Automated Neurite Quantification

Within the GitHub (https://www.github.com/Bearnie-H/MTG-Software) repository is a Python script used for quantifying neurite growth. This is the *DRG-Neurite-Quantification.py* script, found within the *Python* sub-directory.

In order to use this tool, you must clone the full repository as described in the main **README** file, and then install the pre-curated list of Python packages as described in the Python-specific **README** file. Once all of the dependencies have been installed correctly, this script can be used.

This script generates two sets of quantification outputs:

- A distribution of neurite lengths, as measured from the centroid of the DRG body
- A distribution of neurite orientations, deviating from the mean overall orientation

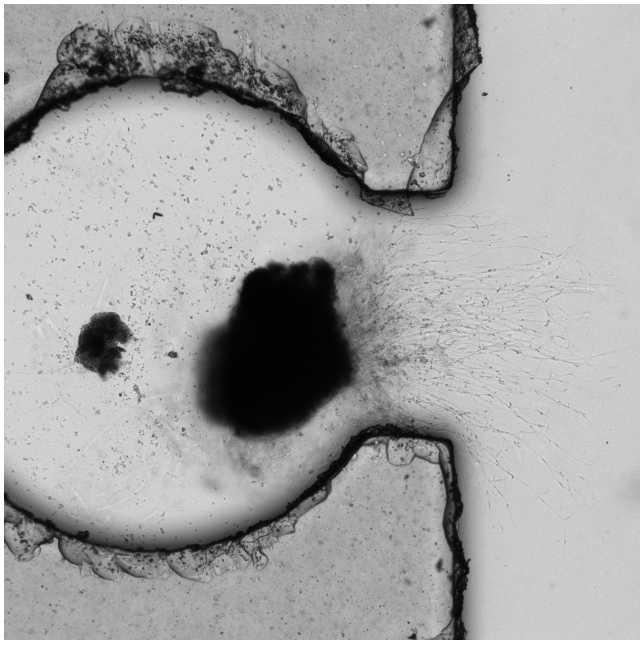
Image Requirements

This script is designed to work with a pair of input images:

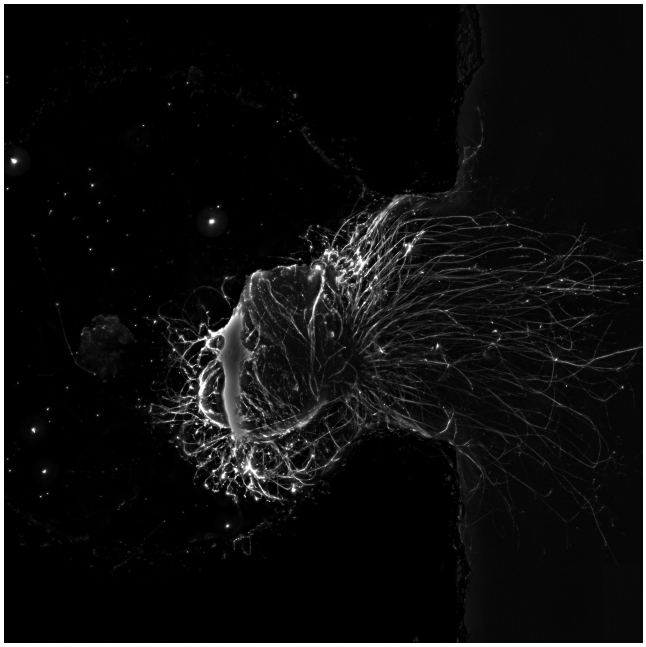
- A bright-field image
- A maximum intensity projection fluorescence image

These must be taken at the same location, and with the same magnification. An example pair of images are shown below:

Bright Field



MIP



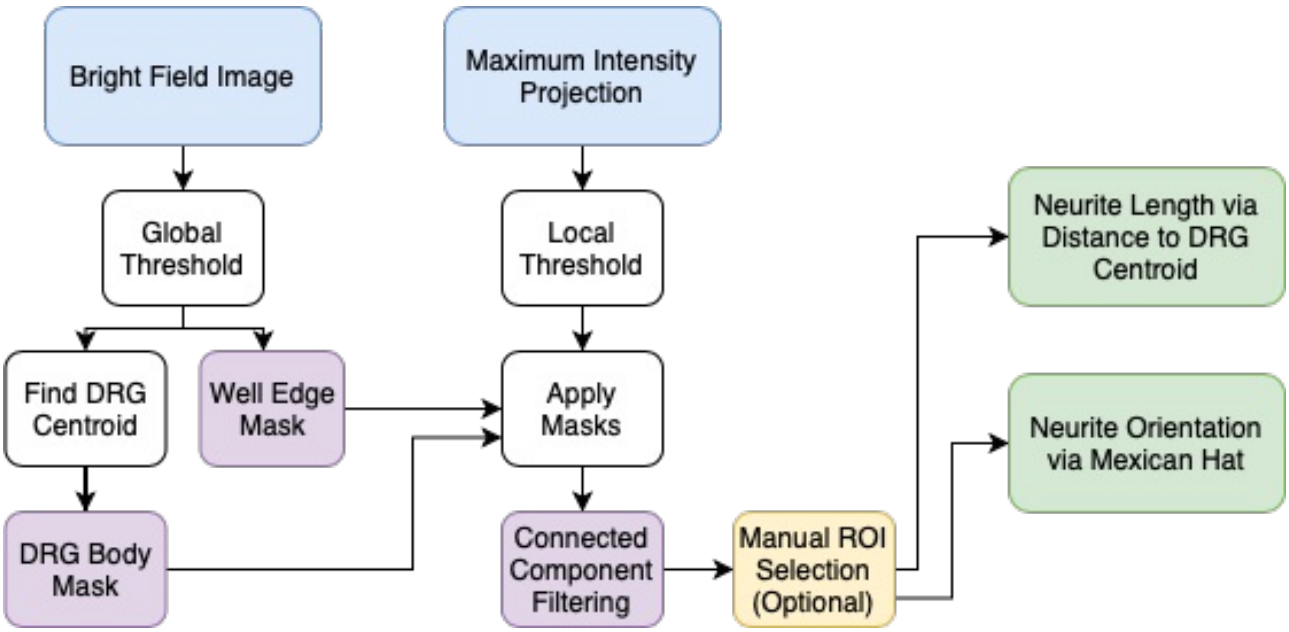
These images need not be single-channel 8-bit images, but will be converted into this format within the script. As noted, these images must be taken at the same location and under the same magnification as features from the bright-field image are used to index into the maximum intensity projection.

Other Input Requirements

In addition to the input images to process, this script requires additional command-line parameters to fully specify how to proceed with analyzing the images. For the full set of parameters, and a description of what they do, see the *--help* menu provided by the script. **NOTE: This is currently being implemented and the exact set of parameters will likely change once finalized**

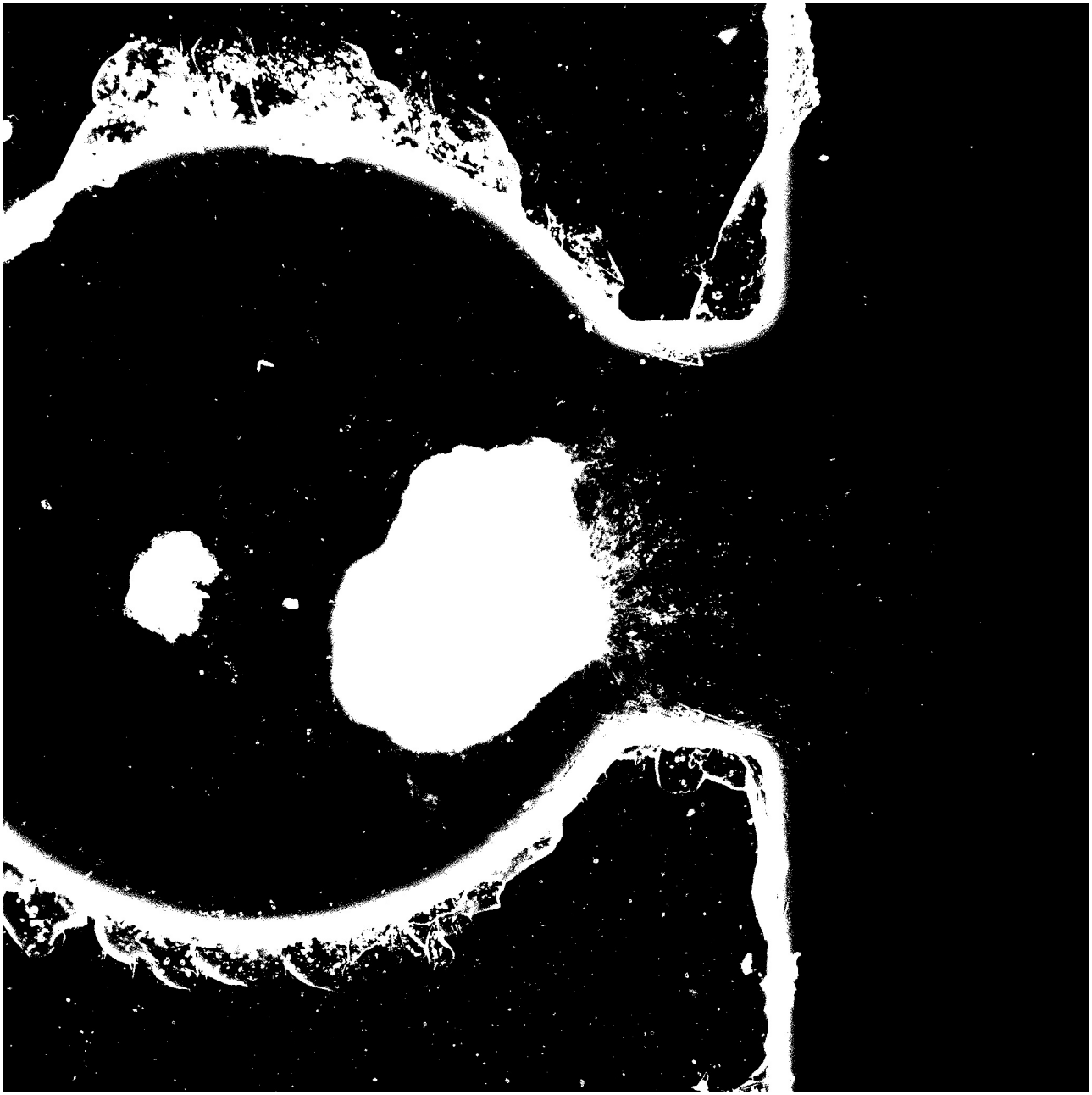
Processing Steps

The processing of the images within this script is described by the following flowchart:



Bright Field Image Processing

The bright field image is first segmented using a global threshold, with the threshold level selected via [Otsu's Method (https://en.wikipedia.org/wiki/Otsu%27s_method)](). This results in a binary image, such as the following:



With this image, three results are extracted to be used with the maximum intensity projection image.

DRG Centroid Identification

The centroid of the DRG body is identified from the binarized bright-field image by leveraging the fore-knowledge that the DRG body should be the largest, circular-ish shape within the well of the chip. With this in mind, a sequence of growing circular kernels are convolved over the image to identify regions of strong cross-correlation. By tracking the centroid of the peaks of the correlation map as the kernels increase in size, the centroid of the DRG is found as the centroid location converges.

This method can fail if the DRG body is very non-circular, attached to the walls of the well, or there are other similarly sized pieces of debris within the well. These antagonistic examples are not expected to be the norm, and are therefore considered to be able to be pre-filtered from the candidate images even being considered for analysis.

DRG Body Mask Generation

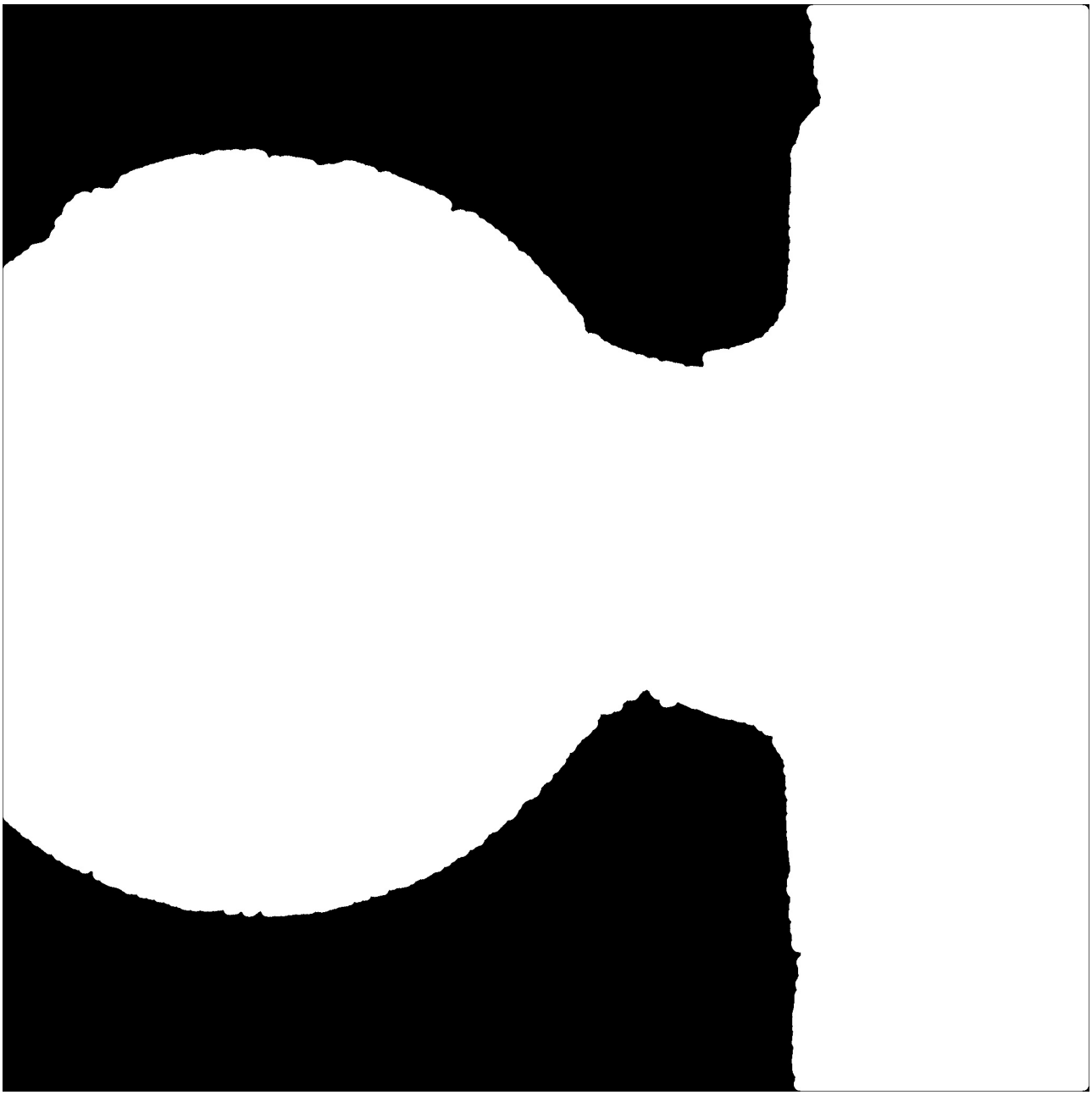
Once the DRG centroid is identified, a mask can be generated to remove the pixels corresponding to the DRG body from the fluorescence image. Using the same binarized image as before, the pixels are interpreted as connected components. These components are sorted by the distance between their centroid and the DRG centroid, in ascending order. The closest component, with at least a minimum area, is deemed to be the actual shape of the DRG body. This shape is then written out to a new image mask to be applied to the fluorescence image. An example is shown below:

Well Edge Mask Generation

The final result to extract from the fluorescence image is a mask corresponding to the walls of the chip. We know that neurites cannot grow out of the gel and into the surrounding PDMS, so any signal coming from the PDMS region can be immediately ignored.

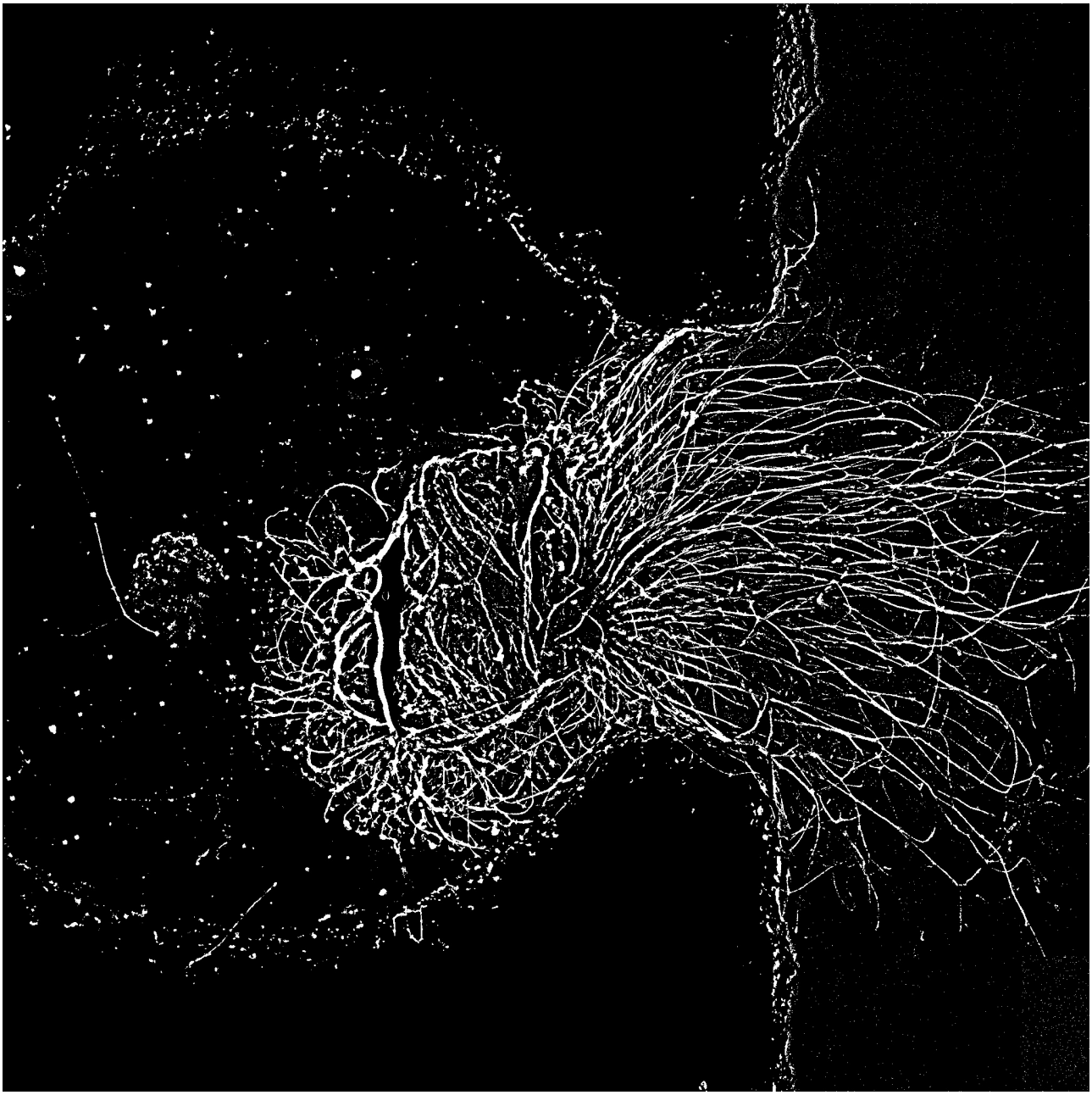
Using the binarized bright-field image, and identifying connected components again, these components are then filtered. The criteria for selecting components corresponding to the well edges are slightly different than for the DRG body. In this case, the components must be at least 66% the width or height of the image.

Once these largest components are written out into a temporary image, the closed contours within the image are identified. Using the location of the DRG centroid, any closed contours which do **NOT** include the DRG centroid on their interior are deemed to be part of the PDMS region, and are masked away. The result is the interior volume of the well left to be included in the final mask:



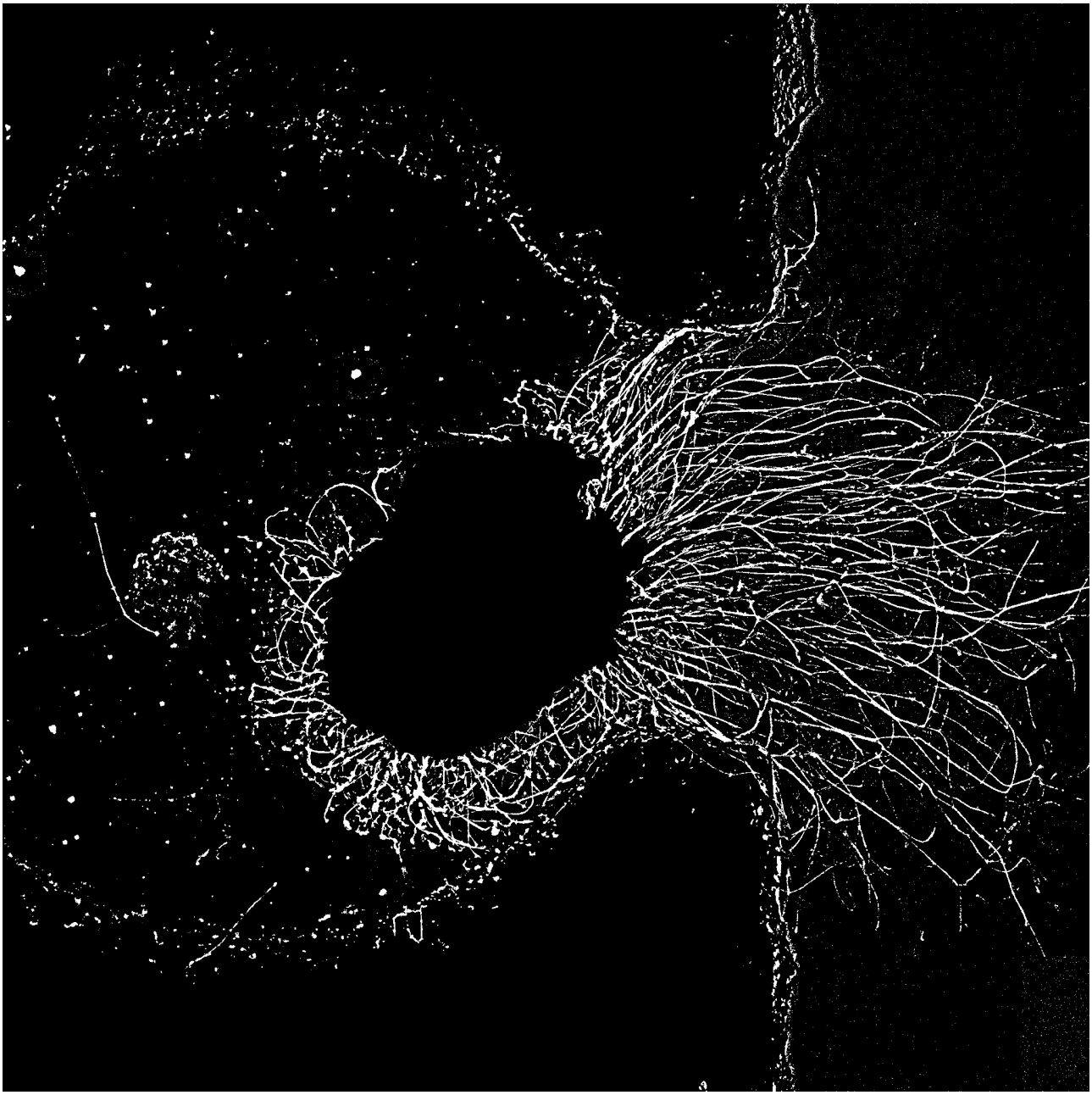
Maximum Intensity Projection Processing

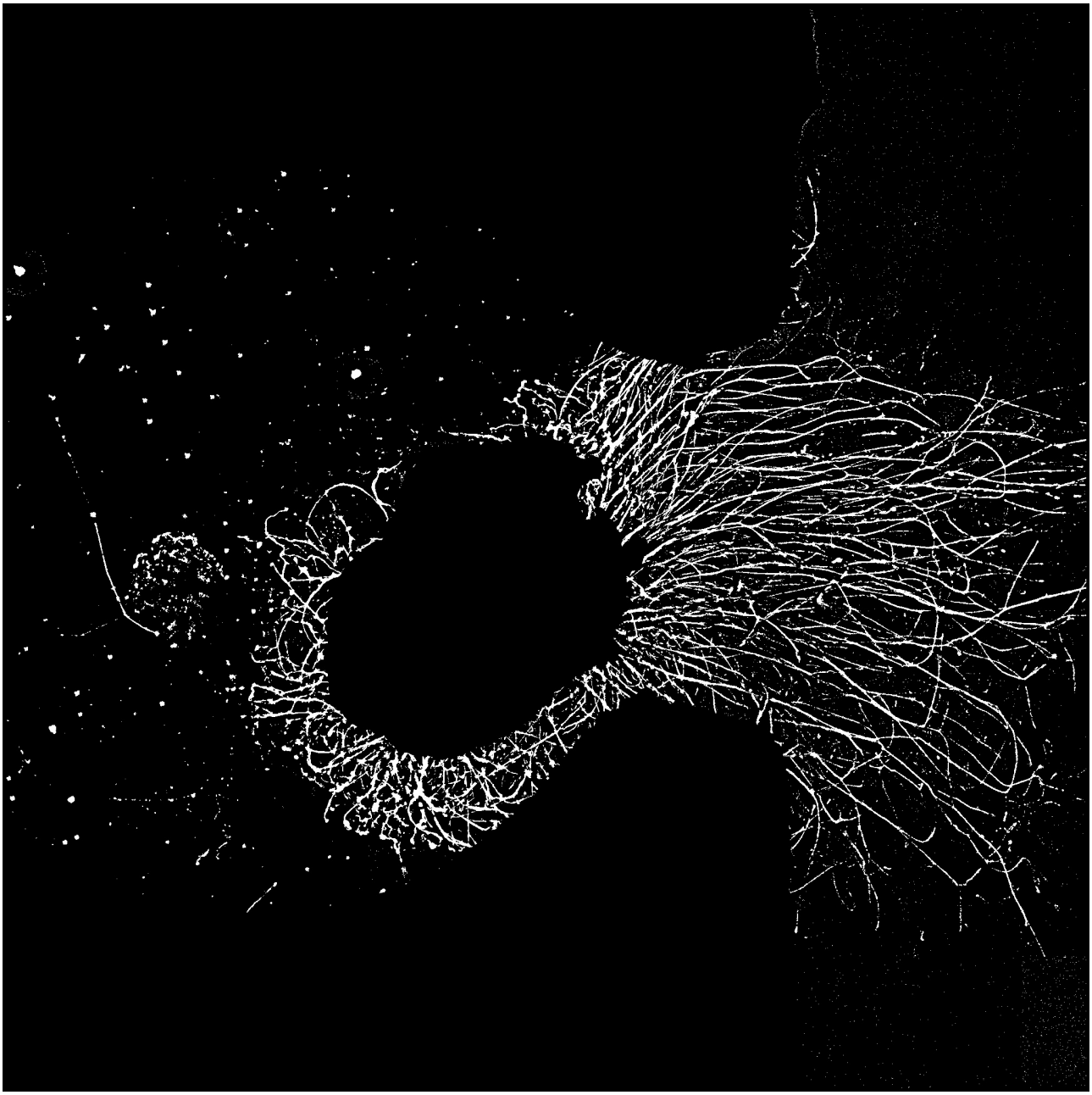
Once the above results have been extracted from the bright-field image, the first step is to apply a local threshold over the image to segment out the neurite signal from the background. A local threshold must be used to account for the generally varying contrast between neurite signal and background. The result of such a local threshold is shown below:



By varying the kernel size and offset value, this local thresholding operation can be tuned. Work remains to be done in tuning the relationship between the length-scale over which neurite signals are expected to dominate and the kernel size to use for this operation.

As can be seen in the image above, the neurite signals are much clearer in the resulting binary image following local thresholding. Next, the masks generated from the bright field are applied as exclusion masks, removing all pixels within the body of the DRG and outside of the wells. This masking must be done **after** the local thresholding to prevent the introduction of an edge artefact in the image due to the mask. The result of the DRG body mask and well edge mask are shown below.





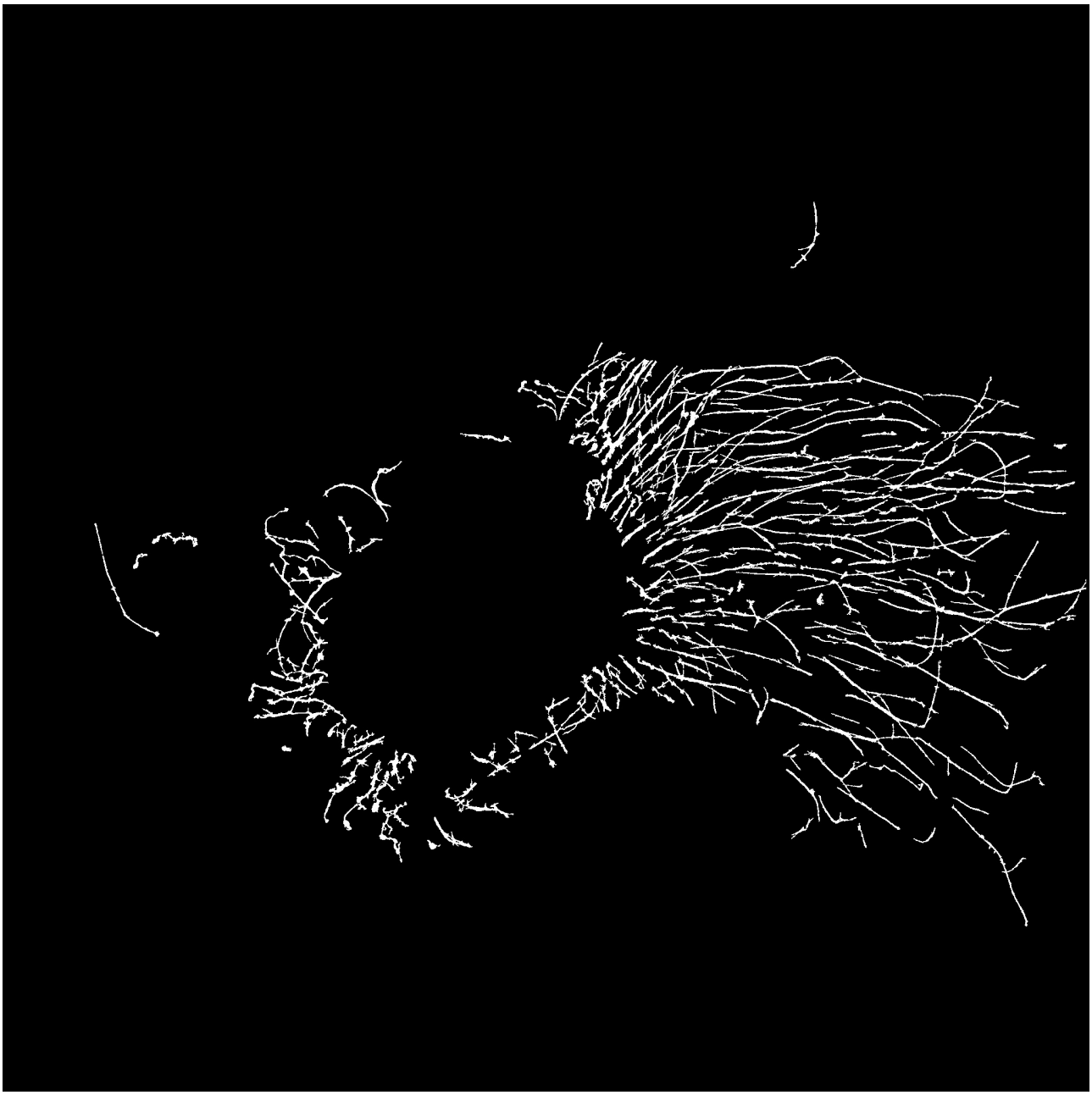
The final main operation applied to this image before the quantification begins is **Component Analysis**.

Component Analysis of the Neurite Candidate Image

Once the masks are applied to the binary image to remove any pixels of either the DRG body, or those known to be outside the well and therefore not neurites, the image is then processed to identify Connected Components (https://docs.opencv.org/3.4/d3/dc0/group__imgproc__shape.html#ga107a78bf7cd25dec05fb4dfc5c9e765f). Each component corresponds to a block of directly connected pixels, of arbitrary shape or size. These identified components are then filtered by applying two criteria to preferentially segment out neurites rather than the noise speckles which can still be seen in the masked images above:

- Components must be of a minimum area.
- For components with aspect ratios near unity, the infill fraction must be less than a particular threshold.

The area threshold performs the bulk of noise removal, as the speckle noise corresponds to components of small area disconnected from any other larger components. The second criterion is more nuanced, and intends to separate the larger speckle noise from the smaller neurite signals. For neurite components, we expect them to have a large aspect ratio. If the aspect ratio is close to unity, this is generally due to overlap of multiple neurites, and therefore we expect a meaningful amount of blank space between the individual neurites. For the larger speckle noise, on the other hand, we expect densely packed pixels and an infill fraction closer to unity. Applying these filters to the identified components leads to the following image:



Here, we can see that some of the faintest neurites have been lost, along with essentially all of the speckle noise. This is considered a worthwhile trade-off for the simplicity of the operation, with some potential improvements to be gained from tuning the specific threshold values.

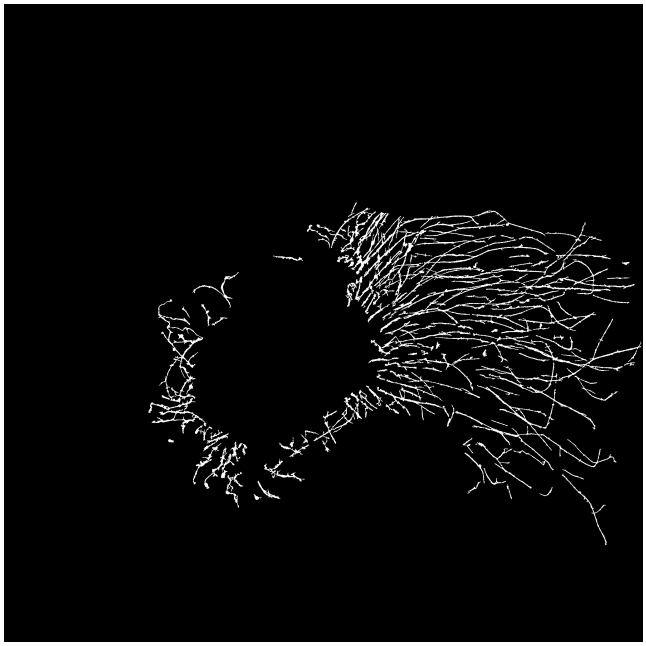
Optional Manual ROI Selection

As a final, optional, processing step, components can be masked out by drawing bounding boxes around any components which should be removed. This is opt-in through the use of a command-line argument, and requires the user to draw polygons using the mouse around the components they would like to **remove**. In

the case of the candidate image above, this would be useful to remove the components on the far left and top of the image, as these don't appear to correspond to actual neurites. This process generates yet another exclusion mask, which is applied to remove the selected regions.

Polygon Exclusion Mask

Resulting Neurites



In the case where the manual ROI selection is not enabled, the polygon mask defaults to including all pixels and is effectively skipped.

Script Result Generation

The script generates two sets of outputs on successful execution.

- A population distribution of neurite lengths within the image
- A population distribution of neurite orientations within the image

Each of these results are extracted from the result of the manual ROI selection image. How these metrics are extracted is described in more detail below.

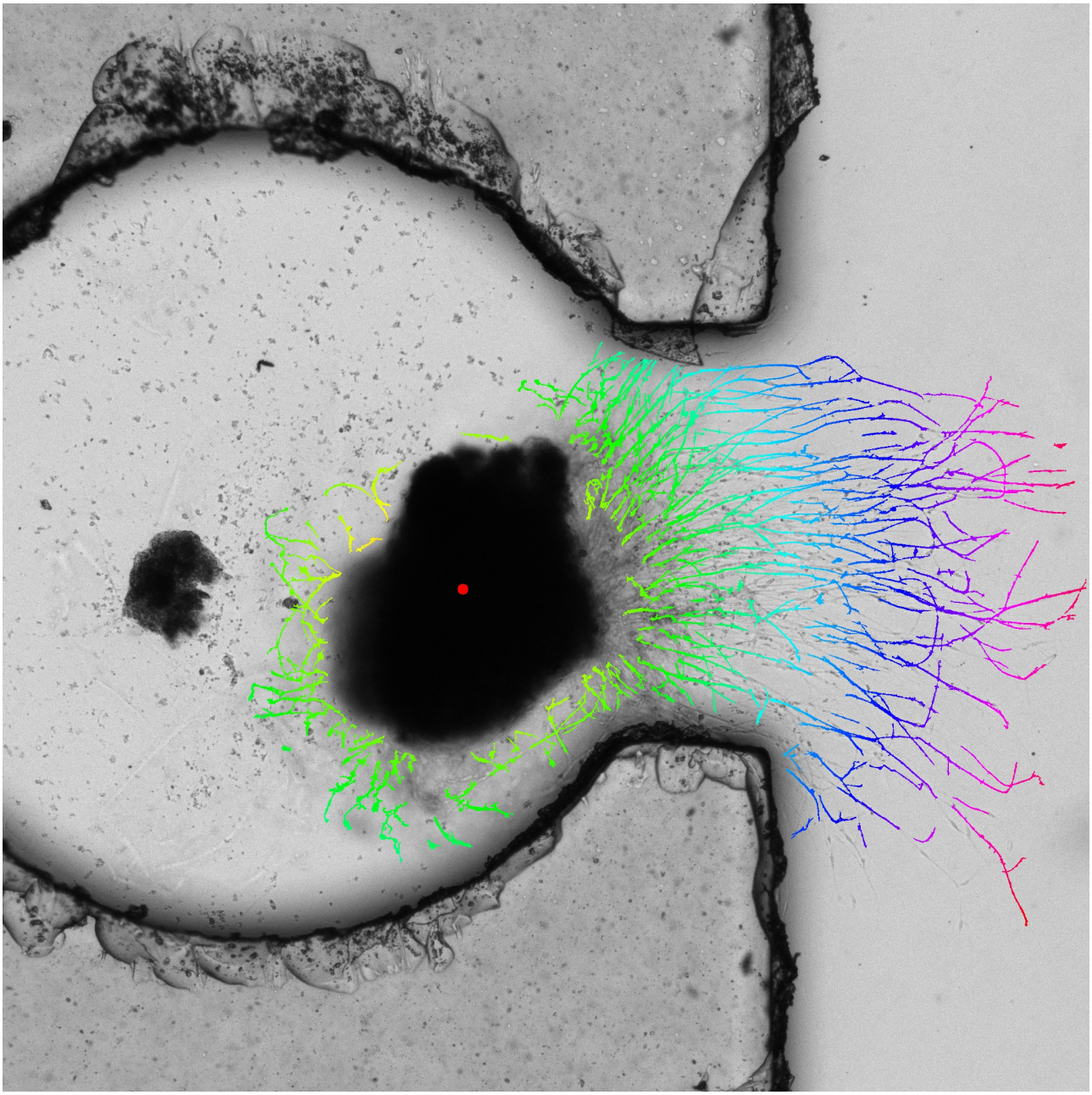
Neurite Length Extraction

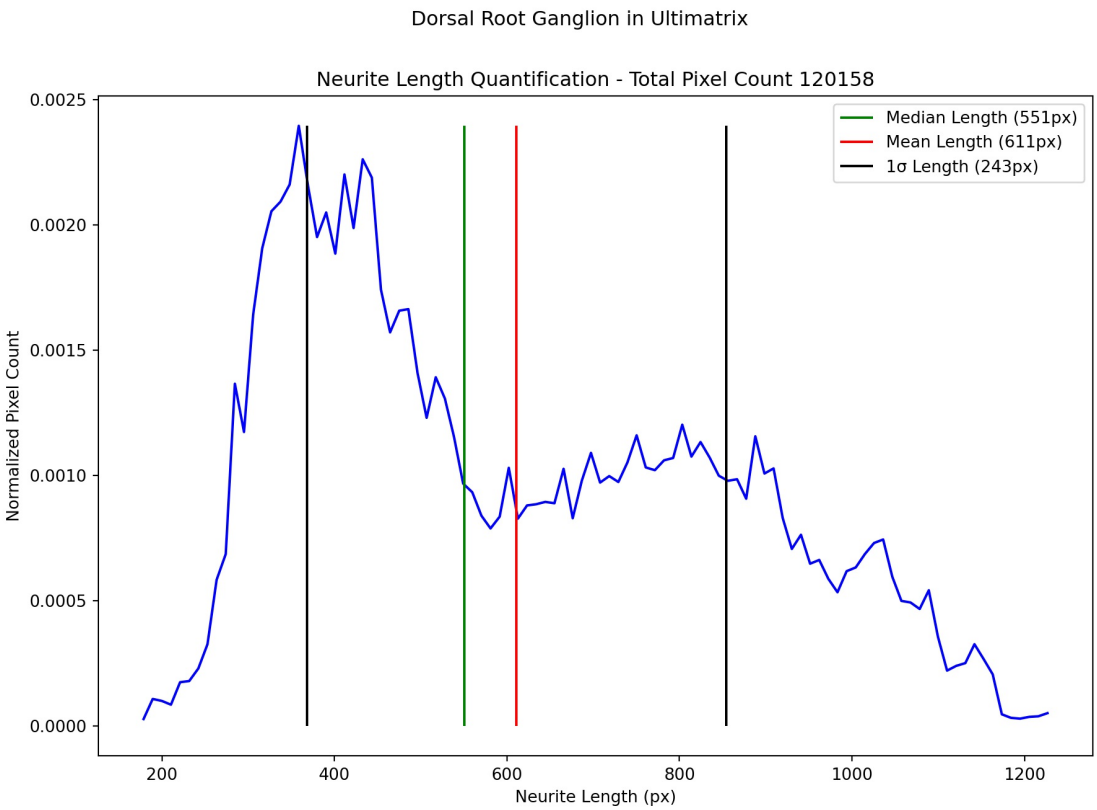
Before discussing how neurite lengths are extracted from the image, what exactly is meant by neurite length must be clarified. While the ideal metric would involve tracing and disambiguating individual neurites for their entire length from the interior of the DRG body to their furthest extent, this is not remotely feasible. As a simpler, yet similarly informative, alternative, the script instead considers each neurite to literally be composed of small segments. For a neurite to grow to some length **L**, it must definitionally grow past a length of **l ≤ L**.

The metric computed by the script is therefore defined by counting the number of pixels identified as neurites at each distance from a defined *Origin* point. This matches the desired behaviour of the metric, where longer neurites will correspond to more *signal* at higher distances. This does imply that even with large growth, a meaningful amount of *signal* will still be present at small values. Rather than looking at the mean or median length of this metric, what is more useful is to examine the shape of the tail of the distribution.

Finally, the *Origin* point is defined to be the centroid of the DRG body, as we have no knowledge of where a particular neurite extends from. Rather than picking points on the edge of the DRG body, the centroid appears to offer the least bias in case the neurite originates elsewhere within the mass.

The outputs of the script for this metric are shown below:





The first figure shows a visualization of the identified lengths, with length mapped onto colour. This image in particular is only useful for manual interpretation of the identified neurites, and provides a quick and understandable visualization of where the neurites are identified to be.

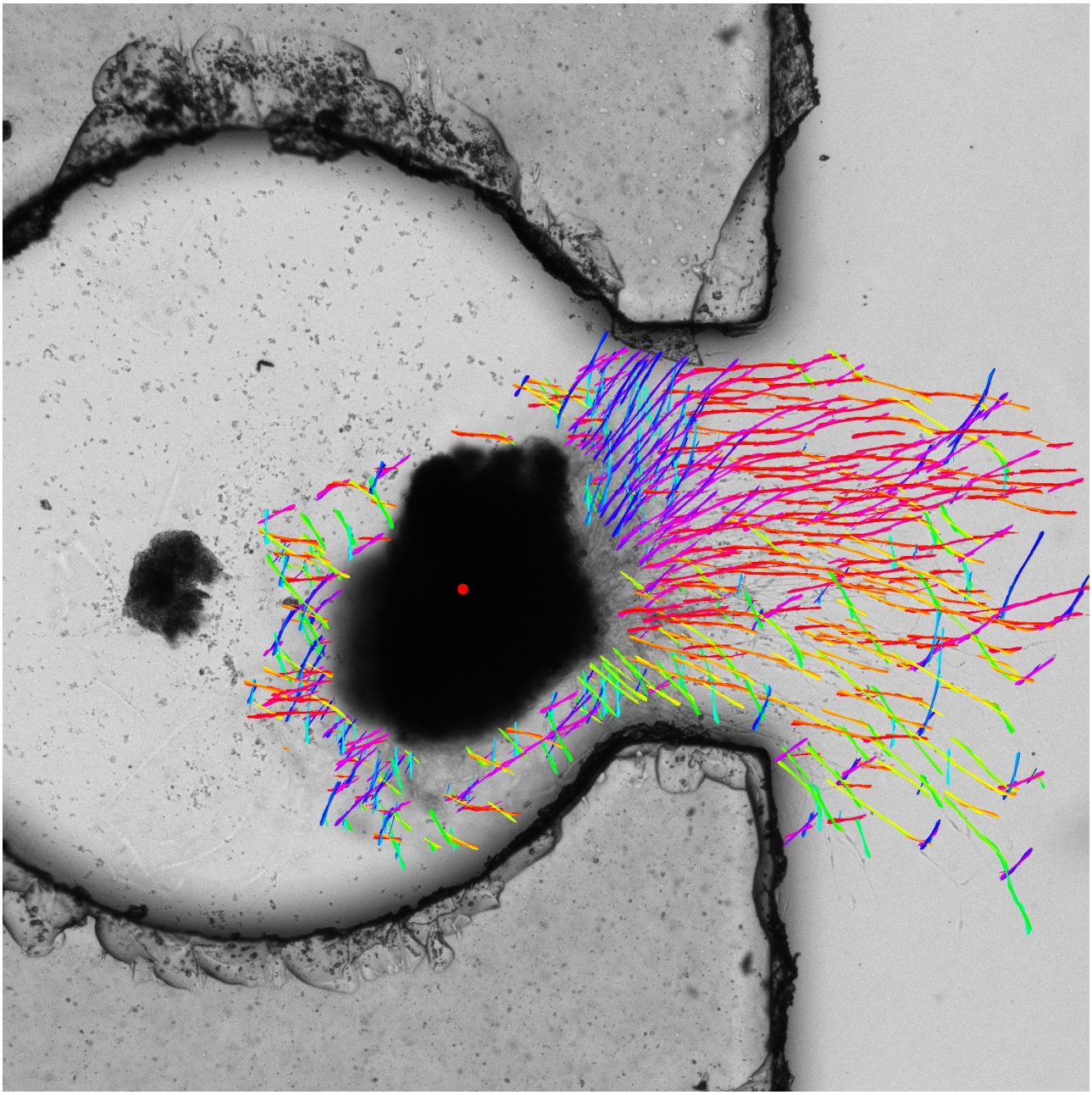
The second figure is the plot of identified neurite lengths over the image. As can be seen, there is a peak at around 350-400 pixels, which corresponds to the light green colour in the visualization. Then, the distribution is relatively flat out to approximately 900 pixels, matching what we see out to blue-purple colour. Finally, the distribution tapers off as neurites end, shown by the smaller amount of red in the left image.

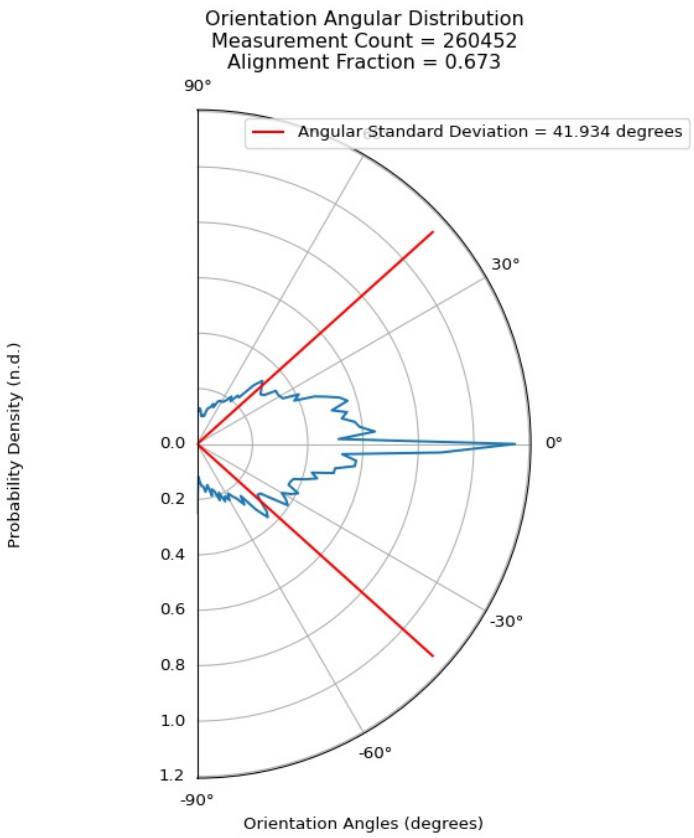
NOTE: In addition to these images, the script is being extended to provide the raw data behind these figures in a format suitable for alternative processing or visualizations. This is not yet complete, but is a work in progress.

Neurite Orientation Extraction

The second quantification metric involves determining the orientation of the neurites within the image. As a major goal of the project involves the demonstration of neurite growth aligned to the direction of pre-oriented rods, we want to be able to show that the neurites and rods are oriented in the same direction. As noted before, this process starts with the same image from the manual ROI selection step.

The exact algorithm used for the orientation detection is beyond the scope of this document, and can be found within the document "Running the Alignment Script.pdf". In short, this algorithm convolves an elliptical kernel over the image, varying the orientation of the ellipse, to identify the most prominent orientation of each pixel of the image. These orientations are then collected and the deviation from the mean orientation is plotted as a distribution to quantify the spread of angles. The original image is also colour-annotated for visualization purposes, and the results are shown below:




Orientation Angular Distribution
Measurement Count = 260452
Alignment Fraction = 0.673
Angular Standard Deviation = 41.934 degrees
90°
30°
0°
-30°
-60°
-90°
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Probability Density (n.d.)
Orientation Angles (degrees)