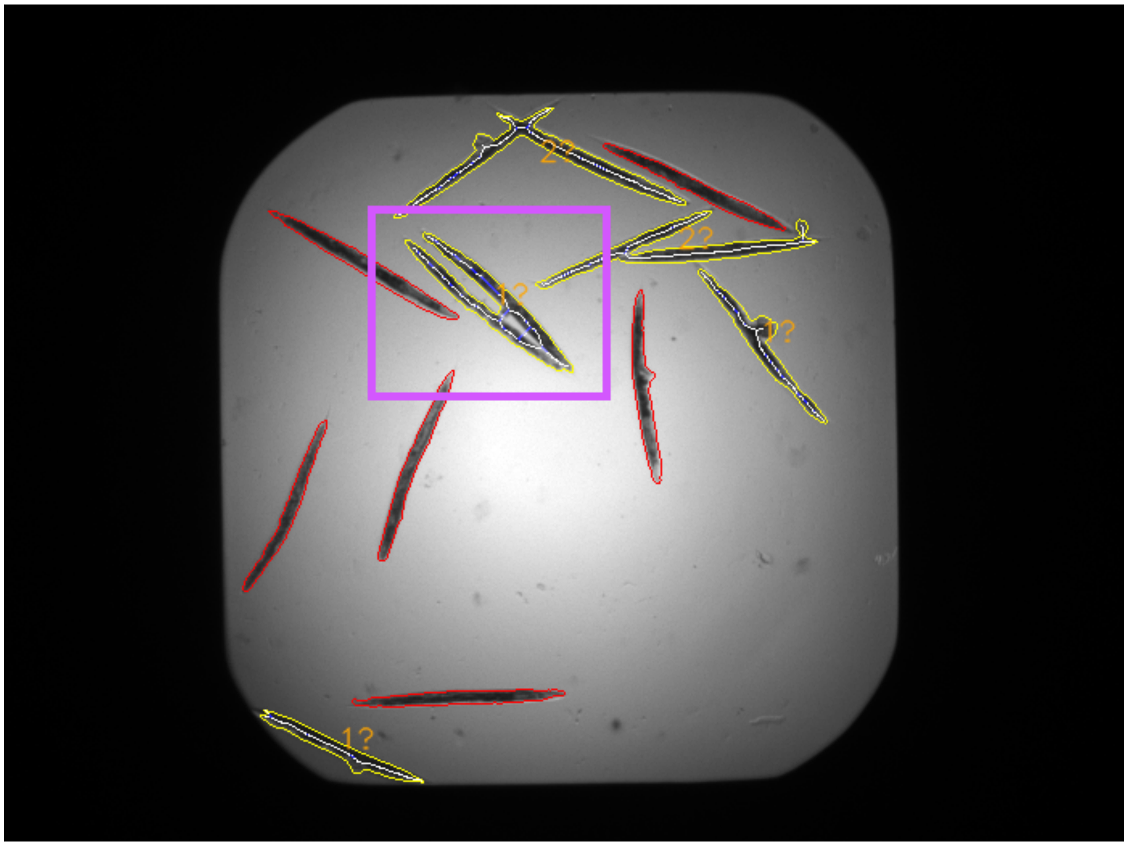
The skeleton (Pink).

Object Labelling

Worm blobs are used as a mask to retrieve its skeleton. This is used to find endpoints. If there is 2 or less endpoints, then it is a worm. Each individual worm blob is saved locally.

Counting

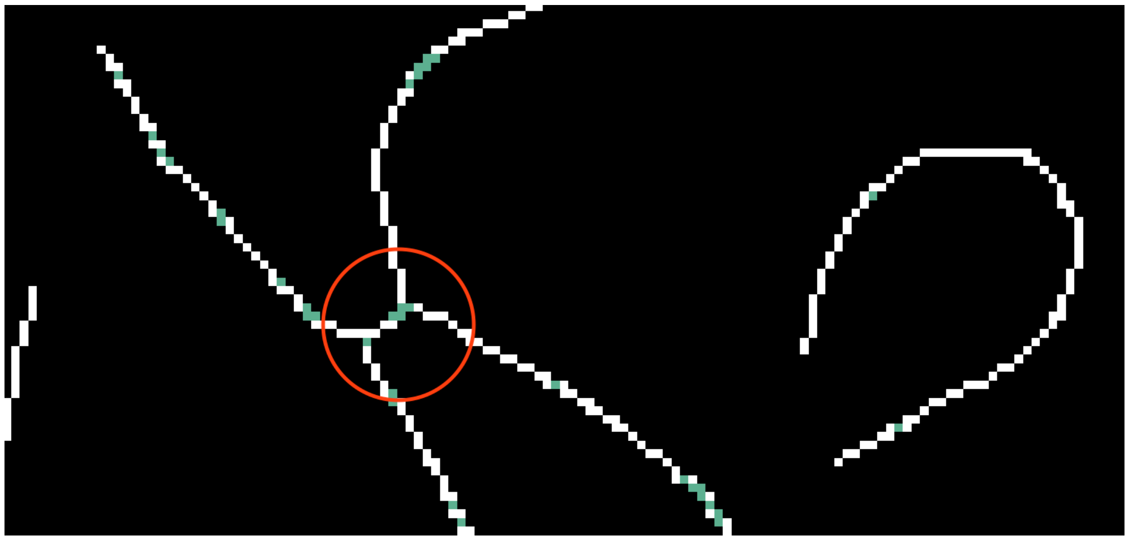
Counting is deduced with the number of worm endpoints. The formula `no_worms = Math.floor(len(endpoints)/2)` assumes that each worm has two endpoints. This is if the skeleton produced is accurate.



The formula detects one worm in a cluster (purple box); There is two. This is related to the fact that the skeleton has detected 3 endpoints. A more precise skeleton would improve results.

Detection of Clustered Worms

Each skeleton's pixels are checked for its number of pixel neighbours. 1 pixel is an endpoint. Two pixels suggests a traversal of the worm. More than 2 suggests an intersection. There have been issues with this method, where the skeleton uses several pixels to draw a curve in the worm. This could be solved by creating a library of verified intersections, which can be used as a reference to detect intersections.



Detected intersections (green) and Actual intersection (Red). Curves are erroneously detected as intersections.

Live/dead Worm Classification

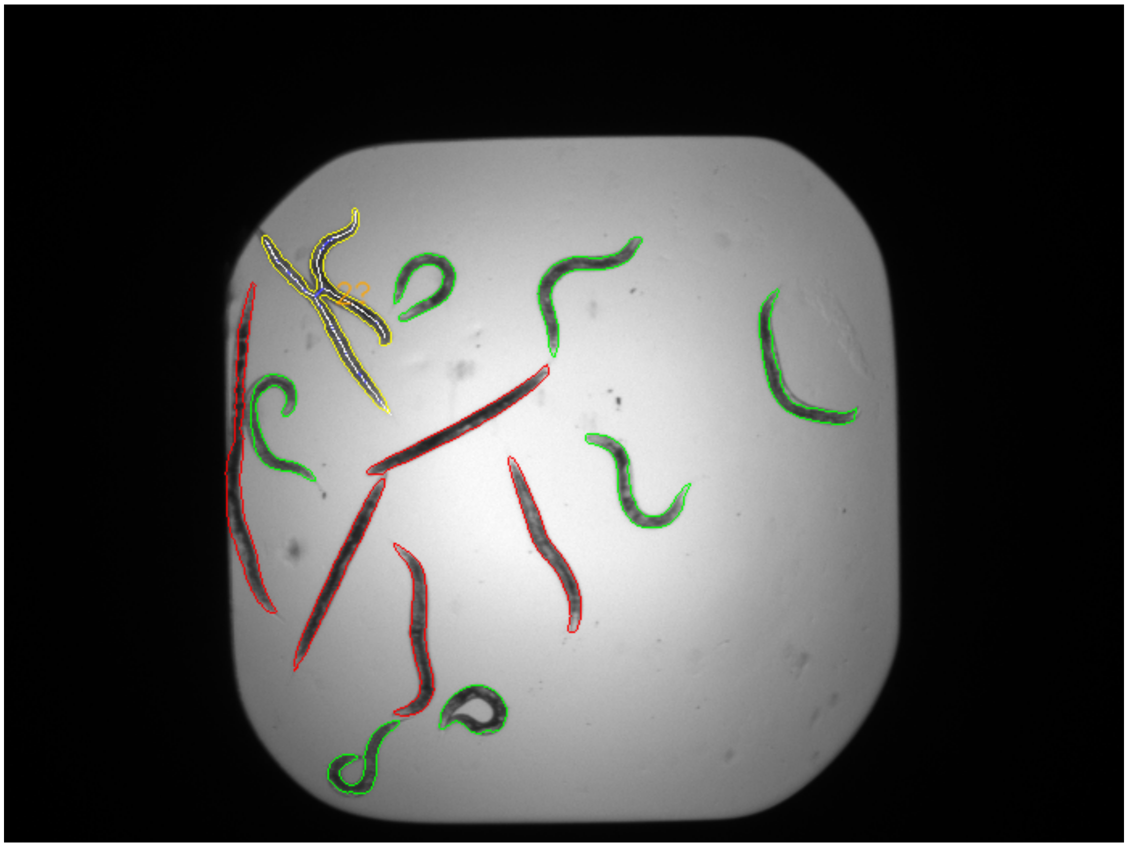
Shape Detection is done by using skeleton endpoints. A length is deduced using endpoints with Pythagoras. This is compared to the number of pixels in the skeleton. A percentage is deduced. The higher the percentage, the straighter the worm, and the more likely it is to be dead. If there is one endpoint, then the worm is in a circle - it is alive.

This worm (A01, Worm 4) has a 99% chance of being a worm.

Texture Detection - Worms are checked against its Gaussian Blur by calculating the Euclidean distance. The results (A percentage of difference) is deduced. The higher the difference, the more uneven in texture it is. are volatile due to lighting amongst other issues affecting the image.

A worm (A01, Worm 12) having a pattern difference of 41%.

The two percentages are weighted and combined to give a `dead_percentage`. If this value is greater than the `death_threshold` it is classified as dead. Each worm's contour are coloured and displayed on an image (which is saved locally for displaying later).



Dead (Red), Living (Green), and Clustered worms (Yellow). A number (Orange) represents number of worms in cluster.

Evaluation

Results are saved in a `.csv`, which can be found in the `/data/results_data/` directory from the `.py` files (base directory). Coverage and Worm Count are checked against ground truths. Overall Coverage is measured by comparing the number of pixels.

Overall Coverage Percentage	Overall Classifiable Worm Count Percentage	Overall Worm Count Percentage (Including Clusters)
93.30927835	60.43298969	90.80412371

Overall, the segmentation coverage is fairly accurate, having a 93% accuracy. This could be improved with better source material. Classifiable Worm count is low due to the number of clustered worms. The image has a good percentage of worms counted (including clusters). Verification on the status of the biological status of a worm is absent; checking statistics on dead worms is unfeasible.

References

[1] T.Y. Zhang and C.Y. Suen. A fast-parallel algorithm for thinning digital patterns. Image Processing and Computer Vision, 27:235–239, 1984.

[2] Linbo Jin, Skeletonizing by Zhang-Suen Thinning Algorithm, Python and Matlab Implementation. 2014. https://github.com/linbojin/Skeletonization-by-Zhang-Suen-Thinning-Algorithm