Automated Cell Segmentation in Julia

Rebecca Carlson, Kathi Hoebel, and Olivia Waring

Our Team

Becca
2nd year HST
Paul Blainey and Nir Hacohen labs
Broad Institute

Kathi
2nd year HST
QTIM
Martinos Center

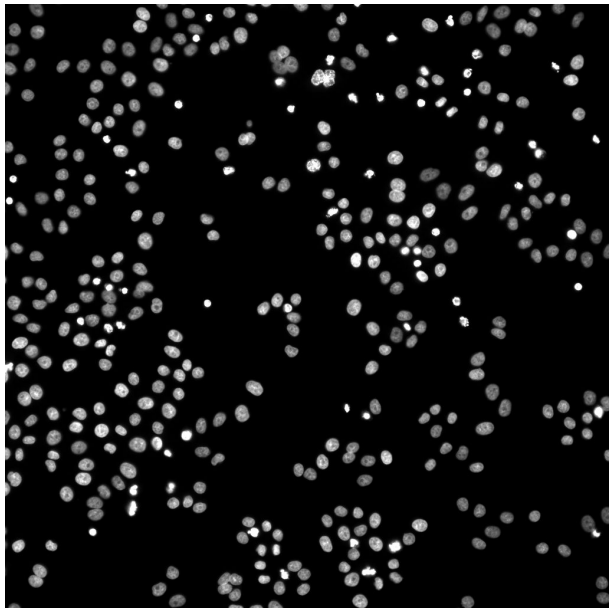
Olivia
2nd year HST
Bruce Walker Lab
Ragon Institute

Cell Segmentation

- Traditionally, images of cultured cells for biological research were both acquired and reviewed manually
- However, it is becoming increasingly common to acquire large datasets with **tens of millions** of cells
    - Our dataset has 900 images of >220,000 cells
- Segmentation on this dataset takes ~30 minutes as currently implemented in Python
- A dataset with 10 million cells (which could be a single large plate) would require **~1 day**
- Many other tasks in image processing are similarly time-consuming but independent across images: easily parallelizable

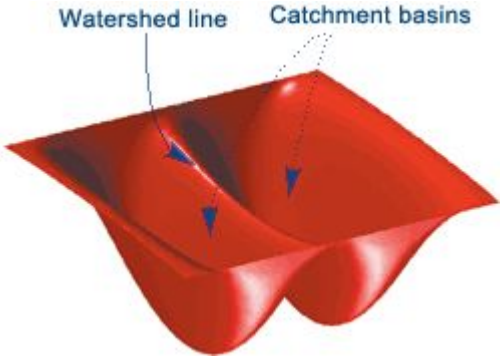
Our goal: **parallelize segmentation with Julia!**

Cultured cells with DNA stain



Marker-Based Watershed Algorithm

- Watershed algorithm: treats image as a topographic surface where bright pixels are peaks dark pixels are valleys
    - The algorithm floods from valleys: boundaries formed when water from different sources merge
- For cell segmentation, a marker-based version of the algorithm is commonly used
    - Cell nuclei act as markers from which topographic surface is flooded
- Add final thresholding step based on cell background to erode regions that meet (allow spaces between cells)



Parallelization

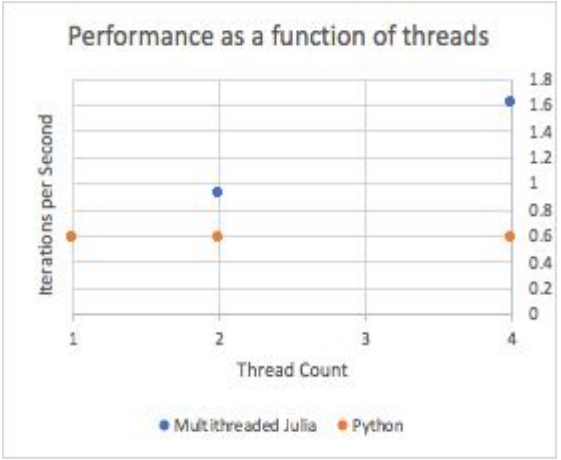
Three main approaches to parallelization in Julia:

- **Multi-threading**
  - Allows atomic operations to be conducted simultaneously
  - “Experimental” in Julia 1.0
- Multi-core or distributed processing
  - Divide operations across multiple CPUs
  - Limited by speed of the individual CPUs and speed of the memory access operations
- Coroutines (green threading)
  - Use “Tasks” to switch among multiple computations
  - Relies on a VM dynamic scheduler to allocate resources to particular tasks at different times

Performance Analysis

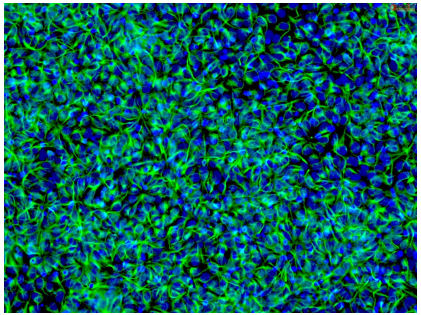
To process 900 1024x1024 .tif files:

Algorithm	Runtime
Native Python Code	25 min 47 sec, avg 0.58it/s
Julia Watershed - Unparallelized	25 min 36 sec, avg 0.59it/s
Julia Watershed - Parallelized (multi-threading, 4 threads)	9 min 19 sec, avg 1.61it/s
Julia Watershed - Parallelized (multi-threading, 2 threads)	16 min 3 sec, avg 0.93it/s
Julia Watershed - Parallelized (distributed)	Future directions

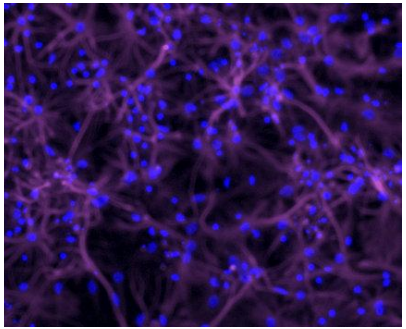


U-Net Based Segmentation: Motivation

- Marker-based watershed still has segmentation errors and requires a nuclear stain for the marker, which is not always available
- Works well enough for many cultured cells, but not well in tissues, where cells are three-dimensional
  - In addition, some cells have unusual shapes or imaging preparations

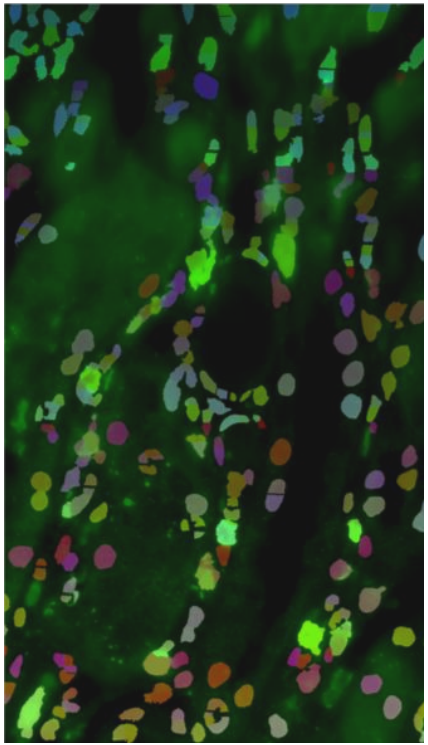


Dense cells

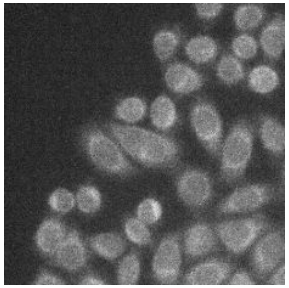


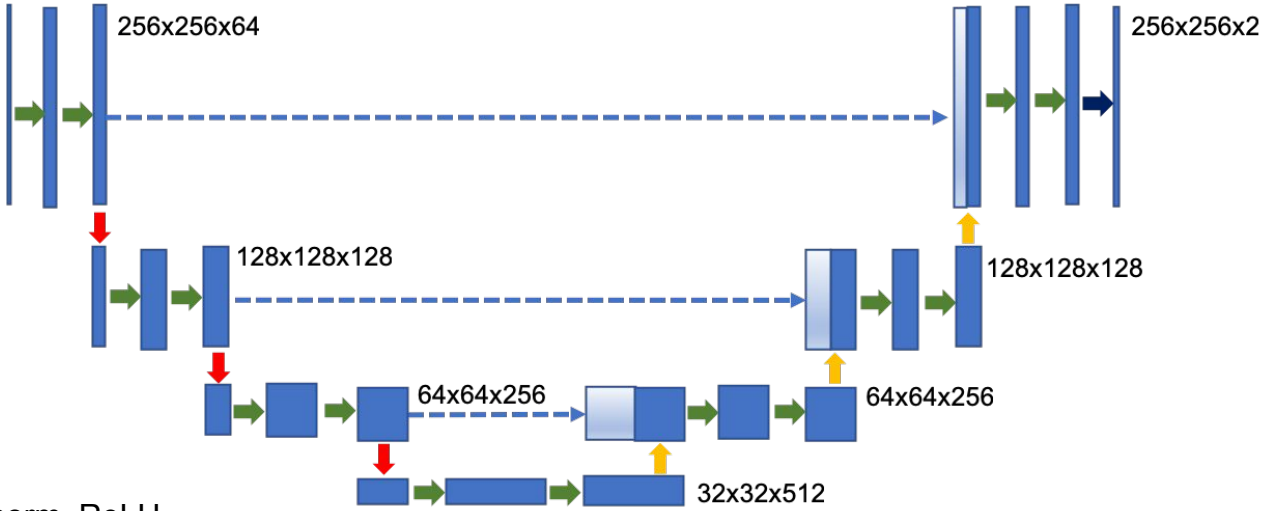
Neurons

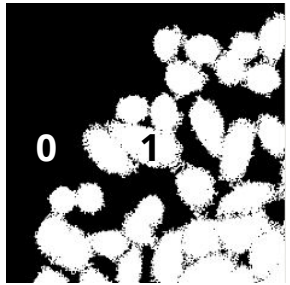
Tissue section



U-Net Implementation







conv 3x3, batchnorm, ReLU

Max pool 2x2

Bilinear upsamling

Conv 1x1, softmax

Limitations and Future Directions

- *Underway:*
  - Implementing distributed processing
  - Training the U-Net model
- Which of the three parallelization strategies offered by Julia is the most powerful and generalizable?
- How robust is DL-based segmentation compared to Watershed?
- Potential of transfer learning: Can we use the DL network to segment cells...
  - That are organized in dense structures?
  - That have unusual shapes (e.g. multinucleated, mitotic)?
  - In tissue sections?
- Can we parallelize other imaging operations in Julia?