Region of Interest Segmentation Graphical User Interface Manual

Table of Contents

[Introduction 3](#_Toc147332226)

[Overview 3](#_Toc147332227)

[Layout 3](#_Toc147332228)

[Segmentation GUI 3](#_Toc147332229)

[System Requirements 4](#_Toc147332230)

[OS and Software 4](#_Toc147332231)

[Hardware 4](#_Toc147332232)

[Software Installation 4](#_Toc147332233)

[Launch and Use 4](#_Toc147332234)

[Launching the GUI 4](#_Toc147332235)

[Loading in images 5](#_Toc147332236)

[Navigating higher dimensions 5](#_Toc147332237)

[Visualization of images 5](#_Toc147332238)

[Preprocessing 5](#_Toc147332239)

[Preprocessing Options 6](#_Toc147332240)

[Tools 7](#_Toc147332241)

[Cropping Tool 7](#_Toc147332242)

[Taper Tool 7](#_Toc147332243)

[Color Distance Transformation Tool 7](#_Toc147332244)

[Segmentation Methods 7](#_Toc147332245)

[Post Processing Options 7](#_Toc147332246)

[Configurable Active Contours 7](#_Toc147332247)

[Masking Tool 7](#_Toc147332248)

[Modularity and Development Options 7](#_Toc147332249)

[Bibliography 7](#_Toc147332250)

# Introduction

## Overview

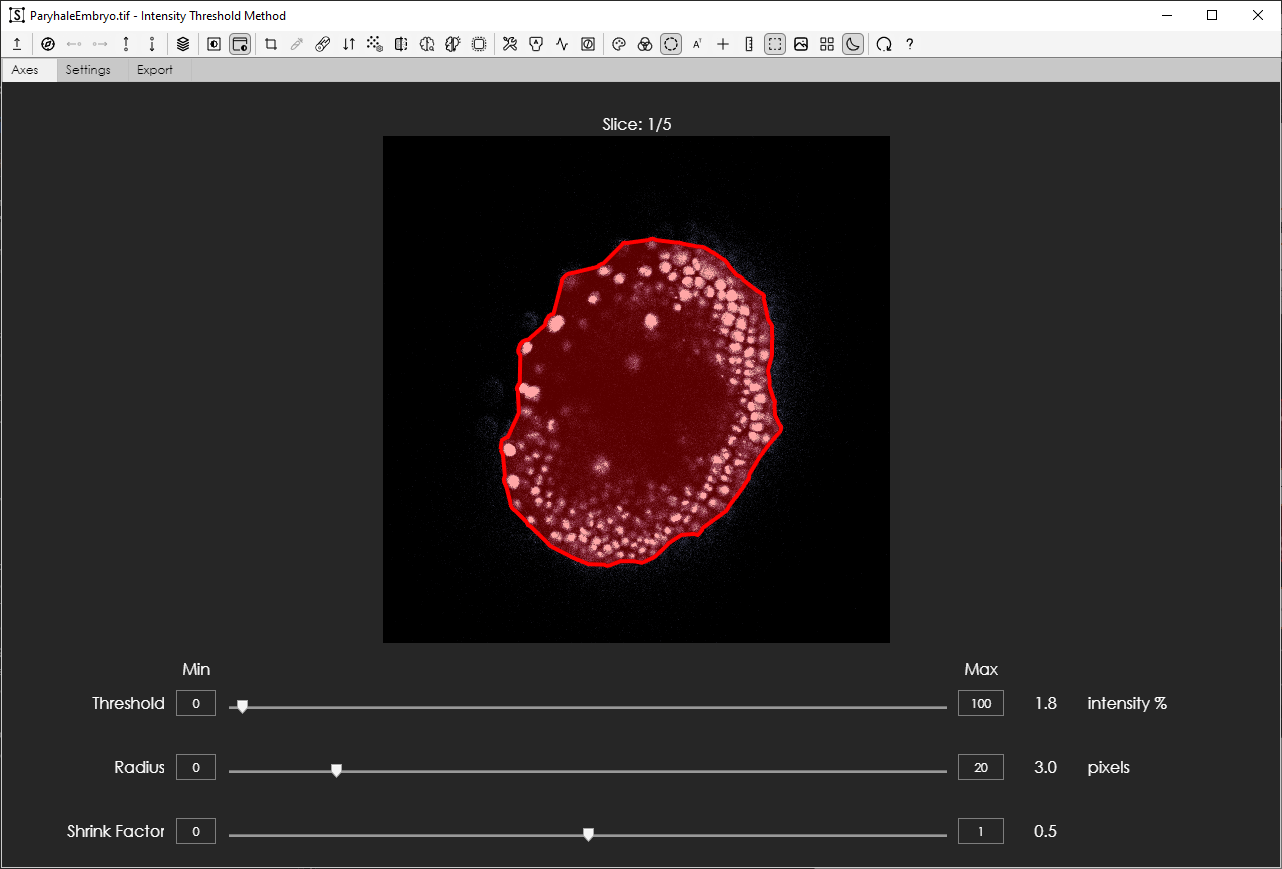
This graphical user interface (GUI) is for single region of interest (ROI) or multiple ROI image segmentation. The GUI utilizes just three parameters set by the user to conduct calculations of either a point cloud, a boundary curve (ordered point cloud), a mask (binary image), or image labels (label matrix). The reduced parameter set simplifies computation for the user and increases reproducibility while still allowing for preprocessing steps to increase robustness for various user cases. Some of the available preprocessing options include log scaling, denoising, preprocessing by a deep network, user defined ROI based tapering, and a color distance transform when dealing with color images if there is a specific color the user wants to target. The GUI is equipped with post processing options as well – user configurable active contours and manual adjustment of masks.

## Layout

### Segmentation GUI

Parameter units

Parameter names



5. Axes

2. Toolbar

7. Parameter sliders

3. Tabs

1. Window Title

4. Index Info

6. Overlay

8. Parameter bounds

**Figure 1 The ROI Segmentation GUI with parts labeled.**

1. Window Title – displays name of file that is loaded in and the parametric segmentation method name.
2. Toolbar – holds all action buttons for data loading/navigation, preprocessing options, post processing options, and visualization settings. Reset and help buttons are also found here.
3. Tabs – holds the axes tab where segmentation is visualized, settings tab where post processing and visualization can be adjusted, and the export tab to specify what and where to save results.
4. Index Info – informs user of slice and timepoint/file index currently being viewed if there are higher dimensions or the user selected multiple files
5. Axes – where the image and segmentation result are plotted. Only a single 2D image slice can be visualized at a time.
6. Overlay – visualization of the results to be exported
7. Parameter sliders – adjusts the parameter values within the range of parameter bounds.
8. Parameter bounds – min and max fields can be set by the user. They represent the lower and upper bound of parameter values, respectively. This can allow the user to zoom into a smaller range or explore across a larger span of values as needed.

# System Requirements

## OS and Software

Windows 10 or greater is recommended but the minimum is for your computer to be running a 64-bit operating system. The software is also compatible with Linux (Ubuntu 24.02 recommended) and MacOS (Sierra and newer have been tested).

MATLAB 2023A or newer and the following toolboxes:

* + - 1. Image Processing Toolbox
      2. Deep Learning Toolbox
      3. Statistics and Machine Learning Toolbox

For additional information concerning requirements, please visit the MATLAB System requirements pages:

**Windows** – <https://www.mathworks.com/support/requirements/matlab-system-requirements.html>

**Mac** – <https://www.mathworks.com/support/requirements/matlab-mac.html>

**Linux** – <https://www.mathworks.com/support/requirements/matlab-linux.html>

## Hardware

Recommended

* + Double Threaded Quad Core processor ≥ 3.0 GHz
  + GPU with 2 GB RAM
  + 64 GB RAM to deal with larger data sets (if loading via terminal), otherwise, 16 GB is sufficient to have some overhead.

Minimum

* + Double Threaded Dual Core processor ≥ 2 GHz
  + 8 GB RAM

## Software Installation

The software can be installed from the Kumar Lab GitHub (<insert link here>) by downloading the software subdirectory from the repository. All other directories are optional. Upon downloading the zip folder, extract the contents into the destination folder Make sure to keep all subdirectories in the same place so that the software dependencies do not break.

# Launch and Use

## Launching the GUI

The GUI can be launched from a directory by double click. If utilizing the uncompiled, unobscured code files (.m and .mlapp), MATLAB will launch. Users can launch the GUI from the terminal and assigned to a variable. It can take a file path, 2-4D arrays, and a cell array of file paths as inputs.

Terminal line launch:

app = ROISegmentationGUI;

app = ROISegmentationGUI(ImageVariable);

## Loading in images

* Load image – launch a dialog window that will prompt file selection. Compatible formats include .mat, .tif/.tiff, .png, .jpg/.jpeg, and .gif files but uncompressed .tif files and uncompressed .mat files are most advisable.

If using .mat files, please ensure that the file was saved as version 7.3 and no compression was used. This ensures variables can be partially loaded to reduce memory usage rather than having to fully load a variable in. To save a variable from your workspace in this way, run the following line in the terminal:

save <FileName>.mat <VariableName> -v7.3 -nocompression

**Note** that the GUI will perform lazy loading by default to allow users with computers that have less memory to continue to run other programs in case datasets are large. If the user has sufficient memory, it is advisable to first load the data into the workspace and launch the app with the data variable as the input to the app. This will speed up navigation as well as batch processing when creating segmentation masks.

## Navigating higher dimensions

* Instant Navigation – click this button to instantly navigate to a specific slice in the z stack (3rd dimension of data is greater than 1) or a specific timepoint in a time series (4th dimension of data is greater than 1).
* Left/Right Navigation arrows – click to move through stack’s time dimension. The right arrow moves into later timepoints (+1 timepoint) and left arrow moves into previous time points (-1 timepoint).
* Up/Down Navigation arrows – click to move through stack’s z dimension. The down arrow moves into later slices (+1 z slice), the up arrow moves into previous slices (-1 z slice).

## Visualization of images

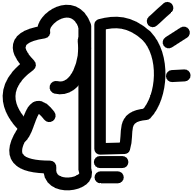
* Auto contrast – will toggle the automatic contrasting of images in the axes. If on, whenever a new image is loaded or the user navigates to a new slice or timepoint, the axes will be automatically contrasted using a range percentile method.
* Manual contrast – launches a histogram of pixel intensity values with a lower and upper bound to be set by the user.
* Color scheme – change the colormap assigned to the grayscale image data displayed in the axes.
* Contour color – change the color of the ROI contour.
* A black background with a black square

  Description automatically generated with medium confidenceTranspose – will toggle the view of the image in the axes as being transposed or non-transposed.
* Major and Minor Axes – will toggle the visualization of the major and minor moments of inertia of the masked ROI.
* Colorbar – will show a color-coded bar atop the axes. The colors will correspond to pixel intensity values in the image.

## Preprocessing

The GUI can apply a variety of functions on the image data that is loaded up. Some functions may either increase separability of pixels, homogenize image regions, change which pixels are considered, weigh pixels differently, etc. These preprocessing options can enhance features that might facilitate segmentation of the ROI.

### Preprocessing Options

* Crop – launches a GUI for cropping the image. The user is prompted to draw an ROI. Any previously set crops can also be cleared via the tool. See [crop tool section](#_Cropping_Tool) for more information.
* Color target – a GUI is launched for the user to select a color target. This color is then used to apply a color distance transform such that each pixel instead takes on a value equal to its distance away from the target color [4, 8]. See [color distance transformation tool section](#_Color_Distance_Transformation) for more information.
* Log Scaling – toggles application of the log function to the data. It will compress the range of values as the dynamic range of an image is scaled differently.
* Reverse polarity – will subtract the image values from the maximum image value to produce a polarity flip. This can be helpful if trying to segment out a dark region in an image instead of a bright object / ROI.
* Change denoising method – clicking this button will launch a list of available options for denoising methods that can be utilized for the user’s data.
  + **Down scale up scale** down sample images and then up sample to smooth through noise.
  + **Median Filter and Smooth** applies a 3x3 median filter and then smooth through the data with a gaussian blur.
  + **Median Filter Von Neumann Neighborhood** applied a sliding Von Neumann neighborhood (4-connected neighborhood with radius 1) median filter to the image.
  + **None** will just clear the denoising method in the computation engine and no denoising method will be applied to the data.
* Denoise toggle – toggles the denoising function on/off without changing the method.
* Artificial Intelligence Network – allows user to select a deep network to feed the data through. By default, the software has a pretrained DeepLabV3+ Resnet-18 architecture that has been trained on a variety of microscopy images from various modalities and synthetic images [3]. The output from processing is the activation layer output.
* AI toggle – will toggle the AI processing on/off.
* Boundary taper – will launch a GUI to allow the user to define a ROI where a taper to the boundary will be applied. A Tukey window normalization is applied, starting with values of 1 at the boundary of the ROI and inside it and then decrease to 0 at the image boundary.
* Change segmentation method – launches a dialog window from which the user can select a 3-parameter segmentation method.
* Auto params – will automatically estimate some approximate parameter values for the selected segmentation method, if there exists some auto param method that has been configured with the current, set method.
* Active contours – will apply active contours to the segmentation result to further refine the segmentation [2].
* Masking tool – will launch the masking tool to allow for manual mask refinement. See [masking tool section](#_Masking_Tool) for more information.

## Performance Visualization

* Auxiliary view toggle – will toggle between the original image view and the auxiliary image being plotted in the axes. The auxiliary image is the image that results from applying preprocessing steps.
* Engine visualizer – will launch a visualizer of the engine processing steps. As parameter values are changed, the resulting output from the corresponding processing step and subsequent computations will be updated. The visualizer can be kept open and will update whenever a new image is loaded or navigation to a new slice or timepoint occurs. Updates will also take place whenever the auxiliary image changes because of the preprocessing options changing. Likewise, when batch processing is taking place, the user can visualize all the steps as the processing takes place across slices and stacks.

Screenshot of visualization

## Batch processing

After setting all preprocessing settings as desired and selecting the preferred segmentation method, users can batch process their data if the data is multidimensional or there are multiple files that were selected. While the engine is processing through individual 2D slices, the view will be updated in the GUI’s axes and the Slice/Timepoint Index label will also update.

To accelerate processing, visualizations aside from the GUI’s axes should be closed and turned off. Additionally, auto contrast should be turned off and the user should manually set the color limits. Note that the more preprocessing and post processing steps that are selected, the longer segmentation computation will take.

# Preprocessing Tools

## Color Distance Transformation Tool

## Cropping Tool

## Taper Tool

# Segmentation Methods

# Post Processing Options

## Configurable Active Contours

## Masking Tool

# Saving and Reloading

## Export

Users can export their settings and results into .mat, .json. and .xml files. These files are formatted to be importable back into the software as well as exportable in an easily accessible format for other uses outside of MATLAB.

## Reloading Settings and Results

Results stored in a segmentation software export file of .mat, .json. and .xml formats can be fed back into the software

# Modularity and Development Options

## Configuring Additional Methods

### Parametric Segmentation

Adding additional segmentation methods as options for the GUI to use…

### AI/ML: Deep Networks

# Bibliography

1. Bradley, Derek & Roth, Gerhard. (2007). Adaptive Thresholding using the Integral Image. J. Graphics Tools. 12. 13-21. 10.1080/2151237X.2007.10129236.
2. Chan, T. F., & Vese, L. A. (2001). Active contours without edges. *IEEE Transactions on image processing*, 10(2), 266-277.
3. Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 801-818).
4. Dony, R. D., & Wesolkowski, S. (1999, May). Edge detection on color images using RGB vector angles. In *Engineering Solutions for the Next Millennium. 1999 IEEE Canadian Conference on Electrical and Computer Engineering*(Cat. No. 99TH8411) (Vol. 2, pp. 687-692). IEEE.
5. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).
6. Kumar, A., Wu, Y., Christensen, R. et al. Dual-view plane illumination microscopy for rapid and spatially isotropic imaging. Nat Protoc 9, 2555–2573 (2014).
7. Parent, M. T., Mautino, A. R., & Kumar, A. (2022). A multi-view line scanning confocal microscope for fast imaging of extended samples. *Biophysical Journal*, *121*(3), 411a.
8. Wesolkowski, S. B. (1999). Color image edge detection and segmentation: A comparison of the vector angle and the Euclidean distance color similarity measures (Master's thesis, University of Waterloo).