Week of 10/16 Deliverables

Team cobalt

Last week's goals

- Replicate good registration results from daniel
- Survey papers and write pseudocode for 1 of each of the following in cell detection workflow:
    - Preprocessing
    - Thresholding/binarization
    - Edge detection/blob detection
    - Clustering /refinement
    - Comprehensive survey: https://www.ncbi.nlm.nih.gov/pubmed/26742143
- Implement algorithm in this paper - Hessian-based DoG for blob detection

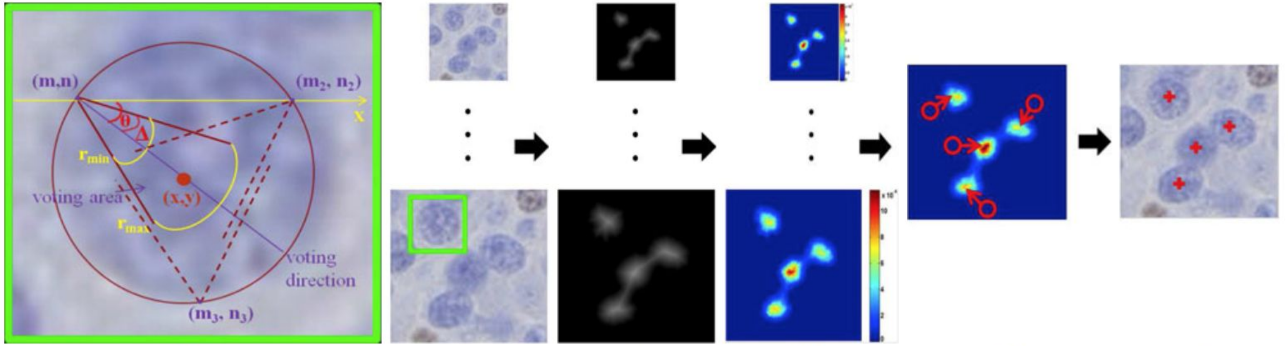
Survey of different cell detection methods

- Reviewed
  - Distance transformations
  - Morphological operators
  - HIT/HAT
  - Hough transforms
  - LoG filtering
  - MSER detection
  - Radial symmetry based voting (RST)
  - Supervised learning (i.e. SVM, random forest, CNN)
  - Review paper: https://www.ncbi.nlm.nih.gov/pubmed/26742143
- Findings: http://bit.ly/2z7Jp39
  - (Look at table of performance of algorithms)

- Most promising methods:
  - LoG, MSER, RST
- LoG = fastest and most well suited for big data
- RST and MSER = more accurate and robust but slower

- Papers of interest:
  - Automatic Ki-67 counting using robust cell detection and online dictionary learning.
  - Detecting overlapping instances in microscopy images using extremal region trees
  - Automatic Nuclei Detection Based on Generalized Laplacian of Gaussian Filters
- Supervised learning is also one of the best performing. Should be visited if the opportunity with data presents itself

Region-based hierarchical voting in a distance transform map



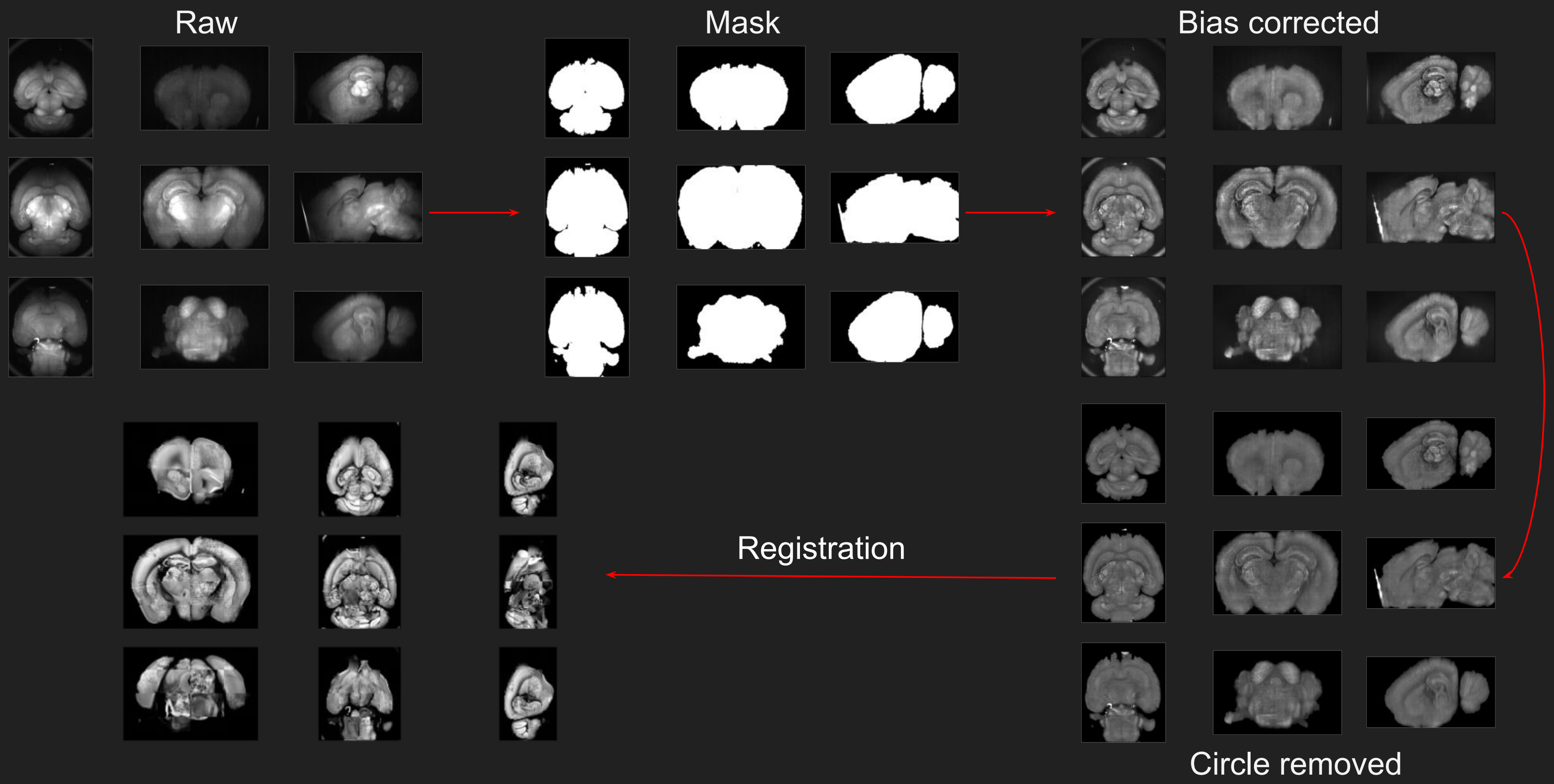
(a) (b) (c) (d) (e) (f)

1. Calculate the euclidean distance map $C_\ell(x, y)$
2. Calculate the cone shaped voting area $A_\ell(x, y)$ for each pixel
3. For each pixels (x, y) in each layer
   (a) Count the number of pixels in its neighborhood S with voting areas that include (x, y)
   (b) Compute the gaussian kernel at this point
   (c) Set the product of the gaussian kernel, count from (b), and EDT as its confidence map value
4. Run mean-shift clustering to calculate the centers.

$$V_\ell(x, y) = \sum_{\ell=0}^{L} \sum_{(m,n)\in S} I[(x, y) \in A_\ell(m, n)] \;\; C_\ell(m, n) \;\; g(m, n, \mu_x, \mu_y, \sigma) \qquad (2)$$

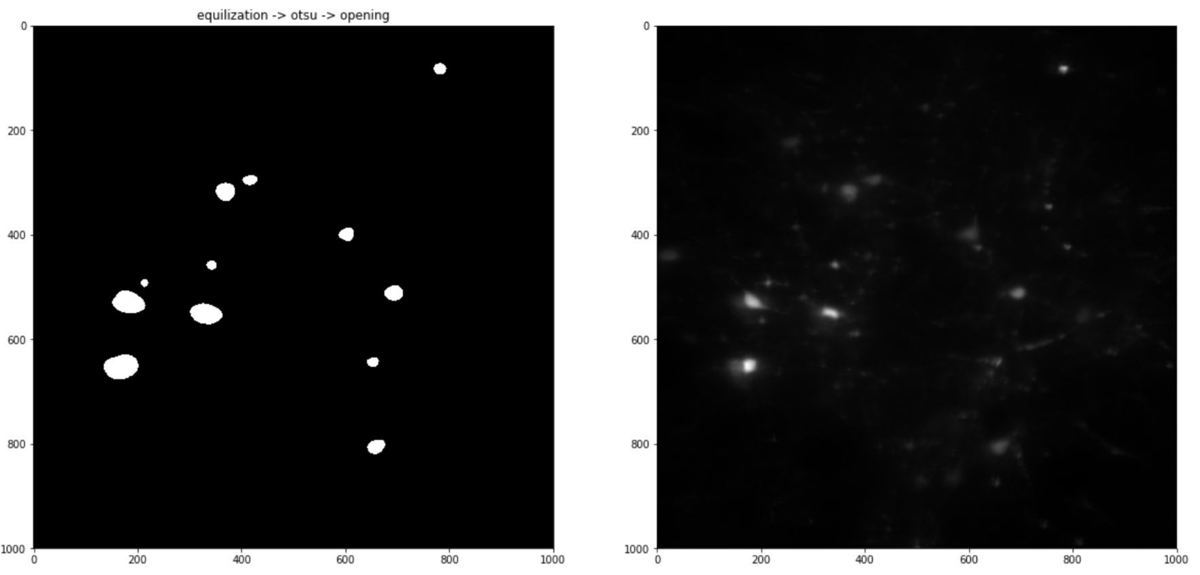
- Wrote pseudo code (will implement next week)
- Math description: https://github.com/NeuroDataDesign/clarity-f17s18/blob/master/docs/jyim6/neurodata-algorithms.pdf
- Paper: https://www.ncbi.nlm.nih.gov/pubmed/24557687

Registration - notebook



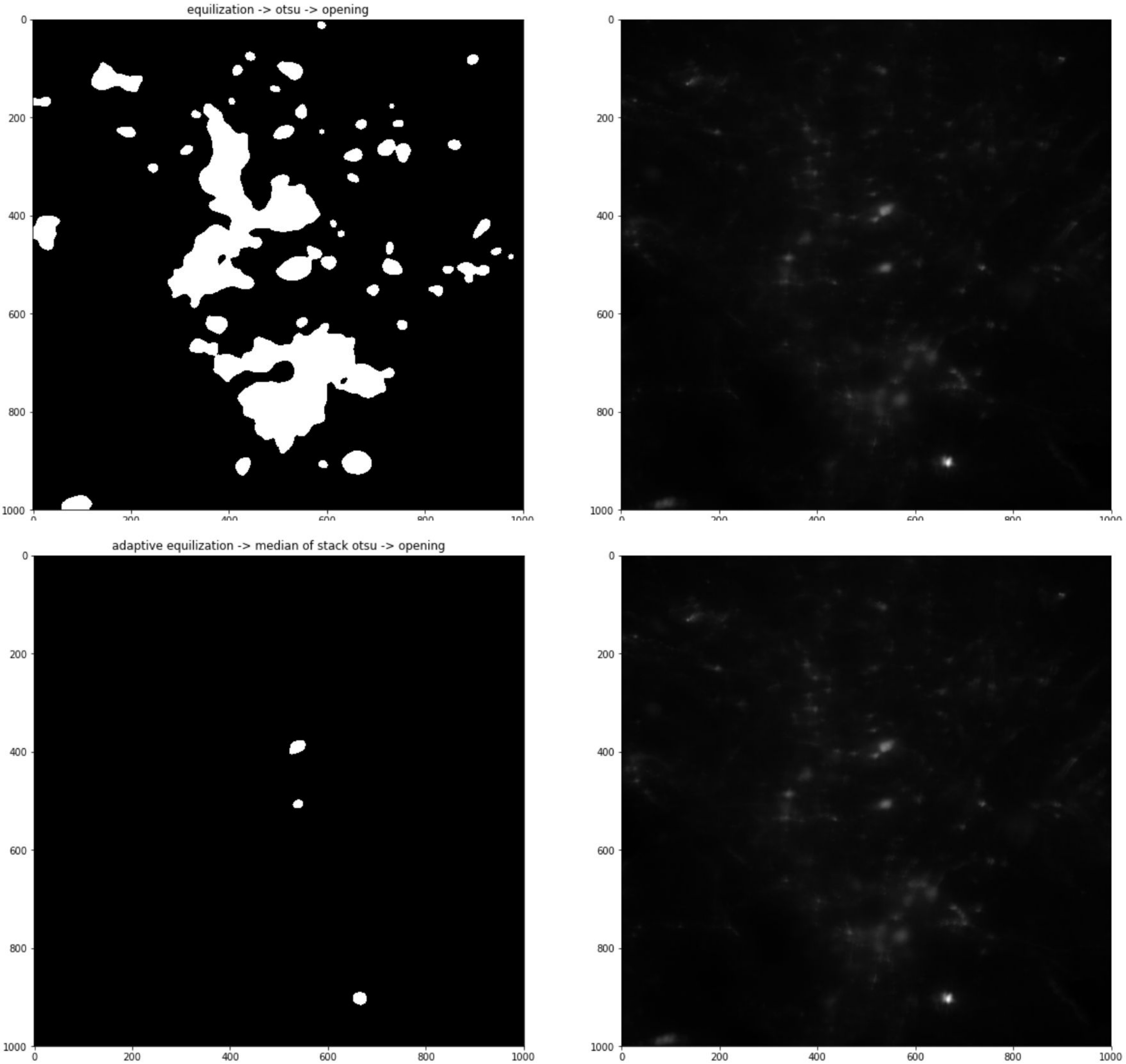
Evaluation of Thresholding and Binarization Preprocessing Techniques

- 2 scenarios
  - Image histogram is well spread
    - Techniques work well
  - Image only consists of low intensity values
    - Not sure this needs to be equalized because these low intensity values could just be noise, and not many actual cells are in the slice



Adaptive equalization > otsu > erosion > opening

Works well for images with brighter points



Same process, but for a low intensity image.

- Binarization picks up a lot of noise

Median of otsu thresholds for each of the slices in the subvolume

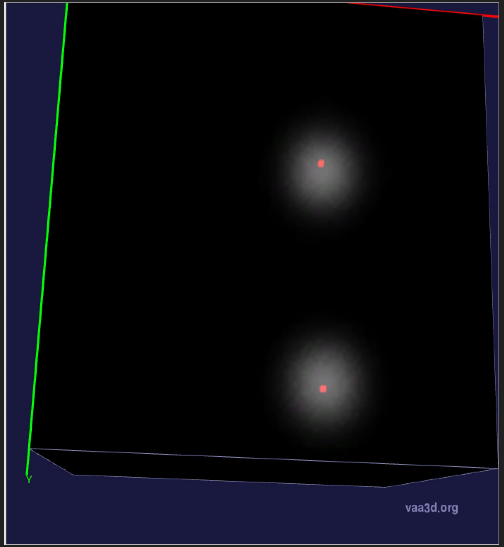
Hessian-based DoG - Blob Detection

1. Perform scale invariant Difference of Gaussian (DoG) transformation for the given image. For a given voxel *(x,y,z)* let this be *DoG(x,y,z;t)* where *t* is the scale parameter.
2. Find the hessian matrix of *DoG(x,y,z;t)* and call it *H(DoG(x,y,z,;t))*
3. At each voxel *(x,y,z)* find if the hessian matrix at that voxel is a negative definite matrix.
    a. This can be done by finding out the three leading principal minors *D1, D2 and D3* and checking if *D1 < 0, D2 >0 and D3 < 0.*
    b. If a voxel satisfies this condition, then that voxel and the 6-connected candidate is considered as a candidate blob region *T*
4. For each of the candidate region *T* find the R_T (Regional Blobness), S_T (Regional flatness) and A_T (Average intensity of the region) given by the below formula.

$$R_T = \frac{3 \times \left|det\left(H_T\right)\right|^{\frac{2}{3}}}{pm\left(H_T\right)} \qquad S_T = \sqrt{\lambda'^2_1 + \lambda'^2_2 + \lambda'^2_3}$$

5. Input the above three parameters to a variational Bayesian Gaussian Mixture Model (VBGMM) and segment into blob and non-blob regions using Bayesian (or MCMC approximation) Inference.

- Implemented - [Code]
- The notebook will be implemented next week along with the cell detection package



Hessian-based DoG - Blob Detection

Good things

- Low intensity blobs are identified with good amount of precision
- The post-pruning method to remove false positives is completely unsupervised unlike many blob detection algorithms

Bad things

- Hessian analysis is time consuming and took ~ 8 minutes for a 1000 x 1000 x 100 volume
- How bayesian inference is performed is not clearly explained in the literature

Key takeaway

Though we are yet to quantitatively evaluate H-DoG against different datasets, it looks like a promising algorithm for low intensity blobs and the post-pruning is unsupervised and effective

Next week

- Implement basic python package to perform cell detection (w/ LoG)
  - Input: Single subvolume of data
  - Output: csv of cell centroids, metrics using blob-metrics
  - Delivered on our github repo
- Quantitatively test ensemble method on simulated data
  - DoD: markdown file with results on simulated data (see here for template)
- Compile a list of preprocessing/noise removal methods traditionally used in LSFM
  - DoD: markdown file with current state-of-the-art for LSFM preprocessing and thresholding including links to papers