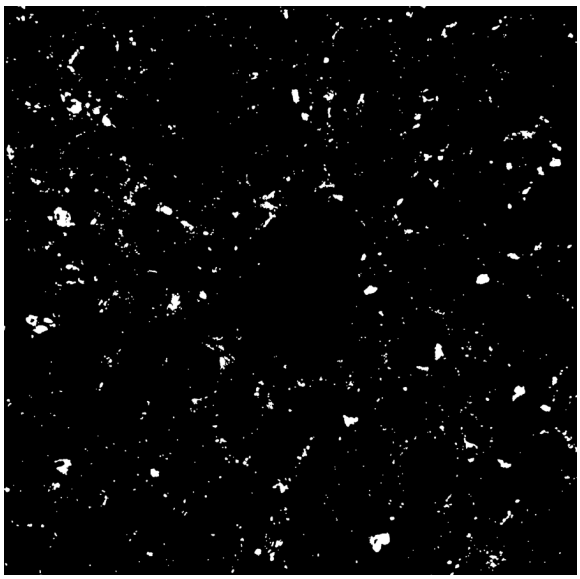
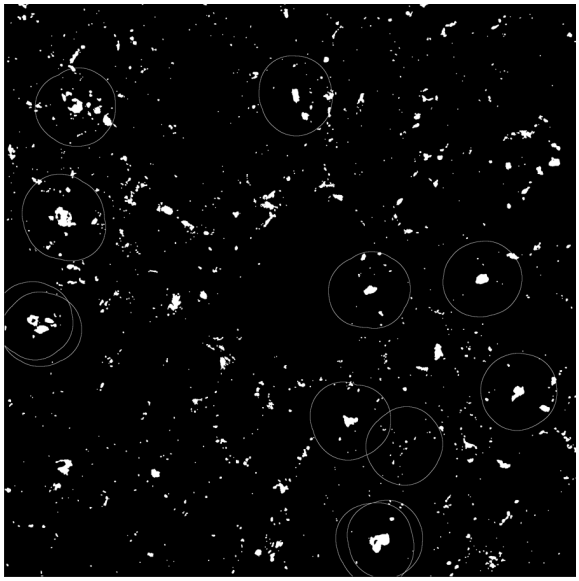
Brief Explanation of the Journey (Informal)

The starting point idea was to extract the green and blue channels, similar to the previous Protein Quant project. The green channel is the non-background matter (channel 1), and the blue channel is the nucleus channel (channel 2). Initial approach was to do cell proxy based on a nucleus on the blue channel, and count the green area inside the proximation area.

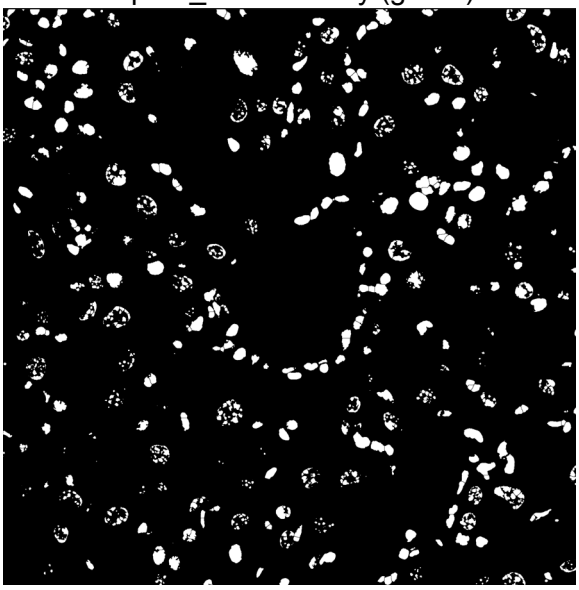
The first scripts relied on ROIs (Region of Interest). ROIs are very useful. Their algorithm is, for a selected point, what is around this point that is important to the point? At this time, the script was using “Otsu Dark” for thresholding, which didn’t result in good threshold masks, especially for the green areas.



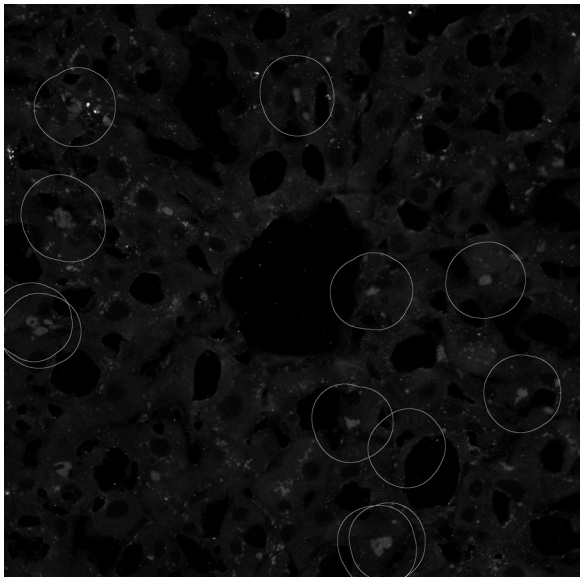
Shkbp1-2_a: Cell Proxy (green)



Areas That Were Found Interesting



Shkbp1-2_a: Nuclei Mask (blue)

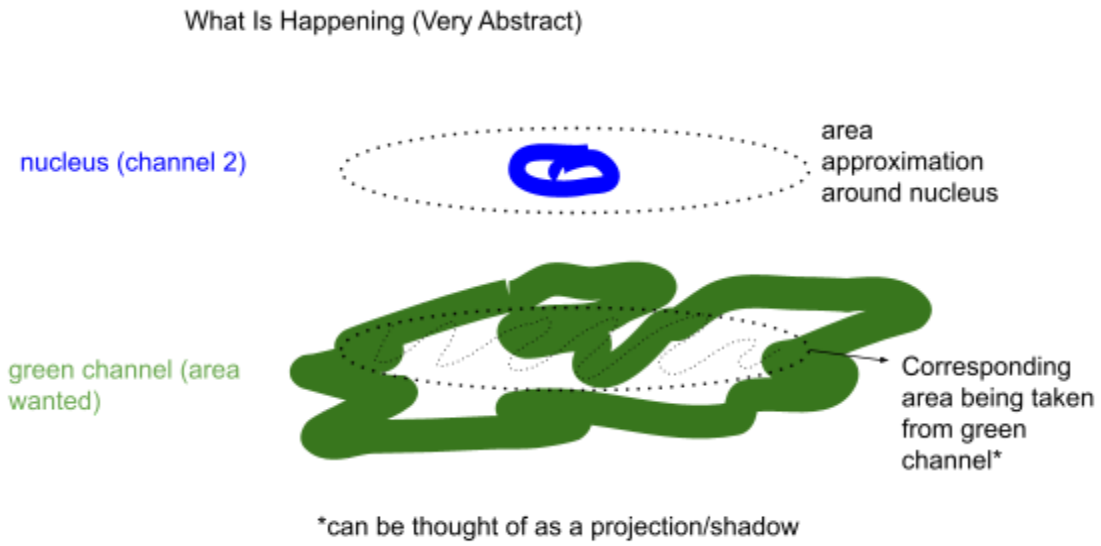


Found Areas on The Greyscale Overlay

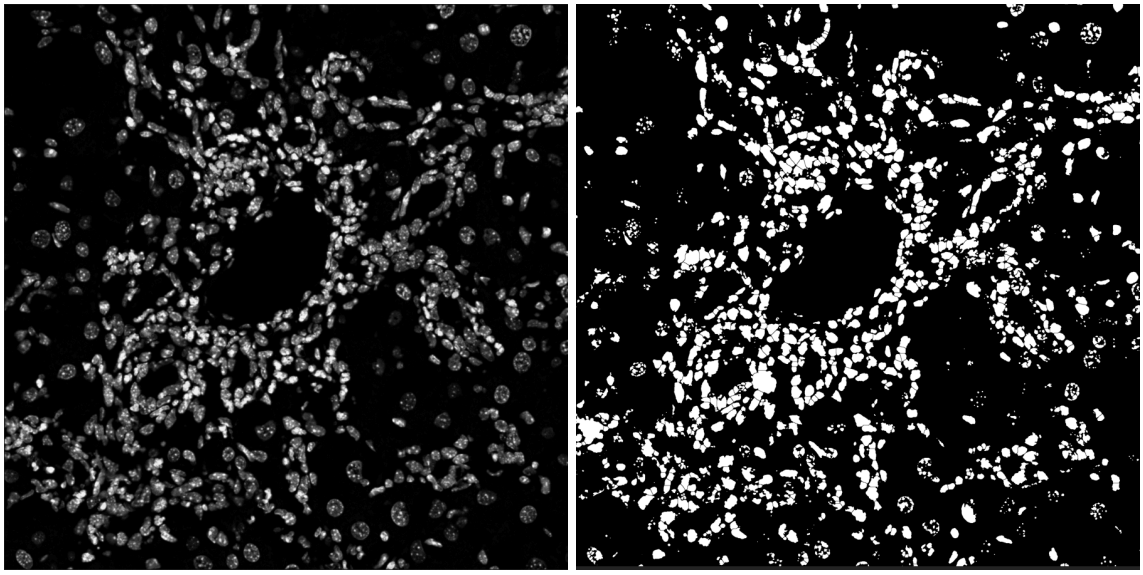
The difference between “Otsu” and “Otsu Dark” is:
Otsu: light object on dark background
Otsu Dark: dark object on light background

The reason for using Otsu Dark at the time was because I was considering what the generated mask would look like instead of the .czi itself. This is fixed in the next iterations.

Looking back at the images with the circle, it is seen that they’re not placed randomly. They are “enlarged” ROI nuclei, and the reason why there are so few is because the noise-derived nuclei are being forcibly removed. The circles more-or-less are centered on lighter regions of the image. The most important discovery of this, and the reasoning behind purposefully not calculating the areas correctly is because the actual test was seeing if the surface areas were being calculated correctly. But there is overlapping, and the areas shouldn’t be round circles. These mistakes are the beginning of the next iteration.

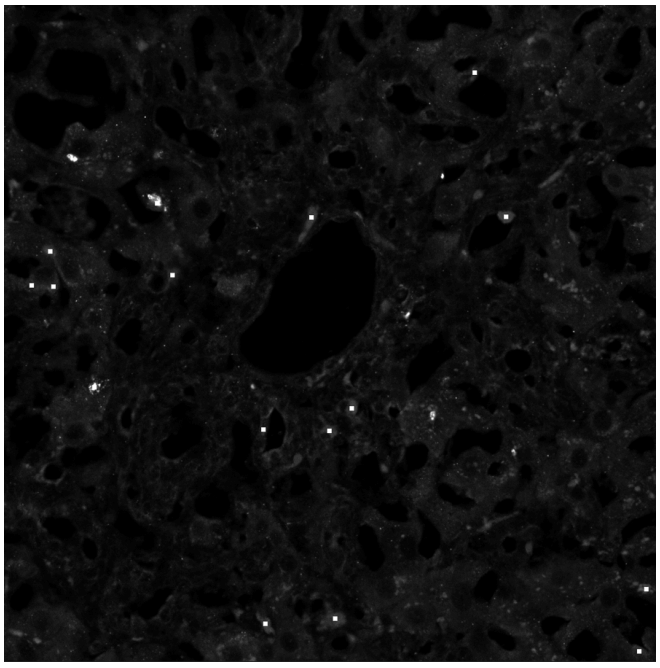


The .czi files aren’t 2D (because of the channels). Therefore, simplifying projections are the best option for accurate outputs. When the mask is being created, the intensity is turned up to the max to get as many details as possible (no gradients in area-calculation mask). Background subtraction and Gaussian blurring are applied to reduce noise, followed by thresholding to create a binary mask. Next, the blue channel is processed to identify nuclei locations. Artifacts are removed, holes inside nuclei are filled, and each nucleus is detected as an individual particle. For every detected nucleus, its averaged center is calculated. These centroids serve as “seeds” that represent the centers of cells, which is more accurate and efficient for area proxy estimation. The green channel is also processed to generate a clean binary mask. Background subtraction and blurring are applied, followed by thresholding. Noise is removed and smooth the mask, and small particles are filtered out so that only meaningful green signal remains. Please look at the following images that show an application of this paragraph.



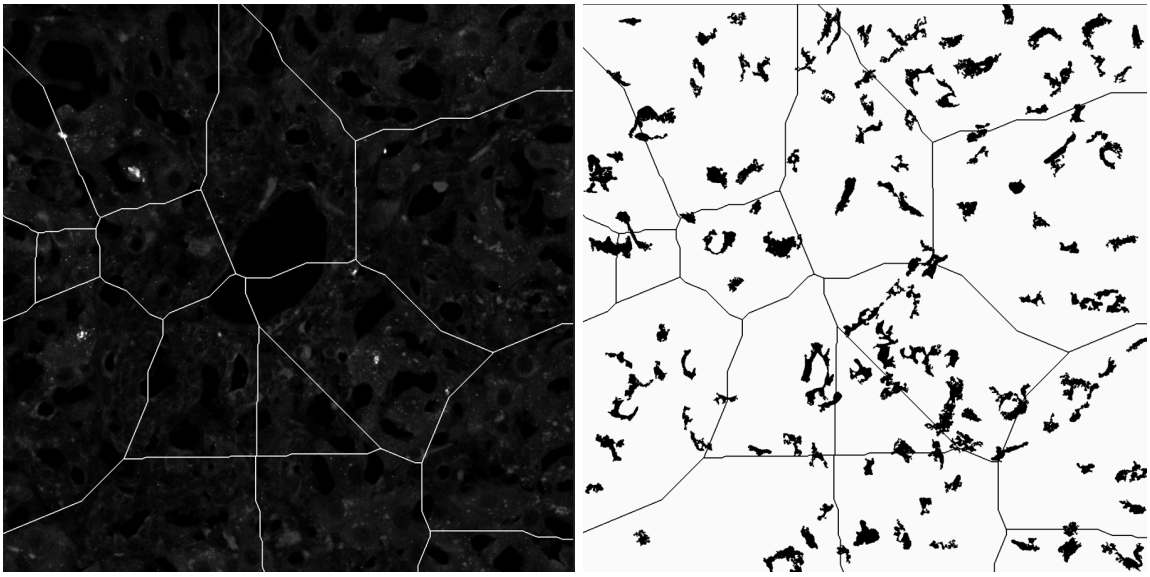
Shkbp1-3_e: Grayscale Nucleus Channel vs. The Threshold Mask

At this point, there is some quality control. In certain iterations of the script, some selected nuclei are even removed if there are a few that are too close to each other. Since the computer can't automatically differentiate between a nucleus, binning and scaling are used for verifying cell positions.



Shkbp1-3_e: Tiny, White Squares Are The Seeds (example)

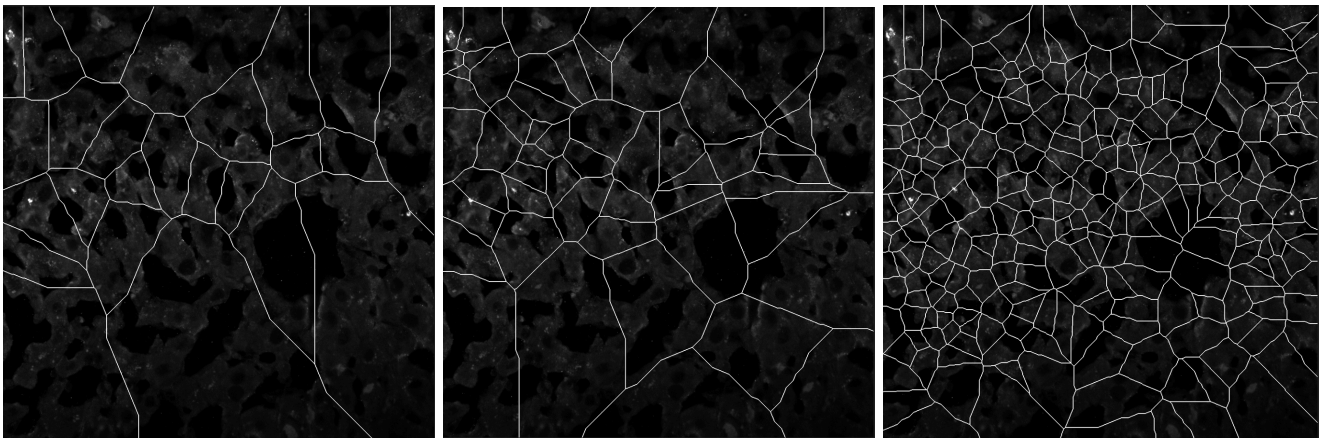
Using the final set of seed points, a Voronoi tessellation is computed. Starting simultaneously from all seed locations, the algorithm assigns every pixel in the image. Each pixel goes into the region of the seed that it is closest to. The next page has the Voronoi map for the given seed image.



Shkbp1-3_e: Rough Voronoi Mappings (example)

Voronoi_ID	SeedX	SeedY	RegionArea_px	GreenArea_px	GreenAreaFraction_%
1	737	105	103978	13110	12.608

For each region, the total area and the number of green pixels are counted. Percentage of green area within each Voronoi territory is calculated. All measurements are saved to a CSV file.

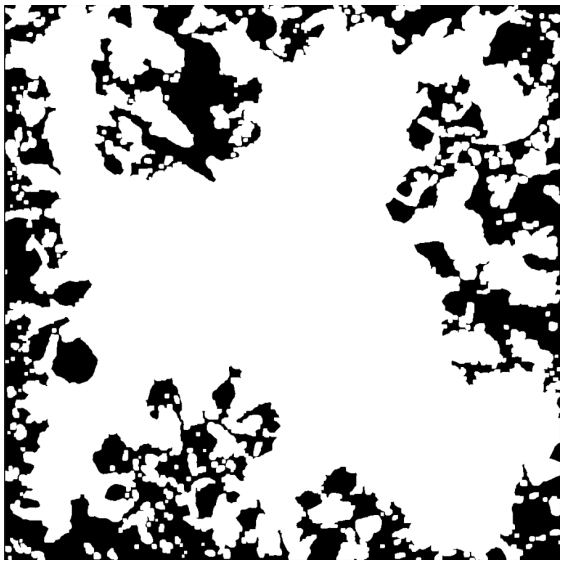


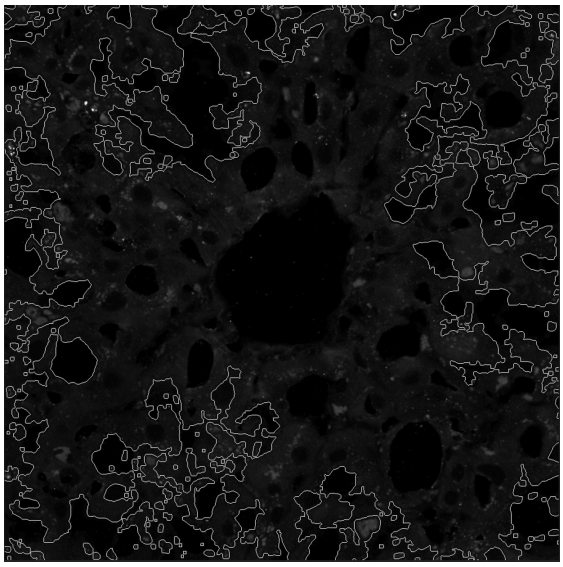
Shkbp1-3_c: Even with the same seeds, different thresholds lead to different regions

Issues with this grouping system:

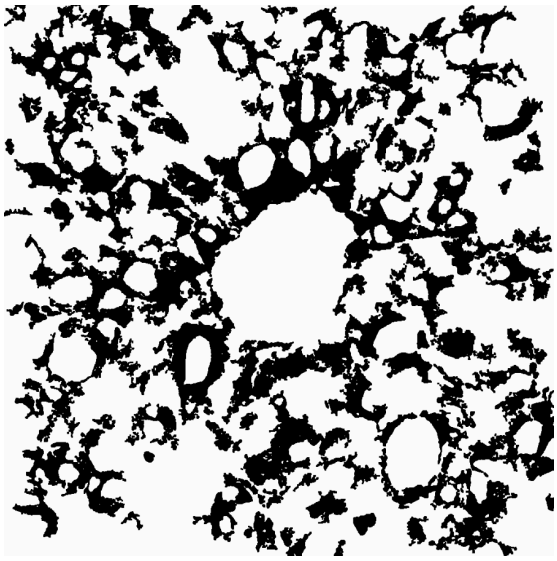
- big polygon regions cutting across unrelated structures
- lots of random empty space

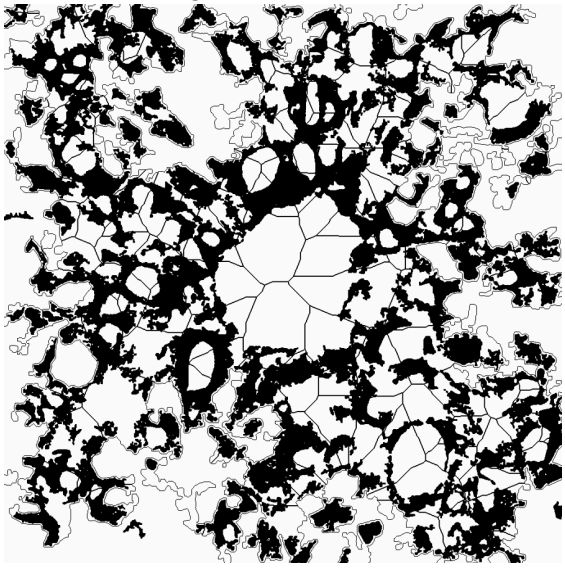
To fix these, first the masking was changed. The main idea for masking is the same as previously explained, but there are more meticulous events happening in the background that lead to better groupings. This is that instead of focusing on the green area (it has gradients, and hard to group exactly even for humans) the background black area is put into a domain mask to optimize removing artifacts. I'm purposely including the first outputs of this approach for Shkbp1-2a for comparison between the initial approach and this version.





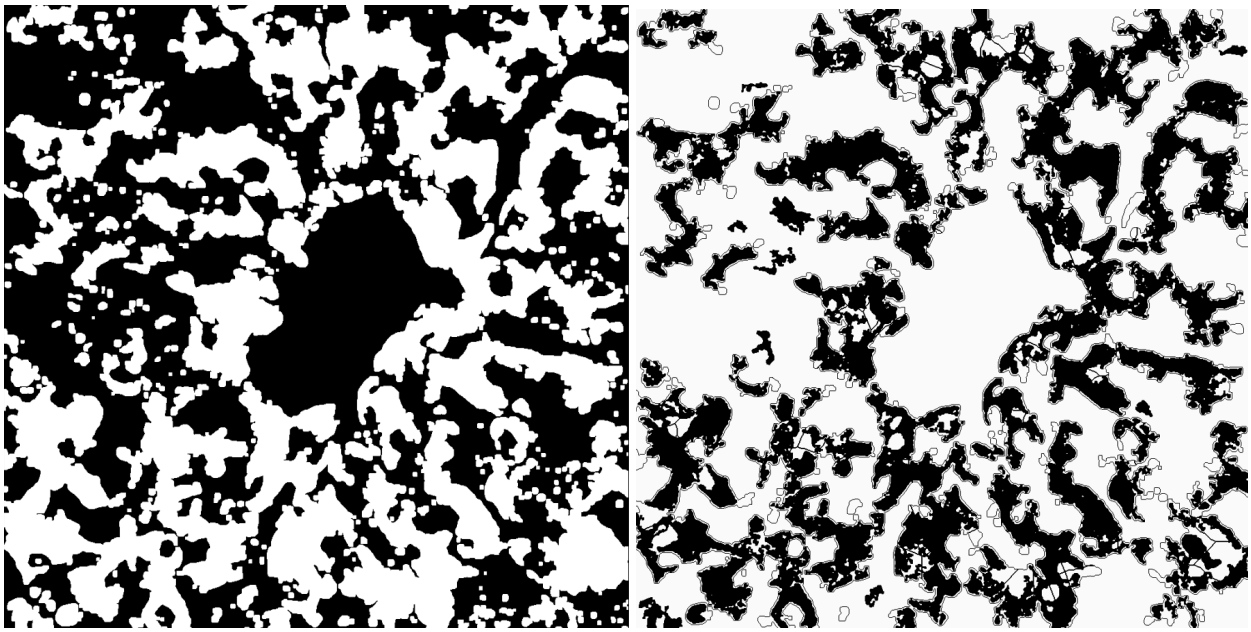
Shkbp1-2_a: Definitely Black Areas (Domain Masking) Selected Domain

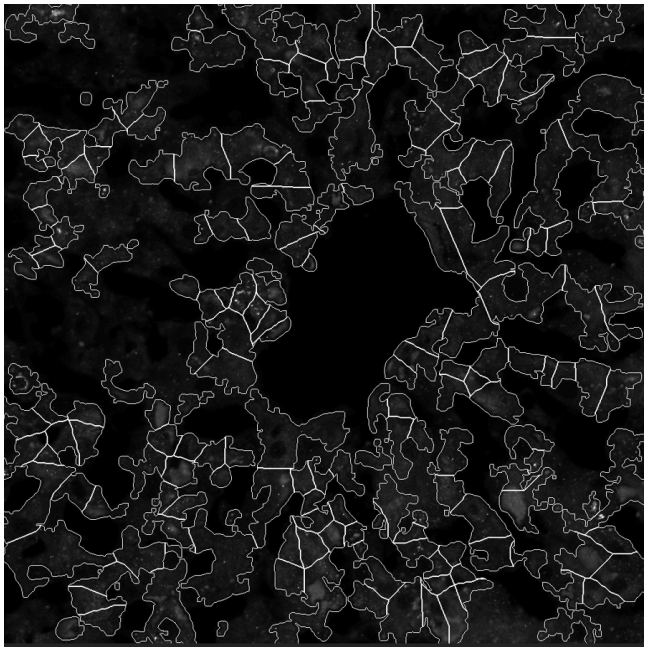




Shkbp1-2_a: Better Green Mask With Thresholds Voronoi Only In Domain

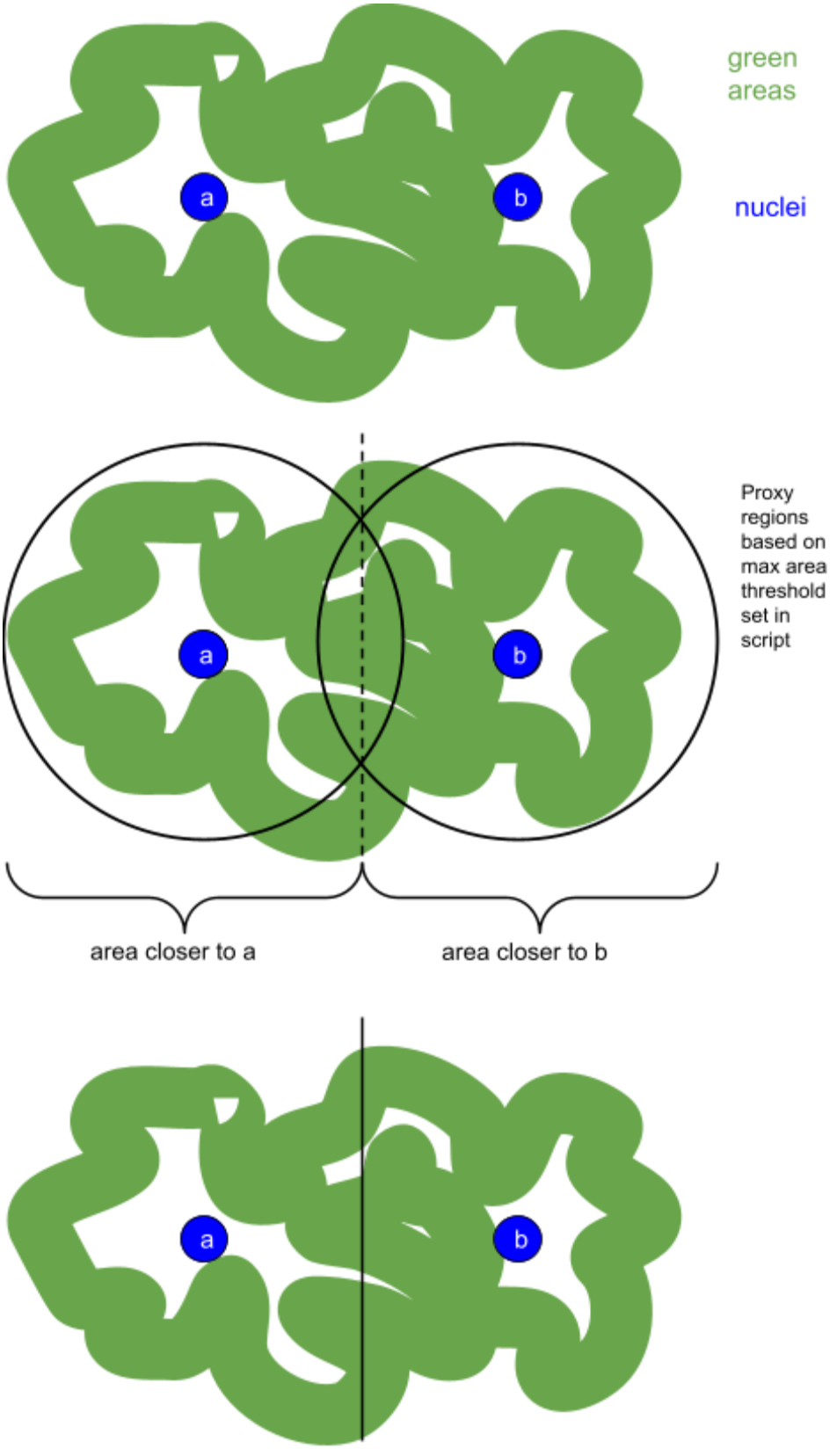
The issue right now is seen in the fourth image above. The Voronoi regions are too rough. This is because until this point, the Voronoi was working based on nuclei only. Therefore, while non-green regions weren't being included in the area calculation, the grouping couldn't "see", metaphorically, the greens. So, in the next iteration the domain is restricted even further.





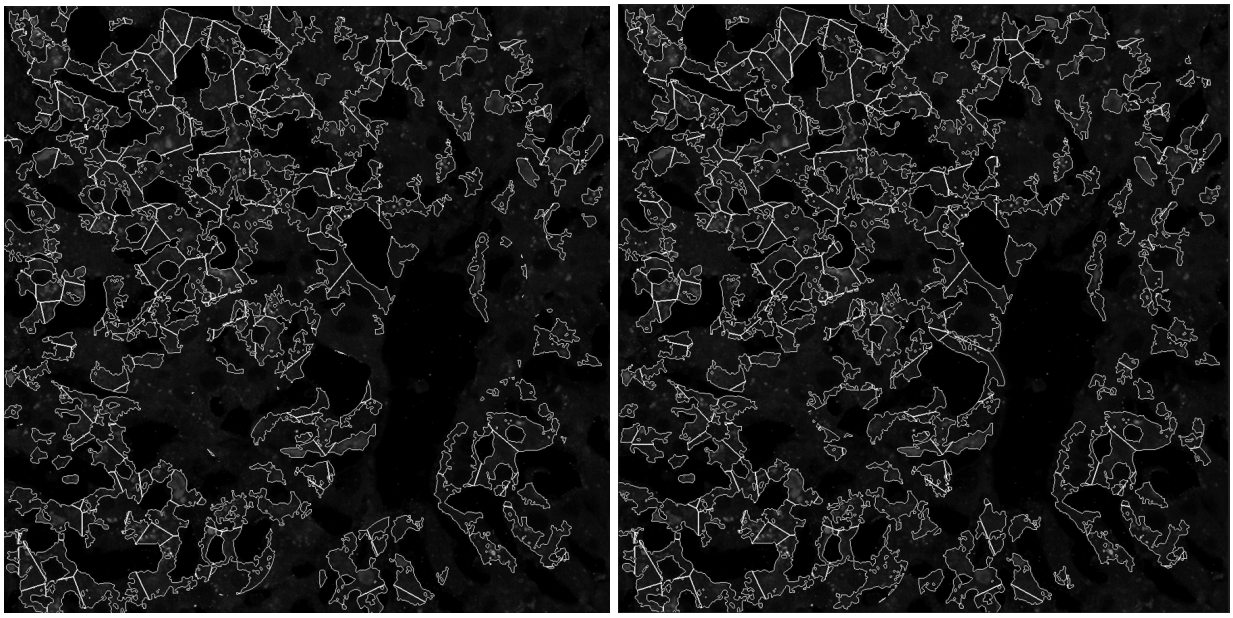
Shkbp1-3_f: Better Domain Masking Chosen Greens Overlay With Greyscale

Important Math Information: Even when the domain boundaries are organic, the lines of separation inside domains are still not-organic looking. This is how the math process happens, and the lines look even straighter than the previous full Voronoi graphs because the regions are so small. Here is a diagram showing how Voronoi calculates for two arbitrary points:



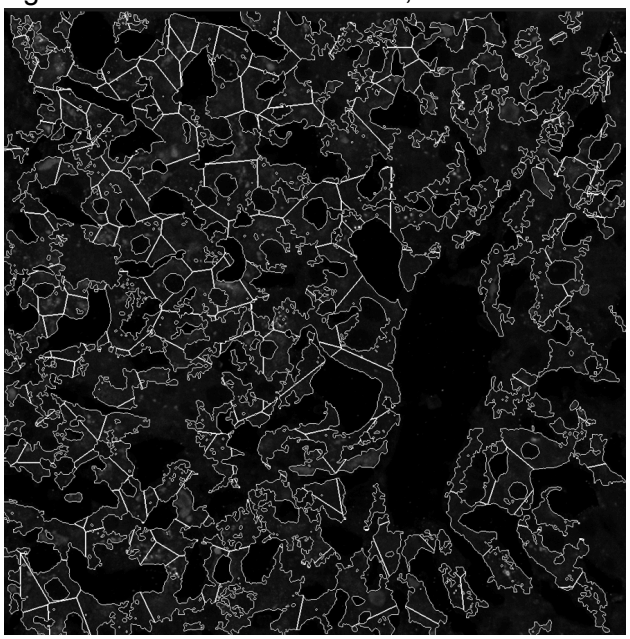
Non-organic straight line seperating “a” and “b”

From this point forward, the thresholds are changed to find the best ones for these computations. The following images are of similar Shkbp1-3_d, that have slightly different parameters. In the following images, the maximum areas are changed (increased from 80 to 170). As you can see, most of the organic boundaries aren't changing. This is because domain masking isn't reliant on this argument.



Shkbp1-3_d: 80 pixels 170 pixels

The most important distinction is that it is no longer using "Otsu", instead "Li". Li is allowing for more generous regions when masking the green. Li allows for larger, and more organic areas. It's more susceptible to noise compared to Otsu, but for this set of images, the program is likely to judge gradients as noise. Therefore, Li fits better.



Shkbp1-3_d: Li instead of Otsu

Some technical notes for reference:

- The program won't allow for batch processing. Even when images are being read from a dedicated folder, each open file is being counted as a separate IO operation.
- Runtime for ~20 files takes around 2 minutes.
- Two dimension simplification doesn't change Voronoi math logic.
- Don't turn on Watershed unless absolutely necessary.