User Guide for Emulate CAR T Image & Data Analysis Tool

April 22nd, 2024

EP-232 Rev. A

User Guide for Emulate CAR T Image & Data Analysis Tool

Overview

Introduction

This protocol was developed to enable the assessment of chimeric antigen receptor (CAR) T-cell recruitment and killing on a co-culture cancer cell line Organ-Chip model.

Contents

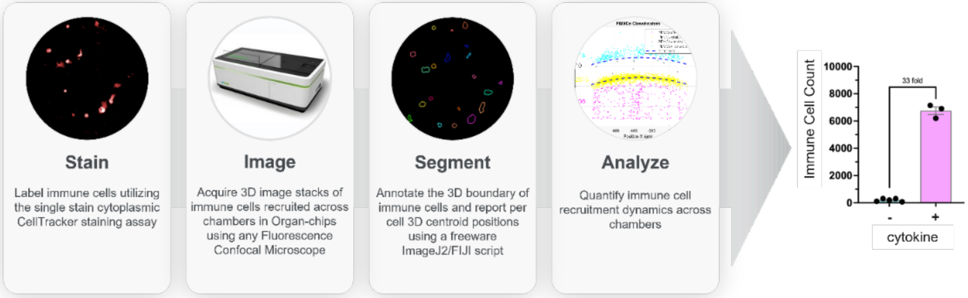
Purpose

Overview

Introduction

Emulate has developed software tools (ImageJ2/Fiji and a Python script) for segmenting and analyzing migratory immune cells and killing response, referred to as the “Emulate CAR T Image & Data Analysis Tool.” This user guide provides guidance on staining and imaging parameters for acquiring three-dimensional (3D) image stacks with a fluorescence confocal microscope as well as how to use the CAR T Image & Data Analysis Tool. This document can serve as a starting point to assist researchers in adapting and developing an analytical routine for their specific microscopy capabilities.

Figure 1


Stain
Label immune cells utilizing the single stain cytoplasmic CellTracker staining assay
Image
Acquire 3D image stacks of immune cells recruited across chambers in Organ-chips using any Fluorescence Confocal Microscope
Segment
Annotate the 3D boundary of immune cells and report per cell 3D centroid positions using a freeware ImageJ2/FIJI script
Analyze
Quantify immune cell recruitment dynamics across chambers
Immune Cell Count
10000
8000
6000
4000
2000
0
33 fold
-
+
cytokine

Imaging and analysis workflow supporting the CAR T model.

Immune Cell Staining Guidance

Overview

Introduction

The ImageJ2/Fiji 3D Image Segmentation Script quantifies:

1. The immune cells stained with the cytoplasmic stain CellTracker™ Deep Red Dye (ThermoFisher Scientific, Cat. C34565). Please refer to "EP-231: Basolateral Recruitment of Chimeric Antigen Receptor (CAR) T Cells in an Organ-Chip Model of Non-small Cell Lung Cancer (NSCLC)" for immune cell staining and administration.
2. The antigen-dependent killing of target cells, stained with CellEvent^{TM} Green (ThermoFisher Scientific, Cat. C10423). Please refer to "EP-231: Basolateral Recruitment of Chimeric Antigen Receptor (CAR) T Cells in an Organ-Chip Model of Non-small Cell Lung Cancer (NSCLC)" for staining.

Chip Imaging Guidance

Overview

Introduction

Confocal fluorescence microscopy is well suited for 3D imaging within diffuse, thick samples. In this user guide, confocal microscopy is the preferred imaging tool for quantifying migratory immune cells. Prior to imaging, the chip should be fixed to preserve the migratory state of immune cells. Please refer to “EP-232: Basolateral Recruitment of Chimeric Antigen Receptor (CAR) T Cells in an Organ-Chip Model of Non-small Cell Lung Cancer (NSCLC)” for details regarding chip fixation.

Optimizing Image Performance

To optimize imaging performance, it is recommended to characterize the microscope’s point spread function (PSF). This can be done by imaging fluorochrome-labelled polystyrene beads with diameters smaller than the microscope’s theoretical resolution. Light diffraction blurs the bead’s image to make it appear larger than its true dimensions. This PSF image defines the resolution in the axial and lateral directions. Because optical microscopes have worse resolution in the axial versus lateral direction, the PSF is approximately a cylindrically symmetrical ellipsoid.

Potential Errors

Fully understanding the microscope’s resolution will help to avoid under- or over-sampling mistakes, which are frequently made during imaging. Table 1 summarizes these errors’ potential effects on imaging as well as their possible consequences when thresholding with the **ImageJ2/Fiji 3D Image Segmentation Script**.

Continued on next page

Overview, Continued

Table 1

Effect of sampling error during imaging and segmentation

Sampling Error	Effect of Imaged Cell Object	Potential Segmentation Errors with Script
Under-sampling	• Partial capture of signal from immune cell	• Higher rates of false-negative results. Partial cell images smaller than the minimum cell size parameter will be missed. Compensation by reducing the acceptable minimum cell value may increase the rate of false positives. • Incorrect 3D centroid measure.
Over-sampling	• Stretched immune cells • Lower signal-to-noise ratios (SNR) due to greater signal spread at the detector	• Higher rates of false-negative results. The threshold-based cell segmentation protocol will struggle to identify true signal from background in low-SNR images. Using the incorrect preprocessing technique to remove the background signal may remove true signal.

Nyquist Sampling Theorem

To avoid sampling errors and ensure imaging completely captures cellular objects, it is important to consider the Nyquist Sampling Theorem, which states that sampling frequency should be at least twice the resolution of the current dimensions. In other words, the minimum justified pixel size is one-half (1/2) the lateral resolution, and the minimum z-step size should be one-half (1/2) the axial resolution. With this guidance in mind, ideal Organ-Chip imaging parameters are recommended in the next section.

Recommended System Parameters

Overview

Since imaging is performed throughout the Organ-Chip, it is recommended to use an objective lens with the parameters listed in Table 2.

The lens' large working distance enables imaging through the entirety of the chip's thick PDMS layer as well as its upper and lower channels. The low numerical aperture (NA) supports faster imaging speeds and has a resolution that will not hinder downstream quantification.

It is important to note why an air immersion objective was selected. Given the closeness in refractive index between water and the chip, it was determined that confocal microscopes with built-in laser autofocusing suffer from defocusing issues when a water immersion objective lens was utilized.

Table 2

Recommended objective lens parameters to image immune cells and caspase objects in the Organ-Chip when imaging with fluorescence confocal microscopy.

	Recommended Parameters
Objective Lens Type	20x Air
Numerical Aperture	0.4
Working Distance [mm]**	8.39

*** Working distance is defined as the distance from lens to focal point without coverslip correction*

Resolution

Importantly, resolution is a function of the objective's Numerical Aperture (NA), the chip system's refractive index, the chip's depth of imaging, and the fluorochrome's emission wavelength. Image acquisition should always be carried out near the resolution limit of the optical system (i.e., justified pixel size and z-step size). For example, based on the recommended objective lens specifications provided above and the emission wavelength of the CellTracker™ Deep Red Dye, the minimum justified pixel size is 283 nm, and the z-step size is 2.8 µm (Table 3).

Table 3

Representative optical resolution and pixel size in confocal microscopy using CellTracker™ Deep Red Dye and 20X air objective.

	Lateral Resolution dx,y	**Axial Resolution dx,y**
Resolution Limit	$0.4\lambda_{em}$ / NA	$0.4\lambda_{em}$ / NA^2
Suggested Protocol Parameters 20x Air, NA = 0.4, λ_{em} = 650 NM	650 nm	5.7 µm
Min Pixel Size	283 nm	2.8 µm

Continued on next page

Recommended System Parameters, Continued

Recommendations

Recommendations to ensure imaging success:

Recommendation	Description
Dynamic Range	It is essential to maximize the dynamic range by avoiding overexposure of the sample during imaging. Overexposure occurs due to long scanning dwell times and/or the excitation laser power being too high and can lead to pixel saturation. When saturated, higher intensity gray levels in the histogram are lost, adversely affecting dynamic range. Pixel saturation paired with incorrect background pre-processing techniques can lead to the inability to properly segment cells with thresholding-based protocols.
Laser Power Drift	By regularly imaging fluorochrome-labelled polystyrene beads, it is possible to monitor potential drift in the excitation laser power. Ensuring constant excitation power is critical for consistent sensitivity and specificity of performance if the threshold-based cell segmentation protocol.
System Drift	PSF geometry can be used as an indicator of potential problems with the objective lens, scanning components, and other relay optics within the microscope. It is important to assess consistency in diffraction pattern and to ensure the resolution in the lateral and axial directions remain unchanged.

Imaging Guidance

Overview

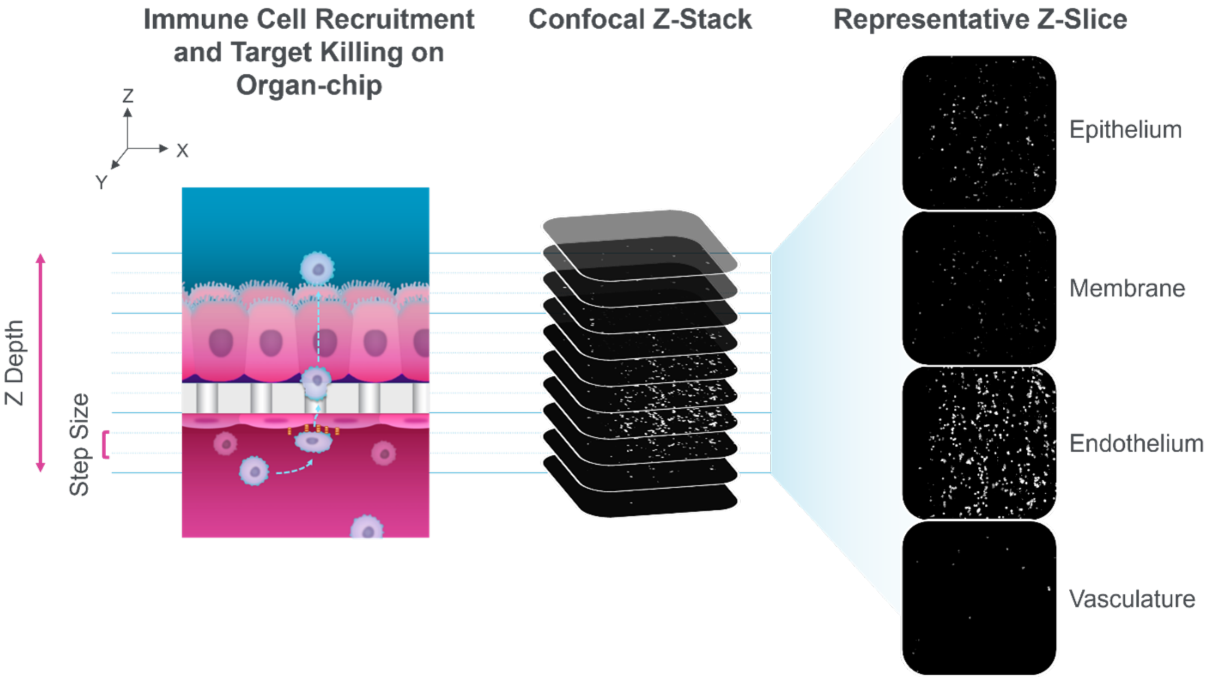
To ensure successful analysis when setting up the image acquisition, the following parameters must be included:

Parameters	Description
Fluorescence imaging of CellTracker™ Deep Red-stained immune cells	Choose the appropriate excitation/emission channel to capture immune cells stained with the CellTracker dye. Excite fluorophore with a laser power that maximizes image dynamic range by preventing pixel oversaturation and maximizing SNR. The analytical tools package analyzes only a single channel image, but this does not preclude extending imaging to additional features, such as nucleic stain or transmitted light. That said, ensure there is no crosstalk between collected channels, or else cells will be incorrectly segmented.
Fluorescence imaging of CellEventTM Green-stained cell killing	Choose the appropriate excitation/emission channel to capture cell killing stained with the CellEvent Green caspase dye. Excite fluorophore with a laser power that maximizes image dynamic range by preventing pixel oversaturation and maximizing SNR. The analytical tools package analyzes only a single channel image, but this does not preclude extending imaging to additional features, such as nucleic stain or transmitted light. That said, ensure there is no crosstalk between collected channels, or else cells will be incorrectly segmented.
Capturing z-stack range to acquire extant of migratory immune cells and the killing response (caspase signal)	Start the first z-slice slightly (~10 µm) below the membrane to capture attached immune cells and any cell monolayer (if applicable). Extend the z-stack height upwards to include the total membrane height (50 µm) as well as the total top channel cell monolayer height (depending on cell type, from 10–200 µm). If possible, it is recommended to extend an extra 10-µm margin on both ends of the z-stack to account for any possible misalignment of samples. Recommendations for the sampling resolution of the z-stack based on the objective are outlined in the guidance above.

Continued on next page

Imaging Guidance, Continued

Figure 2



Representative data captured utilizing confocal imaging.

Microscopy Terms

Term	Definition
Point Spread Function	Measure for the quality of an optical system as it reveals how points are blurred in an image.
Optical Resolution	Ability of an imaging system to resolve two points. In other words, the shortest distance between two points that can be observed as separate entities.
Working Distance	Distance from lens surface to focal point.
Numerical Aperture	Range of angles over which the optical system can accept or emit light.
Nyquist Frequency	Minimum rate at which a signal can be sampled without introducing error.

Installation

Installation Overview

System Requirements

- Windows 10 and above (recommended)

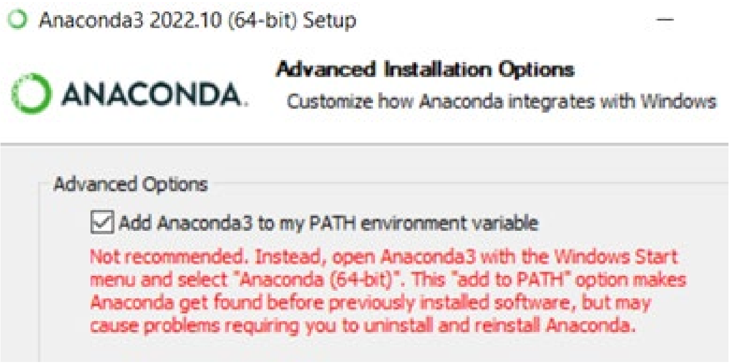
Installing ImageJ Script for 3D Object Counting

Step	Action
1	Download and install the latest version of the ImageJ2/Fiji application (requires v1.52i or later): https://imagej.net/software/fiji/downloads The 3D object counter comes already installed with the ImageJ2/Fiji application. To confirm, it should appear under the ImageJ2/Fiji analyze menu. It contains two entries: ● 3D Object Counter ● Set 3D Measurements
2	Download the latest version of the *Excel Read and Write* plugin: https://imagej.net/plugins/read-and-write-excel - installation
3	Update ImageJ2/Fiji using the “Help > Update…” menu.
4	When the update is complete, click “OK.”
5	Click on “Manage Update Sites” in the ImageJ Updater.
6	Find and check the “ResultsToExcel” checkbox.
7	Click “Close” and then “Apply Changes.”
8	Restart ImageJ2/Fiji before proceeding.
9	Download the most recent version of the ICR_3D Analysis_v0.1.10.ijm onto a computer.

Continued on next page

Installation Overview, Continued

Configure Python Environment

Step	Action
1	Download and install Anaconda 2022.10 for Windows with Python 3.9 (this exact version is required): https://repo.anaconda.com/archive/Anaconda3-2022.10-Windows-x86_64.exe **Note:** During installation, select “Add Anaconda3 to my PATH environment variable” (See Figure 3).
	 Anaconda3 2022.10 (64-bit) Setup ANACONDA. Advanced Installation Options Customize how Anaconda integrates with Windows Advanced Options ☑ Add Anaconda3 to my PATH environment variable Not recommended. Instead, open Anaconda3 with the Windows Start menu and select "Anaconda (64-bit)". This "add to PATH" option makes Anaconda get found before previously installed software, but may cause problems requiring you to uninstall and reinstall Anaconda. Figure 3. Window for selecting advanced installation options.
2	Download and extract the most recent package of Emulate CAR T Image & Data Analysis Tool from Emulate.
3	Navigate to the folder of Emulate CAR T Image & Data Analysis Tool and double click “install.bat”. Then, wait for the Python libraries to fully install. **Note:** If the installation fails, please open Anaconda Prompt (Anaconda3) from Windows Start and run the following commands (for each line, copy and paste it into Anaconda Prompt, and hit enter) to install dependencies. ● pip install matplotlib==3.5.2 ● pip install numpy==1.21.5 ● pip install pandas==1.4.4 ● pip install PySimpleGUI==4.60.4 ● pip install scikit_image==0.19.2 ● pip install nbconvert==6.4.4
4	Refer to the “Running JupyterLab” section to start running analysis.

High-Level Overview

High-Level Overview

Overview Flowchart

Below is the flowchart of the entire pipeline (Figure 4), depicting the flow of data from input through processes to outputs. There are four primary processes in the pipeline:

1. 3D object counting (ImageJ script)
2. Co-localization
3. Feature extraction and analysis
4. Maximum projection

3D object counting takes in a raw image for each FOV, finds all the cell objects, and outputs an object map showing only the cell objects as well as an Excel spreadsheet of the object statistics. The object maps are used by co-localization to determine cell states (live or dead) by overlapping stains between cell tracker and caspase. The Excel spreadsheet of the object statistics is used as inputs for classification and analysis of key features such as counts, volume, and intensities. In a separate process, maximum projection is applied to raw images to calculate the total intensity.

Among the four processes in this application, co-localization and feature analysis are two key pieces. 3D object counting (ImageJ script) is implemented in ImageJ macros, while all the other components are implemented in Python.

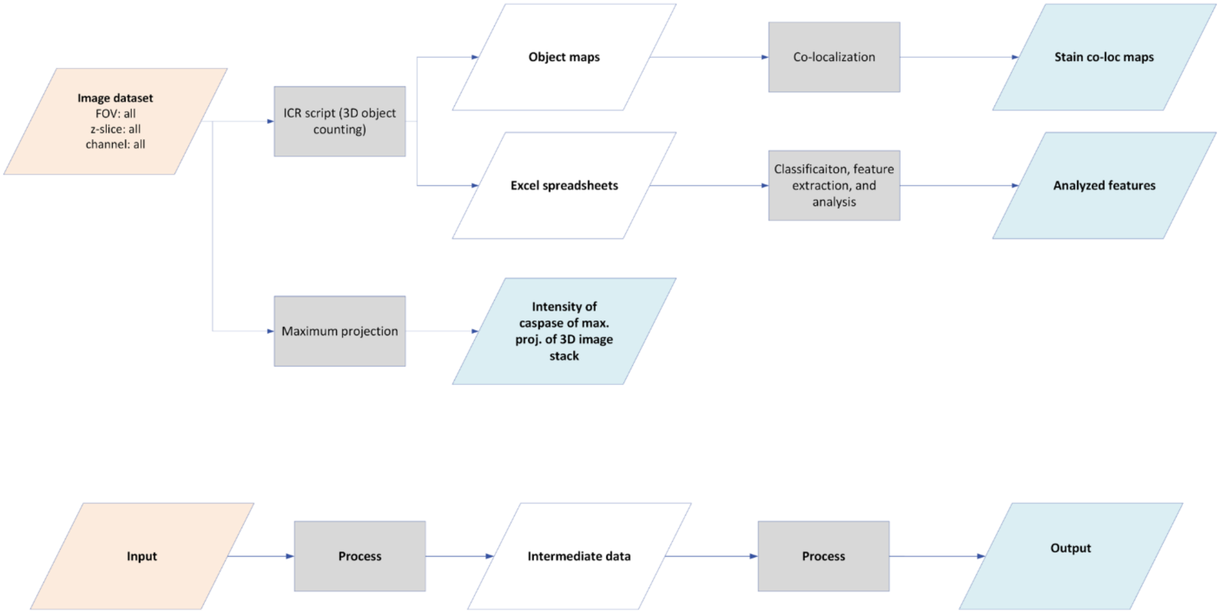


Figure 4. A flowchart illustrating the entire pipeline from a high-level perspective.

ImageJ Script for 3D Object Counting

Overview

Introduction

The ImageJ script for 3D object counting automates visualization and annotation of CAR T cells acquired within a 3D image stack. The 3D centroid position is then calculated per segmented cell. Z-stacked imagery can be acquired using any fluorescence confocal microscope.

Contents

Algorithm

Algorithm

There are two primary steps in the ImageJ script: segmentation and 3D object counting.

Segmentation identifies pixels as candidates to be considered as objects. This step is implemented using threshold-based segmentation, where only pixels with intensity above the given threshold are kept and the others are discarded.

3D object counting, the next step, extracts objects (loosely defined as cells) from the images and discards the background and noises by applying the 3D object counter plug-in to the input dataset of raw images. With object labeling, connected pixels are labeled and essentially merged as a single object. Finally, objects with sizes outside thresholds of the min and max sizes are discarded, keeping only the ones within the thresholds. This process was written in ImageJ macros (high-level commands). The macros call the 3D object counter plug-in in ImageJ, implemented using the connected-component labeling algorithm in Java by S. Bolte & F. P. Cordelières.

Input Data

Flowchart

The input to 3D object counting is the dataset of raw images, described earlier.

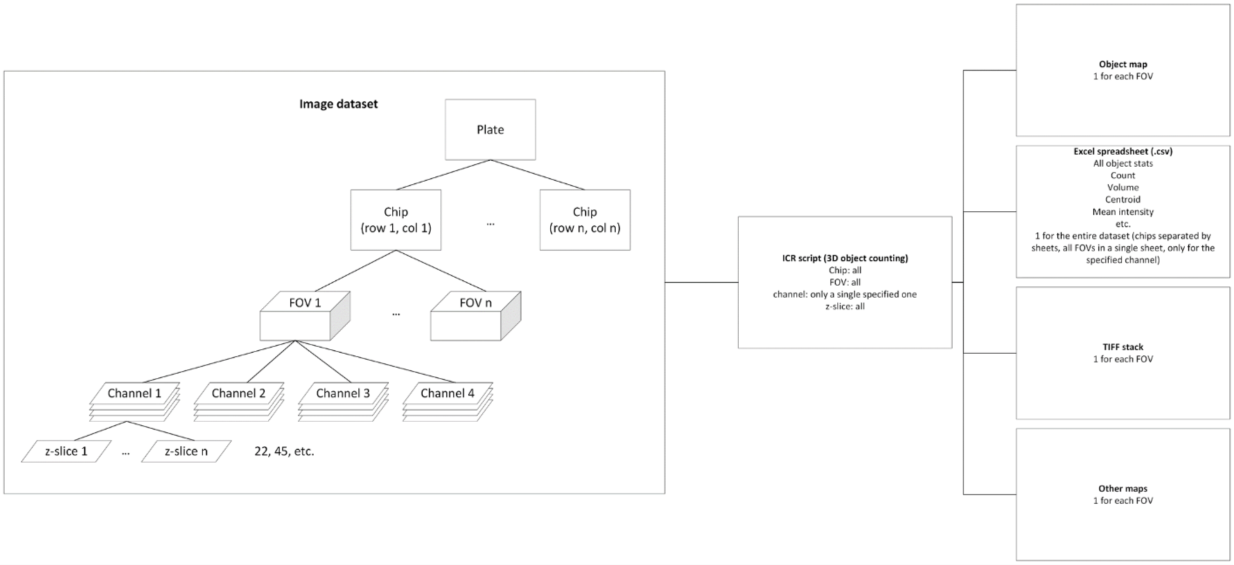


Figure 5. A flowchart illustrating the workflow of 3D object counting.

Accepted formats

- Series of 3D image z-slices, with only a single channel
- (3D image z-stacks with only a single channel are **not** supported)
- TIFF formats are highly recommended

Notes

- **All channels need to be separated. An image must contain only one channel.**
- It is recommended to organize images by chips (separating them into individual folders for different chips) and run the analysis in batches.
- Channels
    1. Cell tracker for CAR T: fluorescent signal from fixed, single-stained CAR T cells
    2. Caspase: fluorescent signal from fixed, single-stained caspase+ objects
- **Please be extra careful when entering parameters, as they are critical for the analysis to be done correctly and can affect downstream analysis and results.**

Continued on next page

Input Data, Continued

Image Acquisition Parameters

Parameter	Definition
Project Name	User defined project name
Chip ID in File Name	Component of image file name defining chip ID
FOV (field-of-view) in File Name	Component of image file name defining imaged field-of-view
Plane in File Name	Component of image file name defining z-plane in image stack
Channel in File Name	Component of image file name defining fluorescence channel
Image File Type Extension	Defines image type for analysis
Source Directory	Defines pathway to folder containing imagery for analysis
Output Directory	Defines pathway to folder created to accept results (i.e., script output)

Image Processing Parameters

Parameter	Definition
Channel ID to analyze	The index of the channel in the filename
Name of channel/stain	Name of the channel
Select if input files are compiled z-stacks	Fluorescence image containing multiple z-planes within an image stack
Number of planes in z-stacks	Number of z-slices in the z-stack
Bottom position of z-stack	In µm, the location of the lowest z-plane within an image stack
Distance between planes in z-stack	In µm, the distance between each z-plane of an image stack
Despeckle	Median filter designed to remove salt and pepper noise by replacing each pixel with the median value of its 3 x 3 neighborhood
Sliding parabola	Image processing tool that removes features smaller than the defined radius

Rolling Ball Radius	Image processing tool to fix uneven background signal. Set radius to at least the size of the largest object NOT part of background.

For further detail on image processing: https://imagej.nih.gov/ij/docs/menus/process.html

Object Processing Parameters

Parameter	Definition
Minimum Intensity Object Threshold Value	Define the intensity value separating background from cell
Minimum Cell Volume Size	Defines minimum cell size detection limit, reported in voxels (pixels^3)
Maximum Cell Volume Size	Defines maximum cell size detection limit, reported in voxels (pixels^3)
Centroid Maps	Annotation mask of geometrical centers for all cells detected
Object Maps	Annotation mask overlay detected signal from all cells
Surface Maps	Annotation mask overlay of all surface voxels

Key Input Parameters

Six key input parameters are:

1. Intensity threshold
2. Min object size
3. Max object size
4. Number of planes in z-stack
5. Bottom position of z-stack
6. Distance between planes in z-stack

Continued on next page

Input Data, Continued

Selecting Parameter Values

The intensity threshold is the most important parameter because it directly affects segmentation. It has a **non-linear** relationship with output object counts because objects can be split/merged differently at various thresholds. Therefore, this parameter should be tuned first.

The min object size threshold is the second most important parameter. It comes into effect after segmentation, when objects are excluded based on their sizes. It has an approximately **linear** relationship with output object counts.

The user needs to be very careful with selecting the appropriate values for these parameters, especially the intensity threshold, because the threshold-based segmentation is very sensitive to these parameters. It is strongly recommended that the **user tries a few sets of parameters on sample images in ImageJ, runs 3D object counting (ImageJ → Analyze → 3D Object Counter), and observes the results to decide the best set of parameters before** using these parameters for the analysis. Any slight changes can propagate down the pipeline and lead to differences in the end results.

Continued on next page

Input Data, Continued

Phenix Harmony Users

For Phenix Harmony (PerkinElmer®) software users, these parameters can be translated to the ImageJ script for 3D object counting:

- The intensity threshold can be copied over directly (See Figure 6, top left)
- The min size threshold needs to be converted from **micrometer^3 (µm^3)** in Harmony to **voxels** in ImageJ. For the conversion, please refer to the specifications of the microscope and imaging being used.

  For a rough estimate:

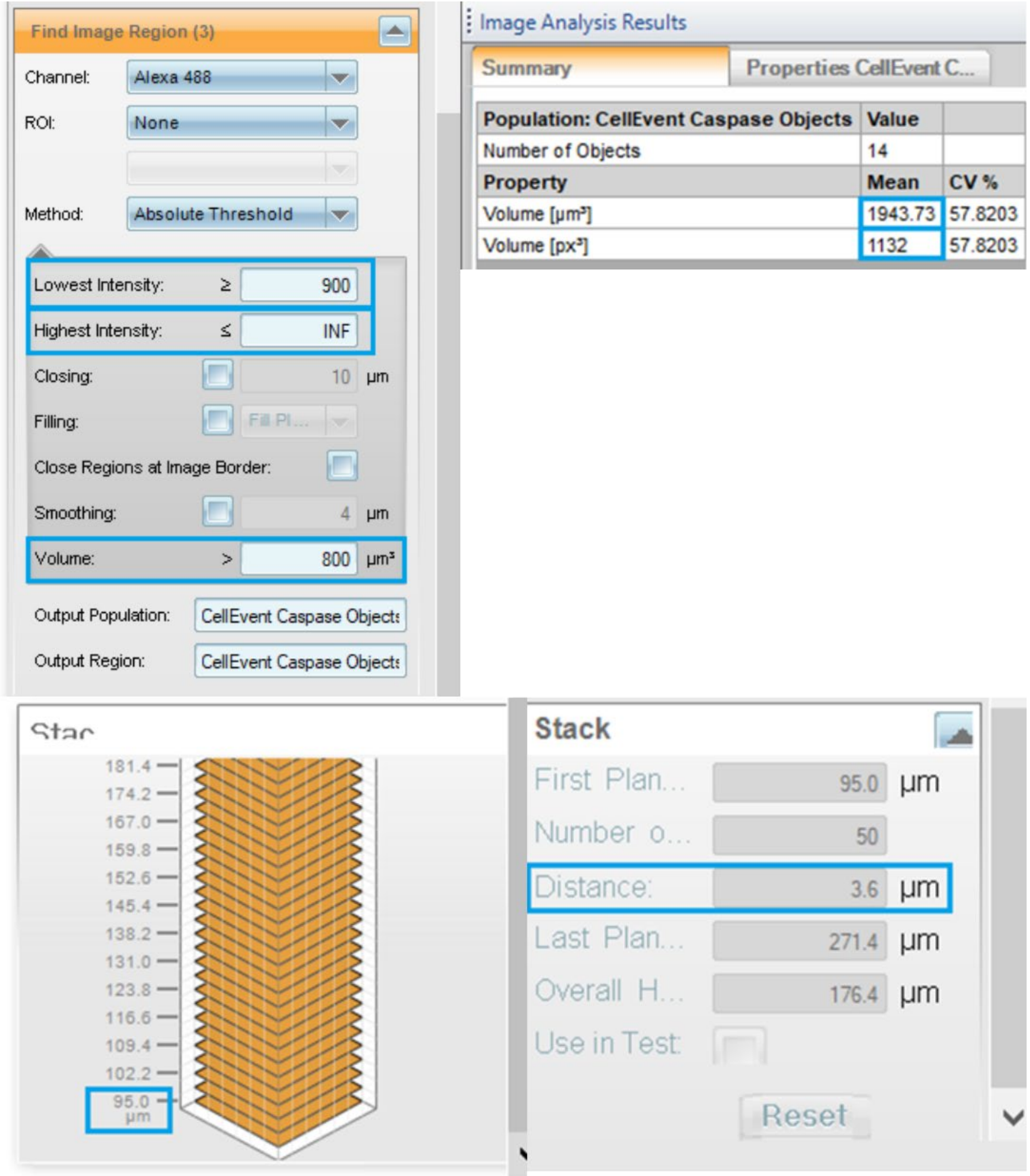
Step	Action
1	Take the volume threshold in Harmony (See Figure 6, top left), which is 800 µm^3 in this example.
2	Calculate the conversion ratio in the "Image Analysis Results" in Harmony (See Figure 6, top right), which is 1943.73 µm^3/1132 px^3 = 1.717 µm^3/px^3 in this example.
3	Convert the volume threshold from µm^3 to voxels (px^3): 800 µm^3 / 1.717 µm^3/px^3 ~= 465 px^3.
4	Subtract 30 voxels to be on the conservative side: 465 px^3 – 30 px^3 = 435 px^3.
5	This value can then be entered into ImageJ.

- The max size threshold in Harmony is INF (infinity), which corresponds to a very large value in ImageJ (for example, 999999999).
- The number of planes in z-stack can be copied over directly (this may vary across experiments)
- The bottom position of z-stack can be found in Harmony → Navigation → Stack (See Figure 6, bottom left)
- The distance between planes in z-stack can be found in stack settings (See Figure 6, bottom right)

Continued on next page

Input Data, Continued

Figure 6. Image Analysis



Find Image Region (3)

Channel: Alexa 488

ROI: None

Method: Absolute Threshold

Lowest Intensity: ≥ 900

Highest Intensity: ≤ INF

Closing: □ 10 µm

Filling: □ Fill Pl...

Close Regions at Image Border: □

Smoothing: □ 4 µm

Volume: > 800 µm³

Output Population: CellEvent Caspase Objects

Output Region: CellEvent Caspase Objects

Image Analysis Results

Summary | Properties CellEvent C...

Population: CellEvent Caspase Objects	Value	
Number of Objects	14	
Property	**Mean**	**CV %**
Volume [µm³]	1943.73	57.8203
Volume [px³]	1132	57.8203

Stack

First Plan... 95.0 µm

Number o... 50

Distance: 3.6 µm

Last Plan... 271.4 µm

Overall H... 176.4 µm

Use in Test: □

Settings for image analysis using Phenix Harmony

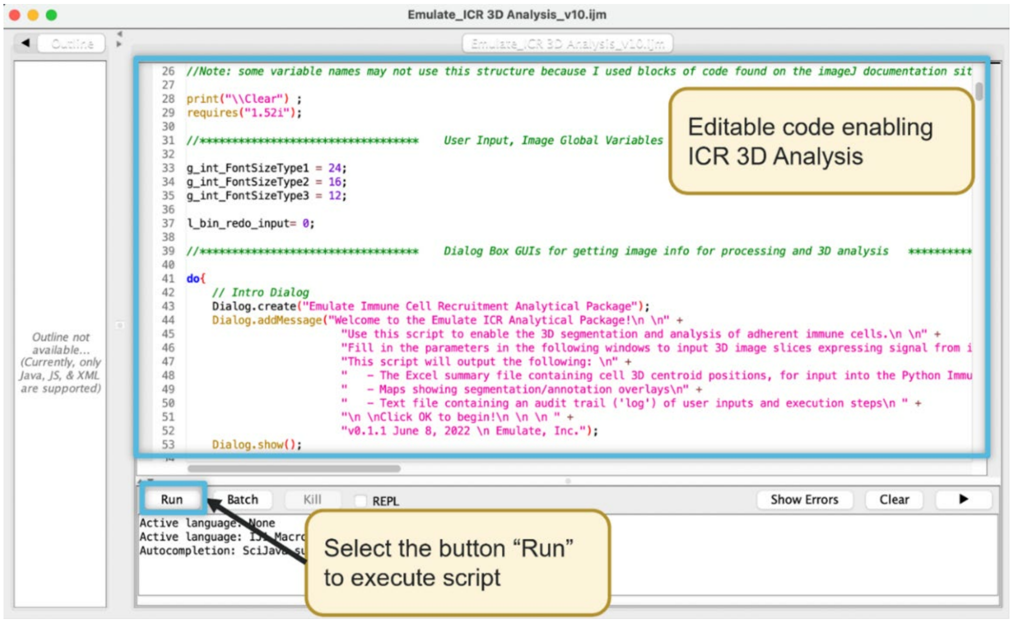
Protocol

Overview

This protocol shows the step-by-step process for running the ImageJ2/Fiji Script for 3D Object Counting to detect CAR T cells in 3D image stacks.

Steps

Step	Action
1	Open ImageJ2/Fiji.
2	Open and load the most recent version of ImageJ Script for 3D Object Counting available from Emulate support.
3	Create an Output Directory within the project folder. **Note:** The Output Directory **cannot** reside within the Source Directory.
4	Select the "Run" button in the macro window to run the script (See Figure 7).



Emulate_ICR 3D Analysis_v10.ijm
Outline
Emulate_ICR 3D Analysis_v10.ijm
Run
Batch
Kill
REPL
Show Errors
Clear

Outline not available... (Currently, only Java, JS, & XML are supported)

```
//Note: some variable names may not use this structure because I used blocks of code found on the imageJ documentation sit

print("\\Clear") ;
requires("1.52i");

//**********************************    User Input, Image Global Variables

g_int_FontSizeType1 = 24;
g_int_FontSizeType2 = 16;
g_int_FontSizeType3 = 12;

l_bin_redo_input= 0;

//**********************************    Dialog Box GUIs for getting image info for processing and 3D analysis   **********

do{
    // Intro Dialog
    Dialog.create("Emulate Immune Cell Recruitment Analytical Package");
    Dialog.addMessage("Welcome to the Emulate ICR Analytical Package!\n \n" +
                        "Use this script to enable the 3D segmentation and analysis of adherent immune cells.\n \n" +
                        "Fill in the parameters in the following windows to input 3D image slices expressing signal from i
                        "This script will output the following: \n" +
                        "   - The Excel summary file containing cell 3D centroid positions, for input into the Python Immu
                        "   - Maps showing segmentation/annotation overlays\n" +
                        "   - Text file containing an audit trail ('log') of user inputs and execution steps\n " +
                        "\n \nClick OK to begin!\n \n \n " +
                        "v0.1.1 June 8, 2022 \n Emulate, Inc.");
    Dialog.show();
```

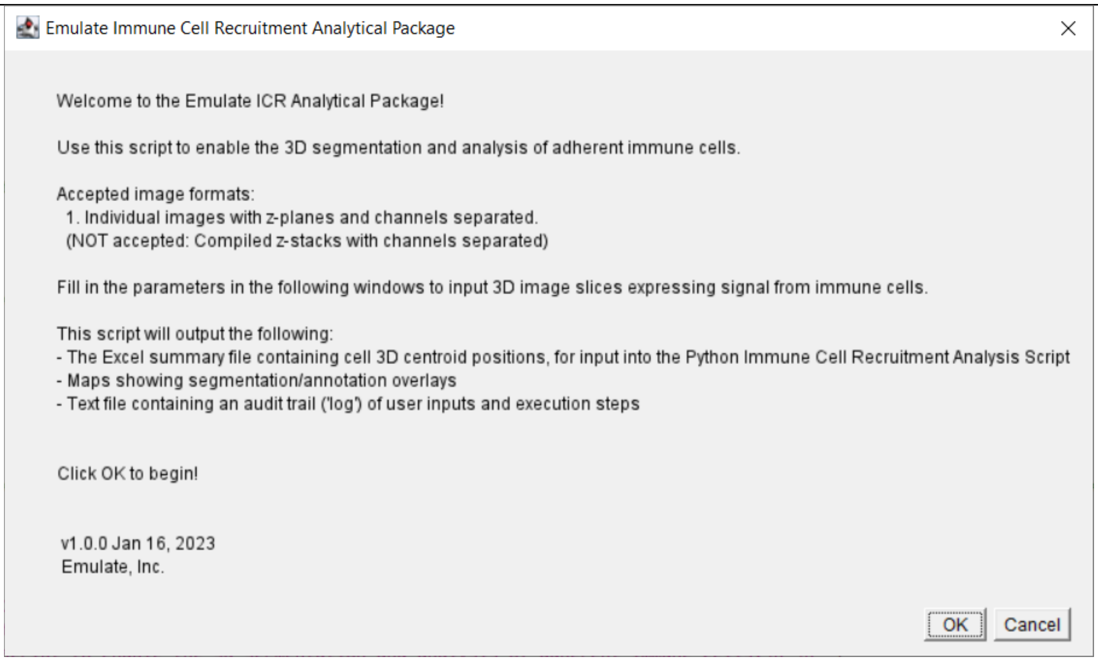
```
Active language: None
Active language: IJ1 Macro
Autocompletion: SciJava su
```

Editable code enabling ICR 3D Analysis

Select the button "Run" to execute script

Figure 7. Representative macro window as displayed in ImageJ2/FIJI.

Step	Action
5	A welcome window briefly describing the script will appear. Select the "OK" button to proceed (See Figure 8).



Emulate Immune Cell Recruitment Analytical Package

Welcome to the Emulate ICR Analytical Package!

Use this script to enable the 3D segmentation and analysis of adherent immune cells.

Accepted image formats:
1. Individual images with z-planes and channels separated.
(NOT accepted: Compiled z-stacks with channels separated)

Fill in the parameters in the following windows to input 3D image slices expressing signal from immune cells.

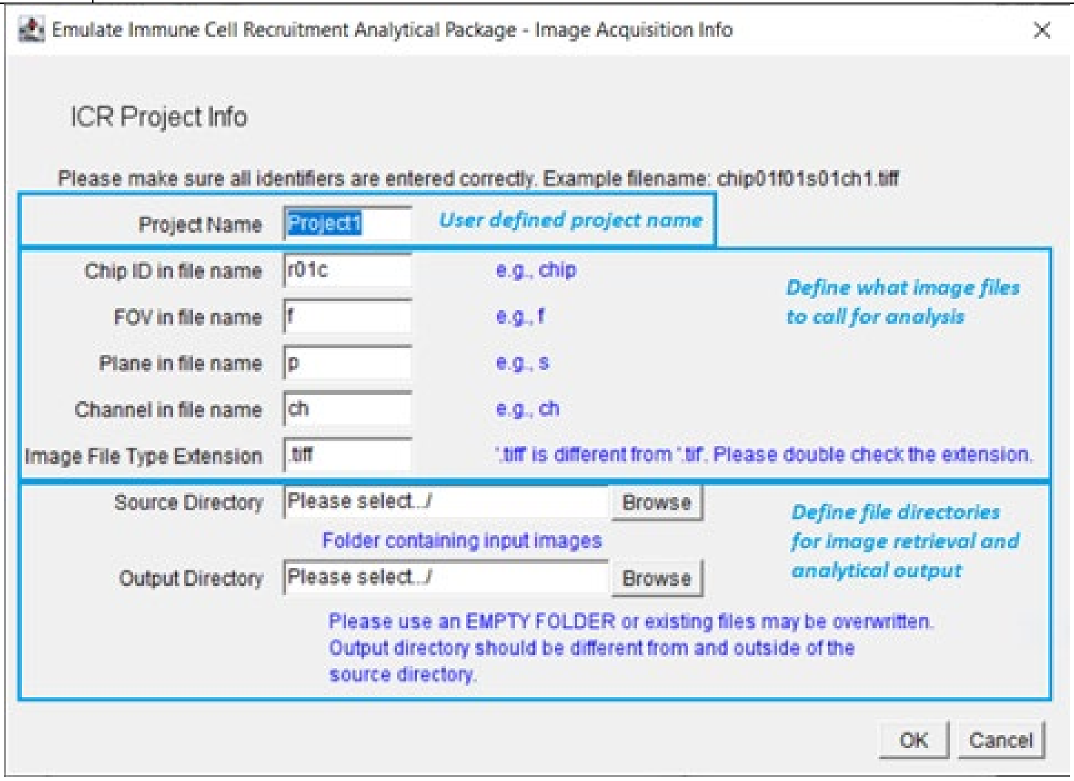
This script will output the following:
- The Excel summary file containing cell 3D centroid positions, for input into the Python Immune Cell Recruitment Analysis Script
- Maps showing segmentation/annotation overlays
- Text file containing an audit trail ('log') of user inputs and execution steps

Click OK to begin!

v1.0.0 Jan 16, 2023
Emulate, Inc.

Figure 8. Welcome window displaying script inputs and outputs.

6	The interface ***Image Acquisition Info*** will open (See Figure 9).



Emulate Immune Cell Recruitment Analytical Package - Image Acquisition Info

ICR Project Info

Please make sure all identifiers are entered correctly. Example filename: chip01f01s01ch1.tiff

Project Name: Project1 — *User defined project name*

Chip ID in file name: r01c — e.g., chip

FOV in file name: f — e.g., f

Plane in file name: p — e.g., s

Channel in file name: ch — e.g., ch

Image File Type Extension: .tiff — '.tiff' is different from '.tif'. Please double check the extension.

Define what image files to call for analysis

Source Directory: Please select.../ — Browse

Folder containing input images

Output Directory: Please select.../ — Browse

Define file directories for image retrieval and analytical output

Please use an EMPTY FOLDER or existing files may be overwritten. Output directory should be different from and outside of the source directory.

Figure 9. Defining project settings.

7	Enter the Project Name.

8	Enter the following information from the file name. This will enable the script to call image files to be analyzed: a. Chip ID in file name b. FOV in file name c. Plane in file name d. Channel in file name e. Image File Type Extension

IDENTIFIER	INCORRECT INPUT	CORRECT INPUT
Example Filename	r02c03f01p01-ch1sk1fk1fl1.tiff	
Chip ID	r02c03	**r02c**
FOV	f01	**f**
Plane	p01	**p**
Channel	ch1	**ch**
Image File Type Extension	.tif	**.tiff**

Table 4. Examples of correct and incorrect filename identifiers.

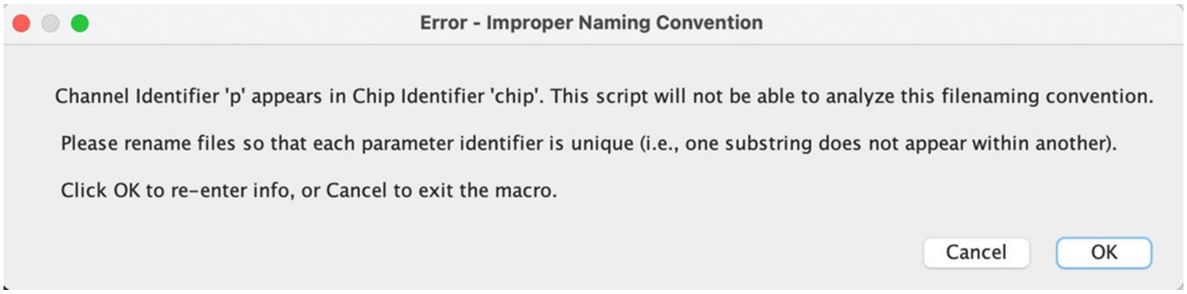
Note: If the Fixed Chip Imaging Adaptor was used for acquiring images, the chip position legend in table 5 can be used:

r01c01	r01c02	r01c03	r01c04	r01c05	r01c06
r02c01	r02c01	r02c03	r02c04	r02c05	r02c06

Table 5. Emulate Fixed Chip Imaging Adapter position legend with 12 chips, where each chip is identified by "r" = row and "c" = column based on position.

Note: All filename identifiers entered above must be unique. For instance, "chip01f01p01ch01.tiff" will not work, because the string "p01" appears within the string "chip01", so the filename identifiers are not unique. "c01f01p01ch01.tiff" will work; even though the identifiers are "c" and "ch", the string "c01" does not appear in the string "ch01", so there is no overlap. Please ensure all filenames follow a similarly unique convention.

Note: If filename identifiers have significant overlap, the script will throw an error prompt to rename the files as necessary (See Figure 10).



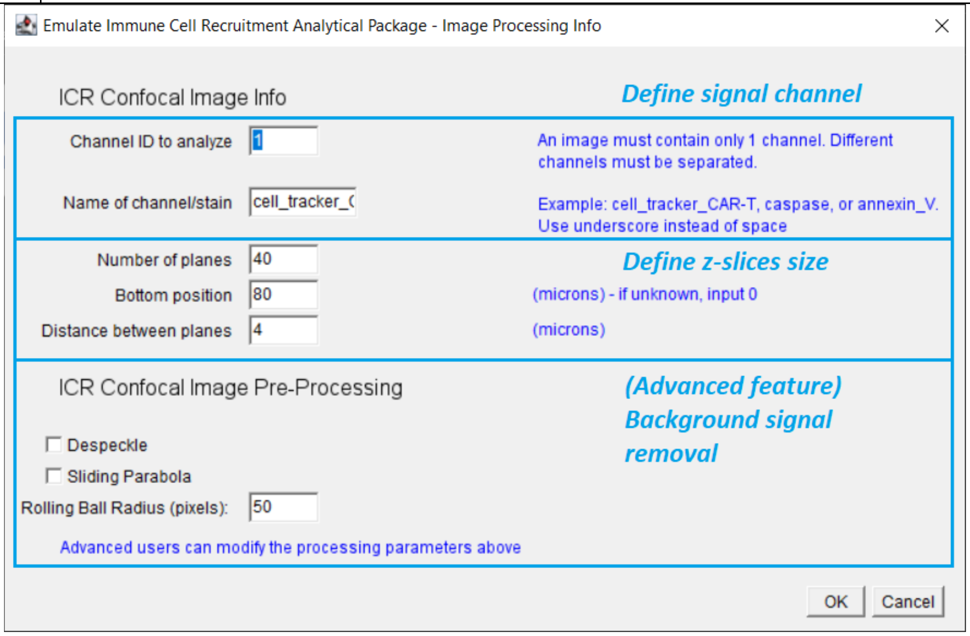
Error - Improper Naming Convention

Channel Identifier 'p' appears in Chip Identifier 'chip'. This script will not be able to analyze this filenaming convention.

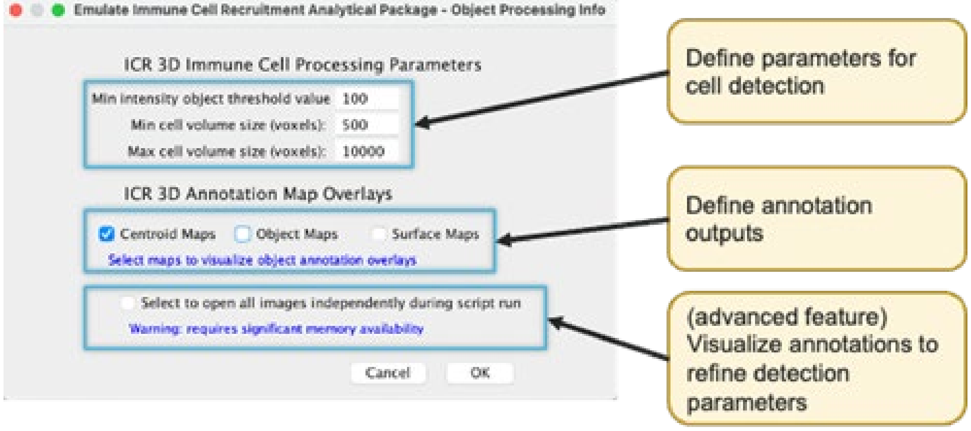
Please rename files so that each parameter identifier is unique (i.e., one substring does not appear within another).

Click OK to re-enter info, or Cancel to exit the macro.

Figure 10. Error warning displayed when incorrect filename identifiers are input.

9	Choose the Source Directory folder by selecting the “Browse” button. In the folder browser window, select the project folder containing the input images.
10	Choose the Output Directory folder by selecting the “Browse” button. Within the folder browser window, select the empty folder created in Step 3. The script will output all annotation maps and data files to this folder.
11	Select the button “OK” when all information has been entered.
12	The ***Image Processing Info Interface*** will open (See Figure 11).
	 Emulate Immune Cell Recruitment Analytical Package - Image Processing Info ICR Confocal Image Info *Define signal channel* Channel ID to analyze: 1 An image must contain only 1 channel. Different channels must be separated. Name of channel/stain: cell_tracker_(Example: cell_tracker_CAR-T, caspase, or annexin_V. Use underscore instead of space *Define z-slices size* Number of planes: 40 Bottom position: 80 (microns) - if unknown, input 0 Distance between planes: 4 (microns) ICR Confocal Image Pre-Processing *(Advanced feature) Background signal removal* □ Despeckle □ Sliding Parabola Rolling Ball Radius (pixels): 50 Advanced users can modify the processing parameters above Figure 11. Defining image settings.
13	Enter the index of the channel to analyze (e.g., 1, 4, 3).
14	Enter the name of the channel or the stain (e.g., cell_tracker_CAR-T, caspase, annexin_V).
15	Enter the number of planes acquired in the z-stack. **Note:** Refer to the “Key Input Parameters” for entering this value.
16	Enter the position, in µm, of the bottom plane of the z-stack. **Note:** Refer to the “Key Input Parameters” for entering this value.
17	Enter the distance, in µm, between the planes in the z-stack. **Note:** Refer to the “Key Input Parameters” for entering this value.
18	(Advanced) This protocol provides four options for pre-processing imagery to filter out background noise. Options include despeckle, subtract background, sliding parabola, and rolling ball radius. **Note:** As fluorescent signal is only generated from the CellTracker dye within a highly diffuse sample, imagery should have a large signal-to-noise ratio. If required, Emulate provides pre-processing algorithms.

19	Select the “OK” button when all information has been entered.
20	The ***Object Processing Info Interface*** will open (See Figure 12).



Emulate Immune Cell Recruitment Analytical Package - Object Processing Info

ICR 3D Immune Cell Processing Parameters

Min intensity object threshold value: 100

Min cell volume size (voxels): 500

Max cell volume size (voxels): 10000

ICR 3D Annotation Map Overlays

☑ Centroid Maps ☐ Object Maps ☐ Surface Maps

Select maps to visualize object annotation overlays

☐ Select to open all images independently during script run

Warning: requires significant memory availability

Define parameters for cell detection

Define annotation outputs

(advanced feature) Visualize annotations to refine detection parameters

Figure 12. Defining cell detection settings

21	Enter the following 3D processing parameter values: ● Min intensity object threshold value ● Min cell volume (reported in voxels) ● Max cell volume (reported in voxels) **Note:** Refer to the “Key Input Parameters” for entering these values.
22	Select which annotation map overlays (if any) should be saved in Output Directory. **Note:** Selection is not required, but the maps enable the review of segmented cells and adjustment of processing parameter values to optimize object identification.
23	Select the box next to the statement “Select to open all images independently during script run” to open each image as it is segmented. **Note:** This will require significant memory and is not recommended for large data sets.
24	Select the “OK” button to run the script. Outputs will populate in the Output Directory folder. Duration for script completion depends on the hardware specifications of the computer running the script.
25	Navigate to the output directory to check the outputs.
26	Repeat this process for the other stains/channels.

Error Messages

Causes

Messages	Possible Causes
Display an info message for the number of sets of images found. If the number is different from the actual number of images, the filename or directory may be wrong.	User may enter the wrong file naming convention/format.
Display a warning message "Script completed successfully. There are n sets of images where no objects are found. Please check thresholds and re-run the program before proceeding to the next steps."	Size thresholds are not appropriate for the data.
Display an error message “No object found in line 492.”	Intensity thresholds are not appropriate for the data.

Note: Please restart the PC for RAM-specific issues.

Setting Thresholds

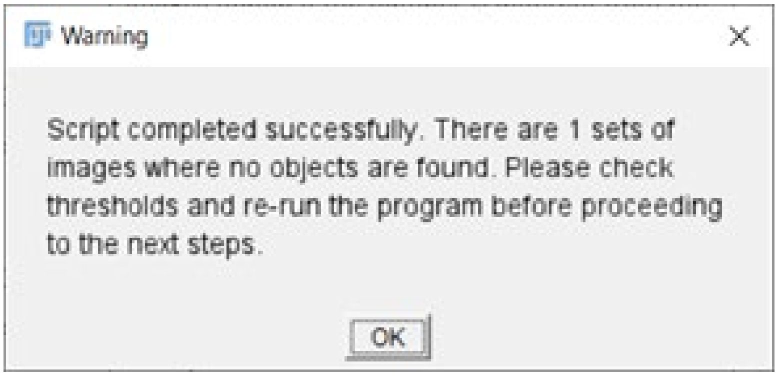
When the thresholds are set incorrectly, **assuming that objects are in fact present in the image**, there are three things that could happen:

Issue	Finding Objects	Script Execution	Results	Error Message
Thresholds are slightly off	The 3D object counter plugin can still find objects, but some artifacts are included, or real cell objects are excluded.	The script in ImageJ executes successfully.	Results are written to files but are not optimal.	None.
Size thresholds are incorrect	The 3D object counter plugin finds no objects.	The script in ImageJ executes successfully.	Results are written to files but are not optimal.	Please refer to figure 13.
Intensity thresholds are incorrect	The 3D object counter plugin finds no objects.	The script in ImageJ is interrupted by the 3D object counter plugin.	Results are not written to files.	Please refer to figure 14.

Continued on next page

Error Messages, Continued

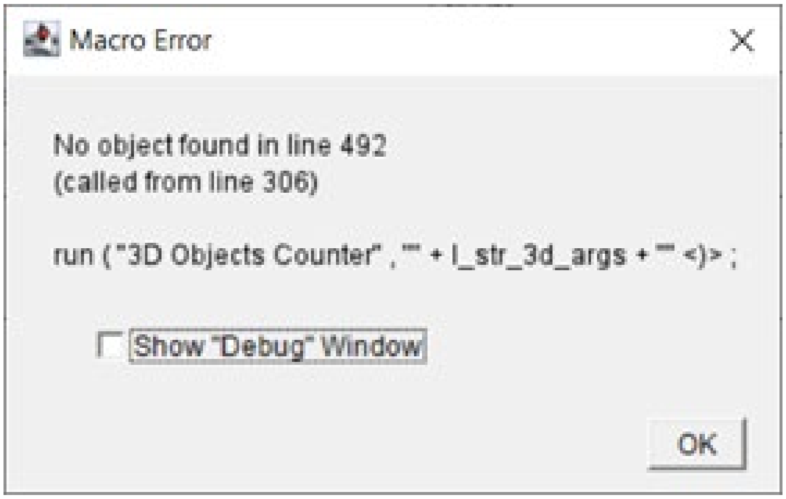
Figure 13



Script completed successfully. There are 1 sets of images where no objects are found. Please check thresholds and re-run the program before proceeding to the next steps.

Error message due to incorrect size thresholds.

Figure 14



```
No object found in line 492
(called from line 306)

run ("3D Objects Counter" , "" + l_str_3d_args + "" <)> ;
```

Error message due to incorrect intensity thresholds.

Output Data

Overview

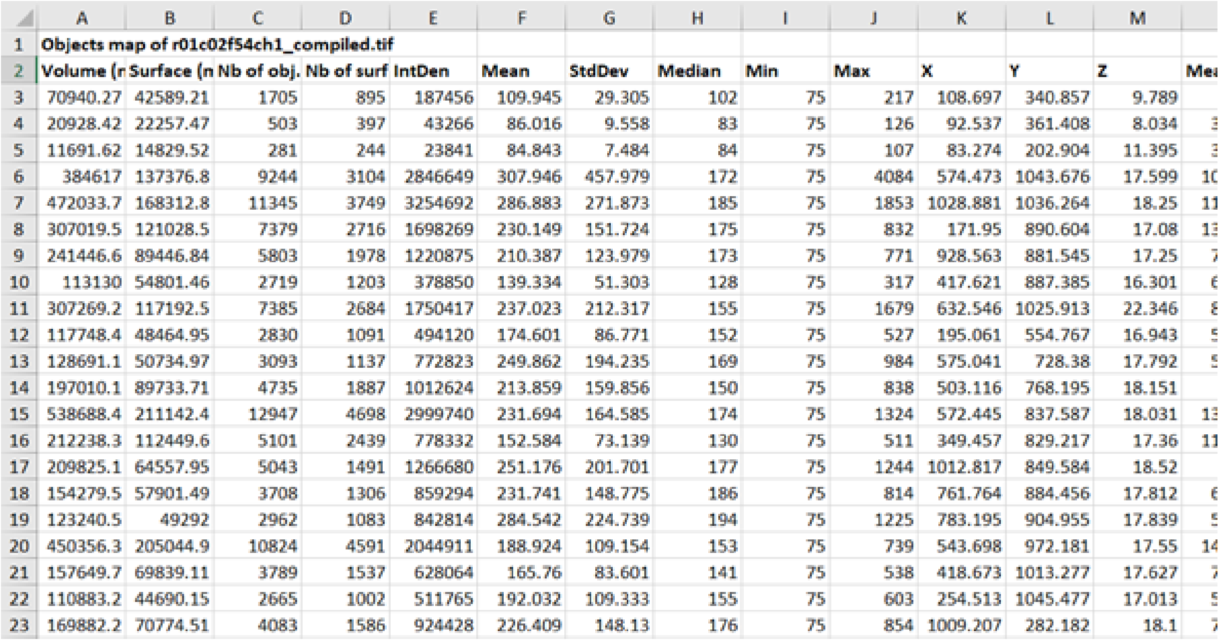
The outputs are statistics (in the form of Excel spreadsheets) and annotated maps (in the form of 3D image stacks, TIFF format) all about the detected objects. The ImageJ script generates an Excel spreadsheet for each image channel, which is essential for downstream analysis, i.e., feature analysis and co-localization.

There are three types of annotated maps: centroid maps, object maps, and surface maps. The object maps display the complete set of pixels identified in each object, where each pixel is labelled with the object index. This type of map provides the most valuable information and is also essential for downstream analysis.

Outputs

The ImageJ Script for 3D Object Counting will output the following:

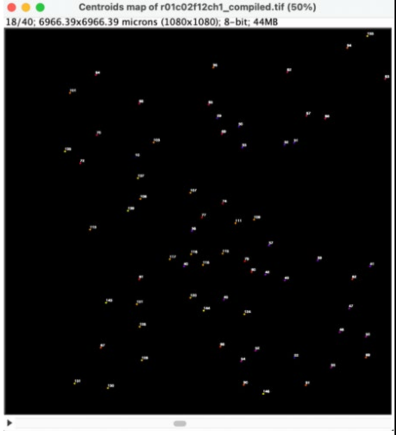
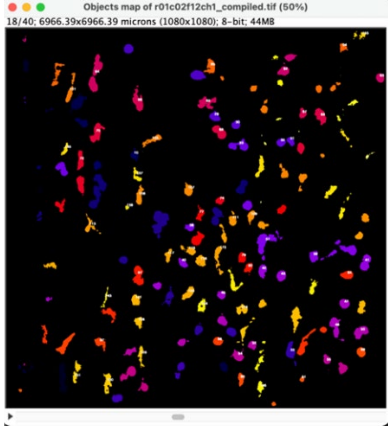
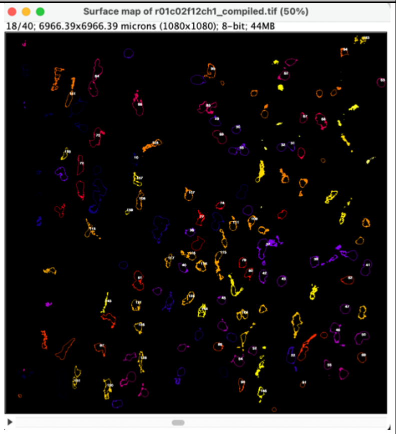
1. Excel file containing object statistics, where the X, Y, and Z columns are per cell centroid coordinates in µm.



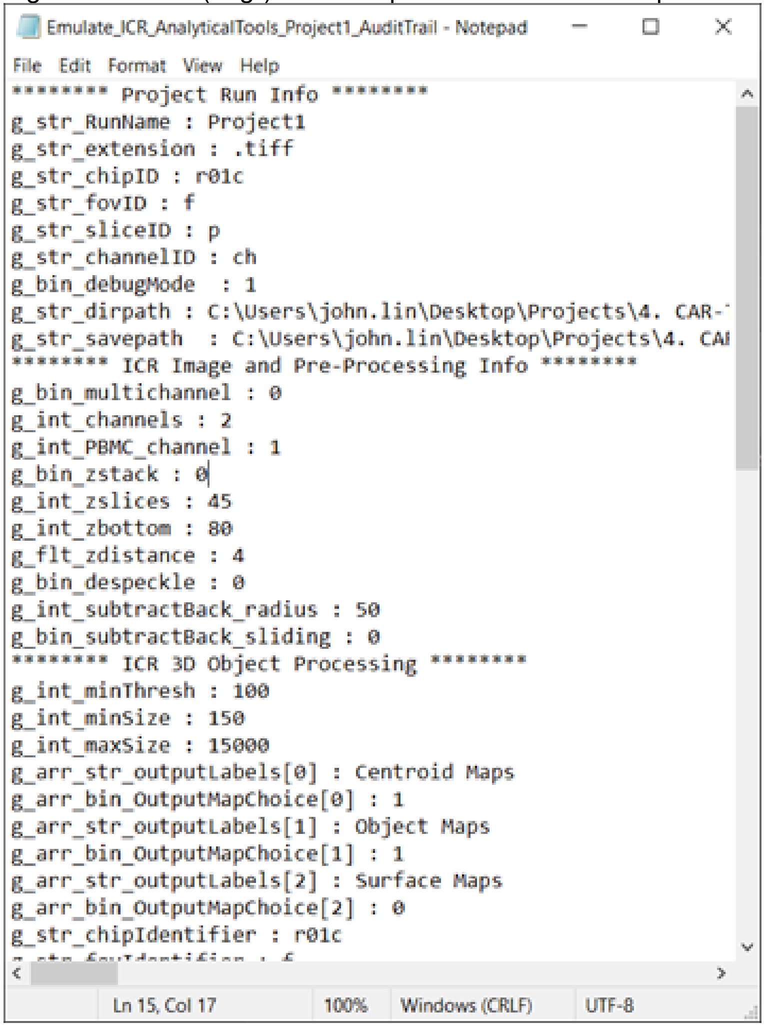
	A	B	C	D	E	F	G	H	I	J	K	L	M	
1	Objects map of r01c02f54ch1_compiled.tif													
2	Volume (r	Surface (n	Nb of obj.	Nb of surf	IntDen	Mean	StdDev	Median	Min	Max	X	Y	Z	Me
3	70940.27	42589.21	1705	895	187456	109.945	29.305	102	75	217	108.697	340.857	9.789	
4	20928.42	22257.47	503	397	43266	86.016	9.558	83	75	126	92.537	361.408	8.034	3
5	11691.62	14829.52	281	244	23841	84.843	7.484	84	75	107	83.274	202.904	11.395	3
6	384617	137376.8	9244	3104	2846649	307.946	457.979	172	75	4084	574.473	1043.676	17.599	10
7	472033.7	168312.8	11345	3749	3254692	286.883	271.873	185	75	1853	1028.881	1036.264	18.25	11
8	307019.5	121028.5	7379	2716	1698269	230.149	151.724	175	75	832	171.95	890.604	17.08	13
9	241446.6	89446.84	5803	1978	1220875	210.387	123.979	173	75	771	928.563	881.545	17.25	7
10	113130	54801.46	2719	1203	378850	139.334	51.303	128	75	317	417.621	887.385	16.301	6
11	307269.2	117192.5	7385	2684	1750417	237.023	212.317	155	75	1679	632.546	1025.913	22.346	8
12	117748.4	48464.95	2830	1091	494120	174.601	86.771	152	75	527	195.061	554.767	16.943	5
13	128691.1	50734.97	3093	1137	772823	249.862	194.235	169	75	984	575.041	728.38	17.792	5
14	197010.1	89733.71	4735	1887	1012624	213.859	159.856	150	75	838	503.116	768.195	18.151	
15	538688.4	211142.4	12947	4698	2999740	231.694	164.585	174	75	1324	572.445	837.587	18.031	13
16	212238.3	112449.6	5101	2439	778332	152.584	73.139	130	75	511	349.457	829.217	17.36	11
17	209825.1	64557.95	5043	1491	1266680	251.176	201.701	177	75	1244	1012.817	849.584	18.52	
18	154279.5	57901.49	3708	1306	859294	231.741	148.775	186	75	814	761.764	884.456	17.812	6
19	123240.5	49292	2962	1083	842814	284.542	224.739	194	75	1225	783.195	904.955	17.839	5
20	450356.3	205044.9	10824	4591	2044911	188.924	109.154	153	75	739	543.698	972.181	17.55	14
21	157649.7	69839.11	3789	1537	628064	165.76	83.601	141	75	538	418.673	1013.277	17.627	7
22	110883.2	44690.15	2665	1002	511765	192.032	109.333	155	75	603	254.513	1045.477	17.013	5
23	169882.2	70774.51	4083	1586	924428	226.409	148.13	176	75	854	1009.207	282.182	18.1	7

A sample Excel spreadsheet of object statistics.

2. Maps showing segmentation and/or annotation overlays.

Centroid	Object	Surface
 Centroids map of r01c02f12ch1_compiled.tif (50%) 18/40; 6966.39x6966.39 microns (1080x1080); 8-bit; 44MB	 Objects map of r01c02f12ch1_compiled.tif (50%) 18/40; 6966.39x6966.39 microns (1080x1080); 8-bit; 44MB	 Surface map of r01c02f12ch1_compiled.tif (50%) 18/40; 6966.39x6966.39 microns (1080x1080); 8-bit; 44MB

3. Text file containing an audit trail ("log") of user inputs and execution steps.



Emulate_ICR_AnalyticalTools_Project1_AuditTrail - Notepad
File Edit Format View Help
Ln 15, Col 17
100%
Windows (CRLF)
UTF-8

```
******** Project Run Info ********
g_str_RunName : Project1
g_str_extension : .tiff
g_str_chipID : r01c
g_str_fovID : f
g_str_sliceID : p
g_str_channelID : ch
g_bin_debugMode  : 1
g_str_dirpath : C:\Users\john.lin\Desktop\Projects\4. CAR-
g_str_savepath  : C:\Users\john.lin\Desktop\Projects\4. CAI
******** ICR Image and Pre-Processing Info ********
g_bin_multichannel : 0
g_int_channels : 2
g_int_PBMC_channel : 1
g_bin_zstack : 0
g_int_zslices : 45
g_int_zbottom : 80
g_flt_zdistance : 4
g_bin_despeckle : 0
g_int_subtractBack_radius : 50
g_bin_subtractBack_sliding : 0
******** ICR 3D Object Processing ********
g_int_minThresh : 100
g_int_minSize : 150
g_int_maxSize : 15000
g_arr_str_outputLabels[0] : Centroid Maps
g_arr_bin_OutputMapChoice[0] : 1
g_arr_str_outputLabels[1] : Object Maps
g_arr_bin_OutputMapChoice[1] : 1
g_arr_str_outputLabels[2] : Surface Maps
g_arr_bin_OutputMapChoice[2] : 0
g_str_chipIdentifier : r01c
g_str_fovIdentifier : f
```

Running JupyterLab

Overview

Introduction

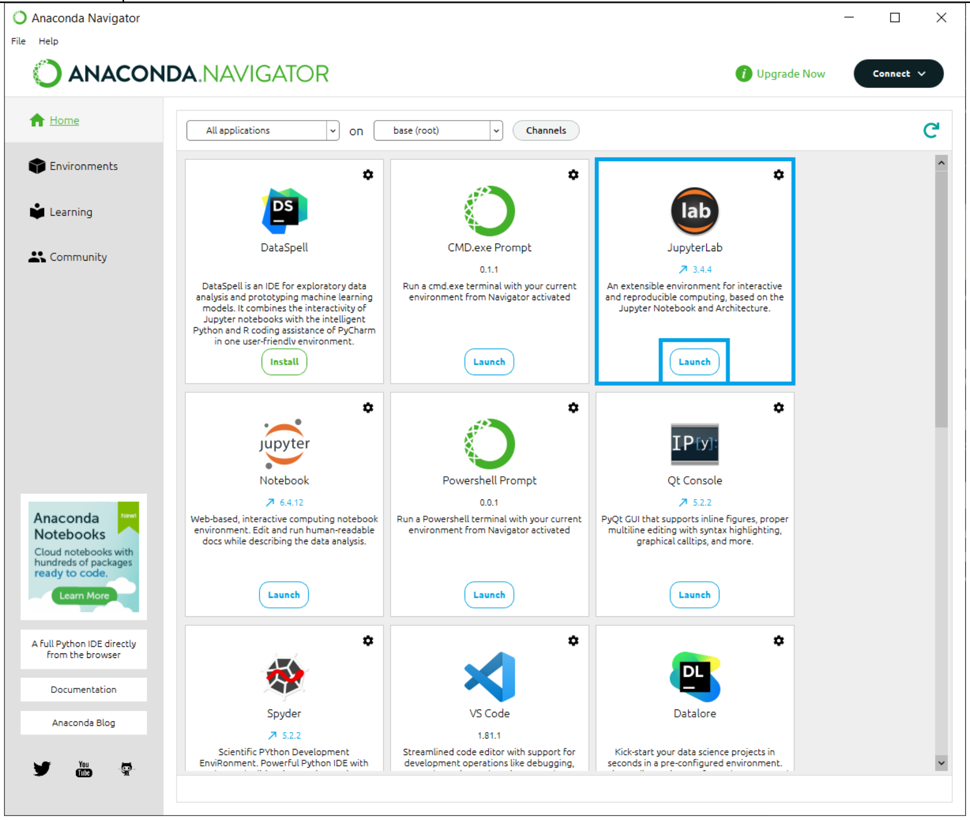
JupyterLab is a browser-based interactive computing platform that permits the visualization of migratory plots that project immune cells and their recruitment status with respect to their location within the xz-plane.

Contents

Protocol

Steps

Step	Action
1	Open the Anaconda Navigator (See Figure 15).



DataSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jupyter notebooks with the intelligent Python and R coding assistance of PyCharm in one user-friendlv environment.

Run a cmd.exe terminal with your current environment from Navigator activated

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Run a Powershell terminal with your current environment from Navigator activated

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with

Streamlined code editor with support for development operations like debugging,

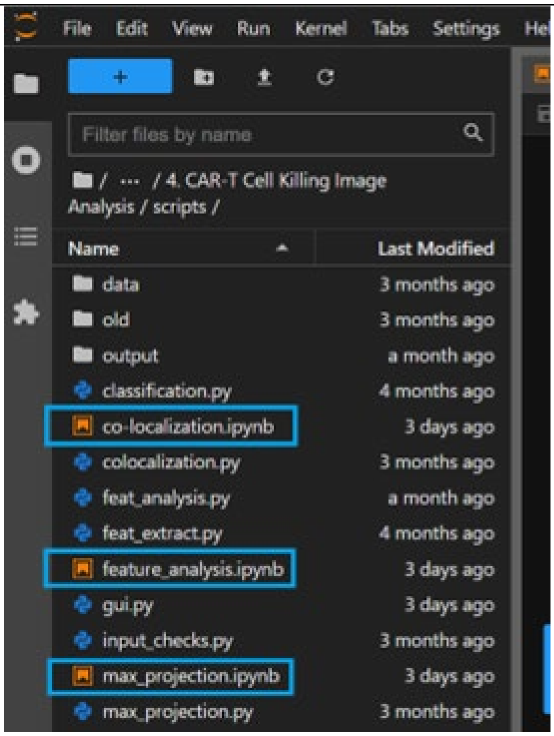
Kick-start your data science projects in seconds in a pre-configured environment.

Cloud notebooks with hundreds of packages ready to code.

A full Python IDE directly from the browser

Figure 15. The Anaconda Navigator is a desktop application that enables easy management of open-source Python tools without using the command line.

2	Select “Launch” button on the JupyterLab module.
3	JupyterLab will open.
4	In the file browser, navigate to the folder that contains the Python programs.
5	Double-click to open co-localization.ipynb, feature_analysis.ipynb, and max_projection.ipynb (See Figure 16).



/ ··· / 4. CAR-T Cell Killing Image Analysis / scripts /

Name	Last Modified
data	3 months ago
old	3 months ago
output	a month ago
classification.py	4 months ago
co-localization.ipynb	3 days ago
colocalization.py	3 months ago
feat_analysis.py	a month ago
feat_extract.py	4 months ago
feature_analysis.ipynb	3 days ago
gui.py	3 days ago
input_checks.py	3 months ago
max_projection.ipynb	3 days ago
max_projection.py	3 months ago

Figure 16. View of folder containing co-localization.ipynb, feature_analysis.ipynb, and max_projection.ipynb programs.

Feature Analysis for CAR T Recruitment

Overview

Introduction

The step after 3D object counting is feature analysis, which leverages the object statistics provided by the ImageJ script to perform a more in-depth analysis of various features of the objects, e.g., counts, volume, intensity, etc., per FOV and per chip. This process can be further broken down into 4 steps:

1. Classification
2. Feature extraction
3. Feature analysis
4. Plotting

These analysis steps are done on a per-channel basis and are repeated for all channels of stains.

Figure 17

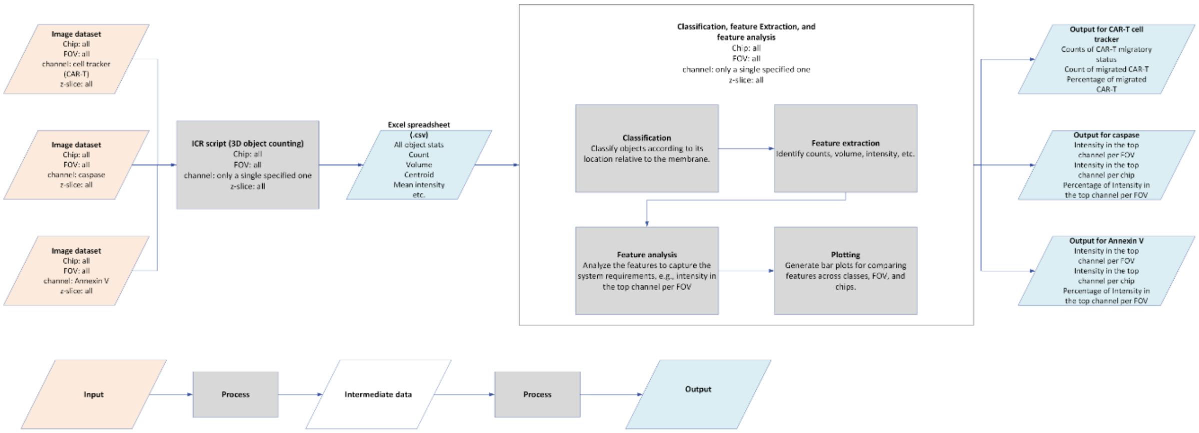


Image dataset
Chip: all
FOV: all
channel: cell tracker (CAR-T)
z-slice: all
Image dataset
Chip: all
FOV: all
channel: caspase
z-slice: all
Image dataset
Chip: all
FOV: all
channel: Annexin V
z-slice: all
ICR script (3D object counting)
Chip: all
FOV: all
channel: only a single specified one
z-slice: all
Excel spreadsheet (.csv)
All object stats
Count
Volume
Centroid
Mean intensity
etc.
Classification, feature Extraction, and feature analysis
Chip: all
FOV: all
channel: only a single specified one
z-slice: all
Classification
Classify objects according to its location relative to the membrane.
Feature extraction
Identify counts, volume, intensity, etc.
Feature analysis
Analyze the features to capture the system requirements, e.g., intensity in the top channel per FOV
Plotting
Generate bar plots for comparing features across classes, FOV, and chips.
Output for CAR-T cell tracker
Counts of CAR-T migratory status
Count of migrated CAR-T
Percentage of migrated CAR-T
Output for caspase
Intensity in the top channel per FOV
Intensity in the top channel per chip
Percentage of Intensity in the top channel per FOV
Output for Annexin V
Intensity in the top channel per FOV
Intensity in the top channel per chip
Percentage of Intensity in the top channel per FOV
Input
Process
Intermediate data
Process
Output

A flowchart illustrating the processes of classification, feature extraction, and analysis.

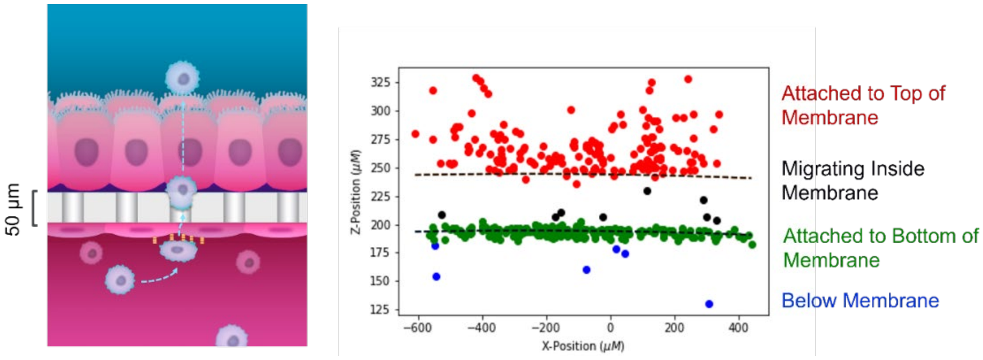
Contents

User Determination of z-Range to Assist Classification

Overview

Since recruitment status is classified based on CAR T-cell migratory patterns within the Organ-Chip, the **Feature Analysis** program for recruitment needs to estimate locations corresponding to the membrane edges. Assuming a large attachment of immune cells along the bottom of the membrane, the cellular clustering pattern can be utilized to fit a curved line in estimation of the membrane's bottom edge (See Figure 18). The upper channel starts + 50 µm above this height, which corresponds to the membrane's upper edge (See Figure 19).

Figure 18



Visualization of the classification of immune cells by z-position relative to the membrane

Note on z-range Requirements

In experiments with lower attachment densities, the accurate detection of the membrane edges can be difficult (See Figure 20). To ensure accuracy in membrane edge detection, regardless of recruitment levels, the Python program requires users to input an expected z-range for the membrane.

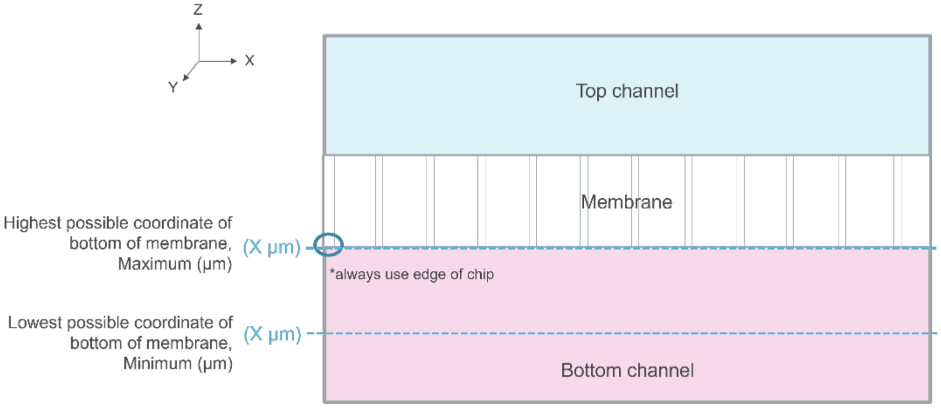
Steps

Membrane edge detection can be estimated with the following steps:

Step	Action
1	Open a representative z-stack in ImageJ2 or a similar viewer.
2	View the lowest plane, in depth, represented within the z-stack.
3	Scroll through the z-planes such that each plane gets progressively higher, in depth, within the z-stack.
4	Record the location (z-depth, in µm) of the bottom edge of the membrane. Depending on image type, the following scenarios will be visible: ● Fluorescence only – aggregation of fluorescence signal from immune cells adhered to the bottom of the membrane. Cells will resemble "pancakes", stretched in the xy-plane and short along the z-plane.

	• Brightfield (if image stack includes a reference channel) – membrane pores become visible. • Brightfield + Fluorescence – visualize both membrane pores and fluorescence signal from adhered cells. **Note:** For the most accurate results, record the value at the edge of the co-culture channel instead of the center of the membrane.
5	For the location of the upper membrane, it is recommended to add 60 µm to the z-location found for the bottom edge of the membrane.

Figure 19 **The Maximum (µm) and Minimum (µm) of the membrane z-range defines the range of coordinates in the z-axis where the lower boundary/edge of the membrane is expected to be.**


Z
X
Y
Top channel
Membrane
Highest possible coordinate of bottom of membrane, Maximum (µm)
(X µm)
*always use edge of chip
Lowest possible coordinate of bottom of membrane, Minimum (µm)
(X µm)
Bottom channel

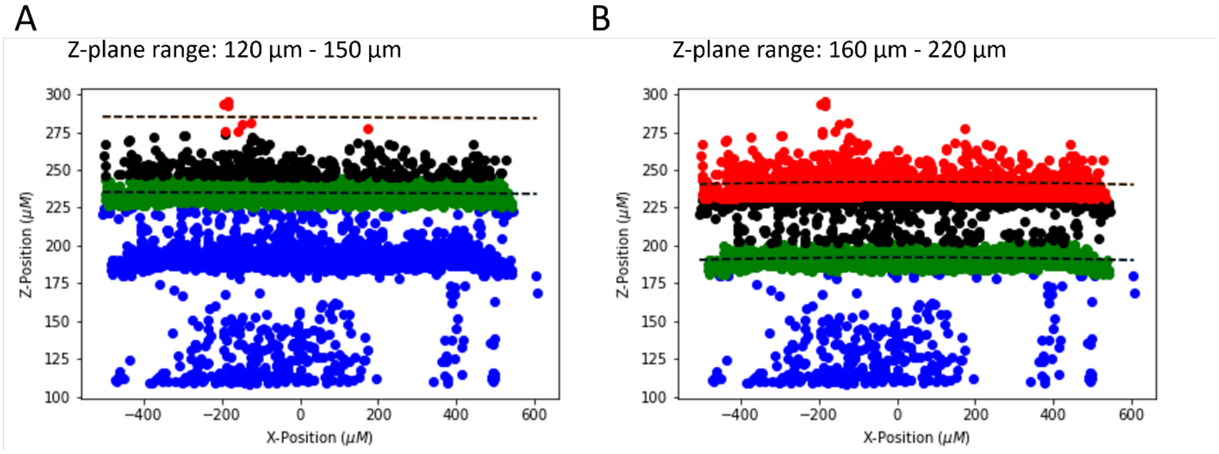
Defining potential membrane location range for correct membrane identification.

Note: If the output XY categorization plots have not estimated the membrane location with respect to cell clustering in the endothelial and epithelial layers, adjust the z-plane range so that the program more accurately categorizes the immune cell objects.

Continued on next page

User Determination of z-Range to Assist Classification, Continued

Figure 20



Program output xz plots demonstrating z-plane range effect on immune cell categorization.

A) Plot with z-plane range incorrectly set to 120–150 µm, where two distinct layers of cells are visible at ~180 µm and ~230 µm (50-µm membrane distance apart). These two uniform layers indicate that the membrane is lower than estimated and is better fit using the z-plane range in B) 160–220 µm as the bottom start of the membrane.

Algorithm

Overview

The first step is estimating the membrane shape and location for each chip using coordinates of object centroids aggregated from all FOV of each chip. The estimation can be done using the “caspase” or “cell tracker for CAR T” channel. The latter one is used for the estimates of the current analysis. It uses a sliding window approach to find the location in which the window with a height of 30 µm contains the maximum number of data points.

Then, a 2nd degree polynomial is fitted to the data points in that window to estimate the location and shape, which correspond to the bottom end of the membrane. The rationale is that a large proportion of the cells in the bottom channel of the chip adheres to the bottom of the membrane. Therefore, the cluster of membrane-adhered cells is a proxy for estimation.

The top end of the membrane is calculated as 50 µm above the bottom end. This process is repeated for each chip. The coordinates of object centroids are taken from the “X,” “Y,” and “Z-pos (micron)” columns in the in the Excel spreadsheet of object statistics from the ImageJ script. Membrane estimates are done on a per-chip basis using a single channel of stain (e.g., cell tracker for CAR T). Once this is done, the same estimates are used for the analysis of the other channels of stains.

Input Data

Overview

The inputs are the Excel spreadsheet from the ImageJ script in the 3D object counting step, one for each channel.

Object Statistics from the ImageJ Script

Key and required fields in the in the Excel spreadsheet of object statistics from the ImageJ script:

Field	Description
“Chip ID”	Separated by sheets/tabs
“FOV ID”	N/A
“NV of obj. voxels”	Number of object voxels in 3D
“Volume (micron^3)”	Total volume of the object. The values are the number of object voxels multiplied by a scalar.
“Mean”	Mean pixel intensity of the object
“IntDen”	Total pixel intensity of the object
“X,” “Y,” and “Z-pos (micron)”	The coordinates of the object centroid

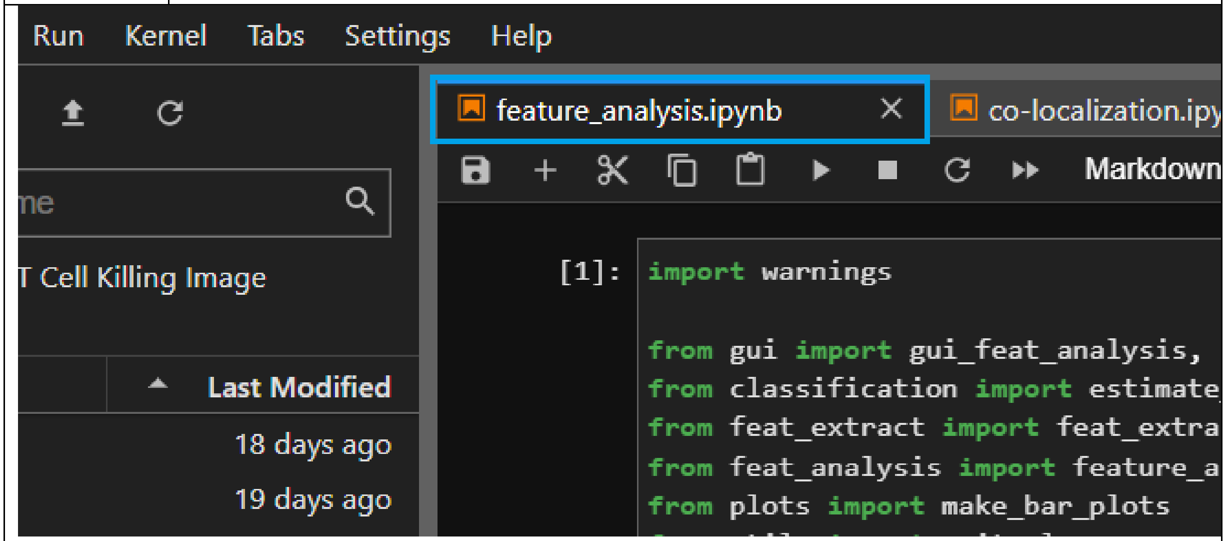
Protocol

Overview

The following protocol shows the step-by-step process for running the Feature Analysis program for CAR T recruitment.

Steps

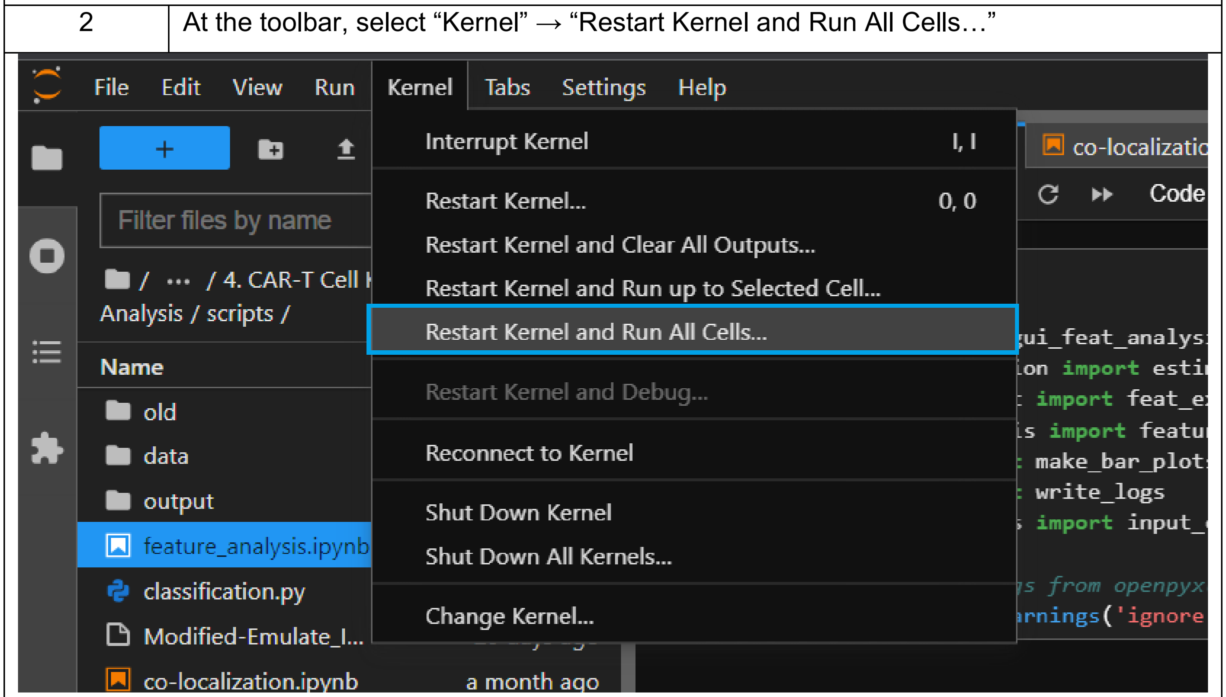
Step	Action
1	In JupyterLab, select the feature_analysis.ipynb tab.



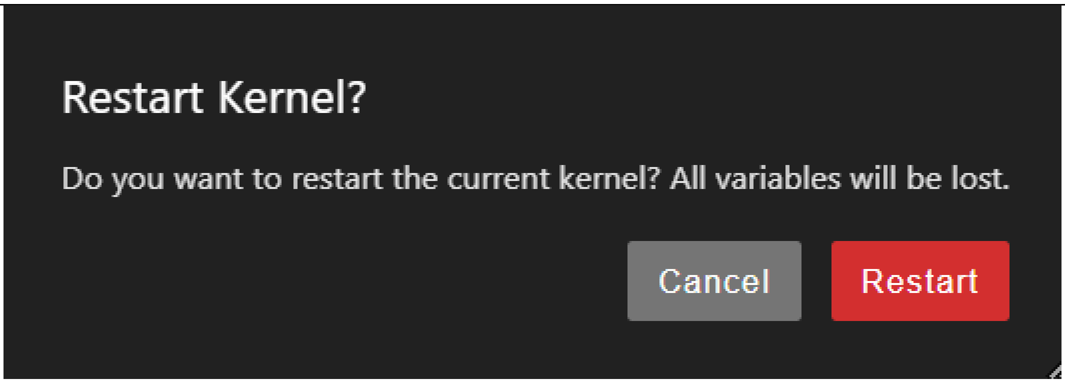
```
[1]: import warnings

     from gui import gui_feat_analysis,
     from classification import estimate
     from feat_extract import feat_extra
     from feat_analysis import feature_a
     from plots import make_bar_plots
```

Step	Action
2	At the toolbar, select “Kernel” → “Restart Kernel and Run All Cells…”



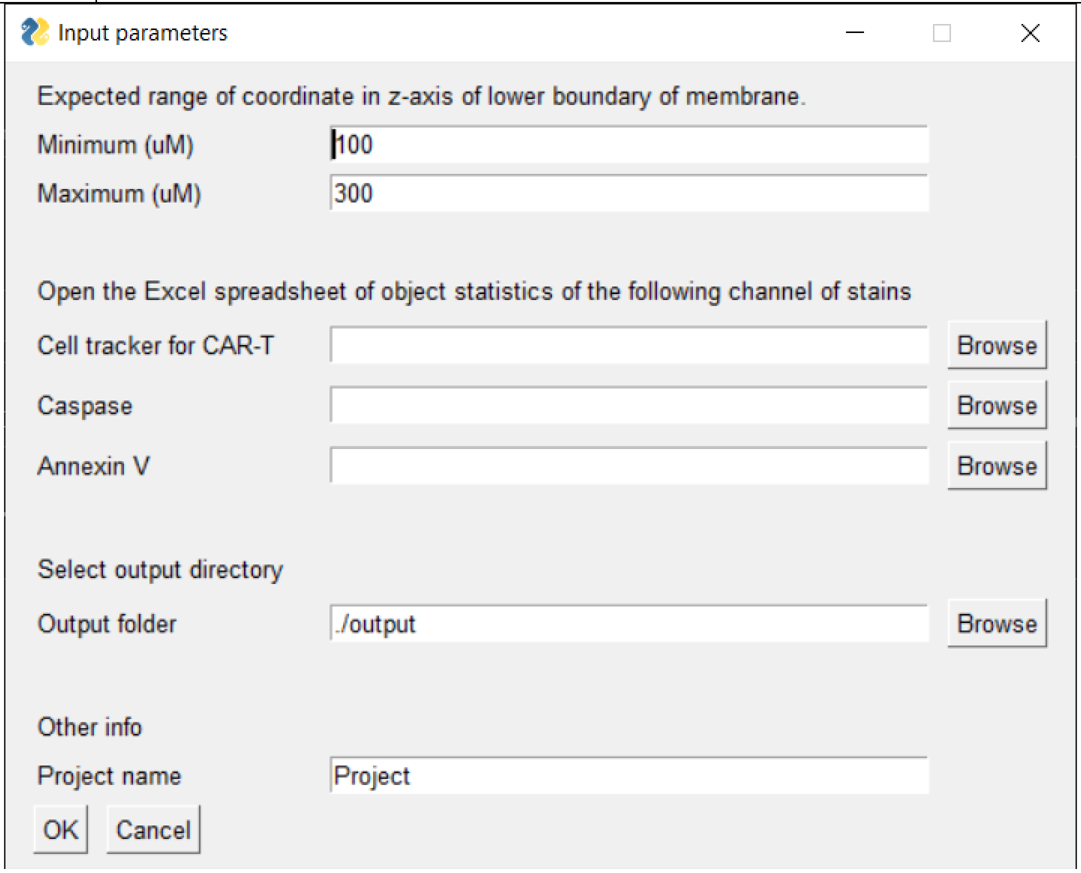
Step	Action
3	If a “Restart Kernel?” dialog pops up, select “Restart.”



Restart Kernel?

Do you want to restart the current kernel? All variables will be lost.

4 A GUI window will pop up asking the user to input parameters.



Input parameters

Expected range of coordinate in z-axis of lower boundary of membrane.

Minimum (uM) 100

Maximum (uM) 300

Open the Excel spreadsheet of object statistics of the following channel of stains

Cell tracker for CAR-T

Caspase

Annexin V

Select output directory

Output folder ./output

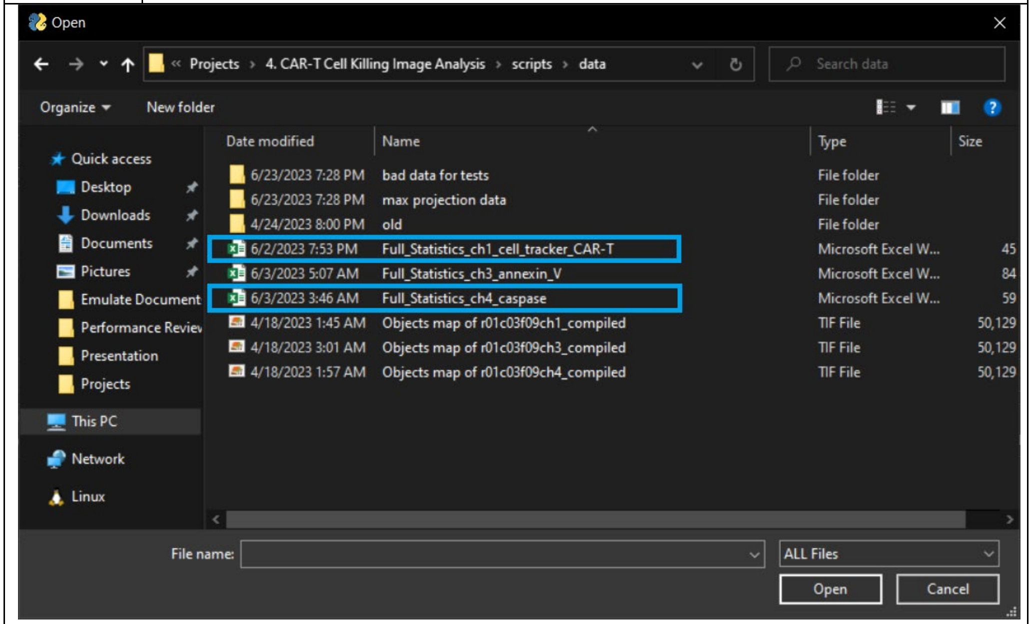
Other info

Project name Project

5 Enter the following parameter values:

Step	Action
1	Enter the estimated minimum and maximum values in the z-axis where the bottom edge of the membrane may be found. These values can be determined in the “User Determination of z-Range to Assist Classification” section. The default values can be used or modified if necessary.

	2	Click “Browse” to locate and select the **Excel spreadsheet of object statistics**, created by the ImageJ Script for 3D Object Counting in the previous section, for the following channels: ● Cell tracker for CAR T ● Caspase



Open

« Projects › 4. CAR-T Cell Killing Image Analysis › scripts › data

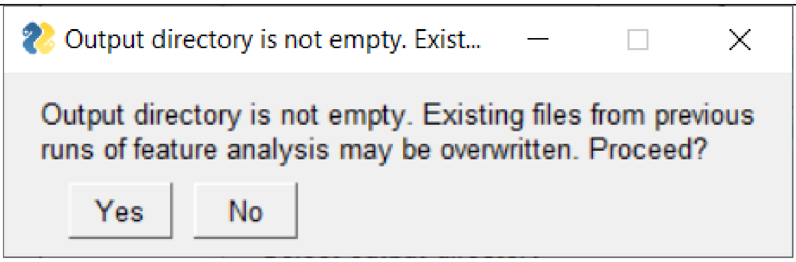
Search data

Date modified	Name	Type	Size
6/23/2023 7:28 PM	bad data for tests	File folder	
6/23/2023 7:28 PM	max projection data	File folder	
4/24/2023 8:00 PM	old	File folder	
6/2/2023 7:53 PM	Full_Statistics_ch1_cell_tracker_CAR-T	Microsoft Excel W...	45
6/3/2023 5:07 AM	Full_Statistics_ch3_annexin_V	Microsoft Excel W...	84
6/3/2023 3:46 AM	Full_Statistics_ch4_caspase	Microsoft Excel W...	59
4/18/2023 1:45 AM	Objects map of r01c03f09ch1_compiled	TIF File	50,129
4/18/2023 3:01 AM	Objects map of r01c03f09ch3_compiled	TIF File	50,129
4/18/2023 1:57 AM	Objects map of r01c03f09ch4_compiled	TIF File	50,129

File name:

ALL Files

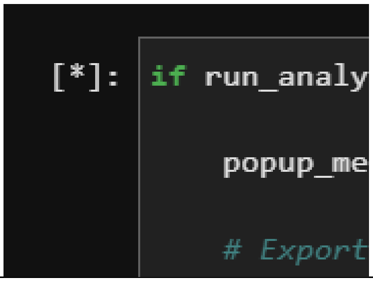
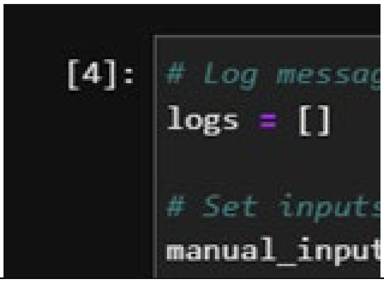
	3	Click “Browse” to locate and select the output directory (or, if it does not exist, create one). **Note:** A sub-folder “feature_analysis_output” will be automatically created under this selected folder to store all outputs from this analysis. **Be sure to use different output directories to store the results of different runs/experiments/projects.**
	4	Enter the desired project name. The default value can be used.
	5	Click “OK” when done.
6	If the output directory is not empty, existing files may be overwritten. Click “Yes” to start the analysis and overwrite existing files, or “No” to return to the previous step to select/create a different output folder.	

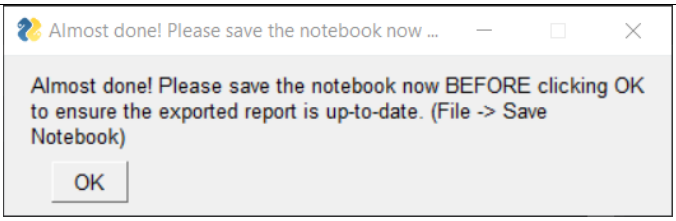


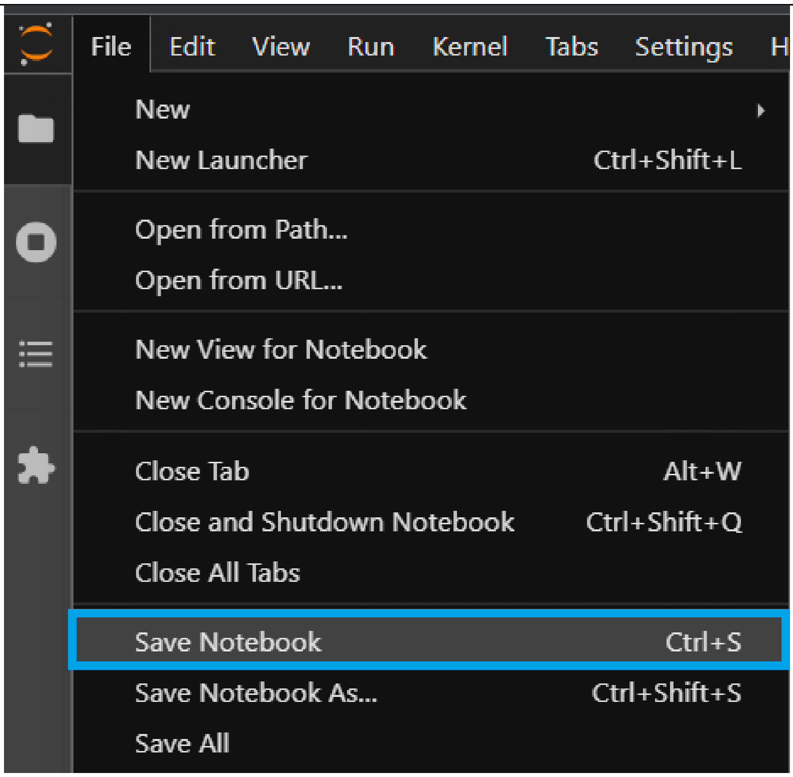
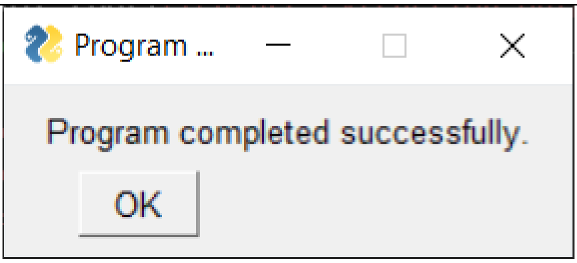
Output directory is not empty. Exist...

Output directory is not empty. Existing files from previous runs of feature analysis may be overwritten. Proceed?

7	While the analysis is running, the results will start populating in the notebook. Keep scrolling to view and follow the outputs while they are being generated. **Note:** When a code block is running, there is an asterisk on the upper left. When the code block is done, a number (execution order) will replace the asterisk.

	 `[*]: if run_analy`
	`popup_me`
	`# Export`
	 `[4]: # Log messag`
	`logs = []`
	`# Set inputs`
	`manual_input`

8	**(Important) When the analysis is almost complete, the following message will pop up. Follow the instructions to save the notebook by selecting "File" → "Save Notebook". This ensures the exported report is up to date.** **Note:** If the Jupyter Notebook platform is used instead of the recommended JupyterLab platform, select "Save and Checkpoint" instead of "Save Notebook."
	 Almost done! Please save the notebook now BEFORE clicking OK to ensure the exported report is up-to-date. (File -> Save Notebook)

 File Edit View Run Kernel Tabs Settings H New New Launcher Ctrl+Shift+L Open from Path... Open from URL... New View for Notebook New Console for Notebook Close Tab Alt+W Close and Shutdown Notebook Ctrl+Shift+Q Close All Tabs Save Notebook Ctrl+S Save Notebook As... Ctrl+Shift+S Save All	
9	Finally, if the program is completed successfully, the following message will appear. If this message does not show up, check the outputs in the notebook to see if the program is still running or has encountered an error.
 Program completed successfully.	

Error Messages

Messages and Causes

Messages	Possible Causes
Display a warning message "Please fill out all fields."	Some inputs in GUI are not filled out.
Display a dialog "Do you really want to exit?"	User accidentally clicks "x" or "Cancel."
Display an info message "Analysis canceled."	User decides to cancel.
If any sheets contain no entries, display an error message, and stop the analysis. The user needs to remove that chip ID from all Excel spreadsheets.	When the 3D object counter in the ImageJ script cannot find any objects in a chip, the sheet of that chip in the Excel spreadsheet will have no entries.
If values of the range are not valid, display an error message and stop the analysis. The user needs to adjust the Maximum (µm) and Minimum (µm) values.	Maximum (µm) and Minimum (µm) (memb_est_z_upper_bound, memb_est_z_lower_bound as internal variables) are a range of coordinates in the z-axis where the bottom boundary of the membrane can be found. If these two values are set such that that no objects in any sheet (chip ID) in the Excel spreadsheet fall into that range, membrane estimation cannot proceed.
If chip IDs are not equal across all Excel spreadsheets, display an error message and stop the analysis.	Chip IDs are inconsistent. They must be present in all input Excel spreadsheets.
Display a warning message about the output directory and ask the user if they want to proceed.	If the user selects an output directory that already exists and is not empty, existing files from previous runs may be overwritten.
Display a warning message and ask the user to save the notebook manually.	The plugin nbconvert does not automatically save the notebook before exporting, so Jupyter Notebook must be manually saved by the user every time it is run. Otherwise, the report may not be up to date.
Error "FileNotFoundError: [Errno 2] No such file or directory"	Microsoft Windows has a MAX_PATH limit of ~256 characters. When the length of any file path exceeds this limit, the error "FileNotFoundError: [Errno 2] No such file or

	directory" will occur in the Jupyter Notebook, which halts the analysis. To mitigate this, check the lengths of the paths of input and output directories. Reduce the layers of directories or shorten file names as much as possible.

Hierarchical Structure of the Output Directory

Hierarchy

The output is a folder of analyzed features in the forms of the statistics tables (csv files) and plots (png files). The output files are grouped by the corresponding channels and their categories. For example, there are currently two sub-folders in the feature_analysis_output folder: cell_tracker_CAR-T and caspase, which are outputs of analyzed data from such channels. Each sub-folder follows almost the same hierarchical structure.

- output(folder)
    - **feature_analysis_output (folder)**
        - cell_tracker_CAR-T (folder)
            - 2D_object_centroid_map (folder)
            - all_class_features_all_fov (folder)
            - bar_plots (folder)
                - features_between_chip (folder)
                - features_between_classes (folder)
                - features_between_fov (folder)
                - stacked_features_between_chips (folder)
                - stacked_features_between_fov (folder)
            - chip_features.csv (table of stats)
            - fov_features.csv (table of stats)
        - caspase (folder)
            - (Same structure as above)
        - feature_analysis.html (report)
        - logs_YYYY-MM-DD_HH-MM-SS.txt (logs)
    - co-localization_output
        - (Described in the next section)
    - max_projection_output
        - (Described in the next section)

Output Data

Overview

The outputs in the notebook are separated into 5 sections:

1. Membrane estimation
2. Classification
3. Feature extraction
4. Feature analysis
5. Plotting

Membrane Estimation

Cell objects in each FOV are separated into four classes according to their centroid coordinates relative to the per-chip membrane estimates. Scatter plots of object centroids and estimated membrane are generated to provide visualization to the user on a per-chip basis, where data points from all FOV are aggregated into a single scatter plot. The top and bottom ends of the estimated membrane are plotted using black dashed lines. The object centroids are plotted in different colors to differentiate their classes:

Color	Class
Red	On top and above membrane
Black	Within membrane
Green	At the bottom of the membrane
Blue	Below membrane

Classification

In classification, the same scatter plots of object centroids and estimated membrane are also generated. The only difference is that these scatter plots are generated on a per-FOV basis, only showing data points in each FOV. This process is repeated for all FOV.

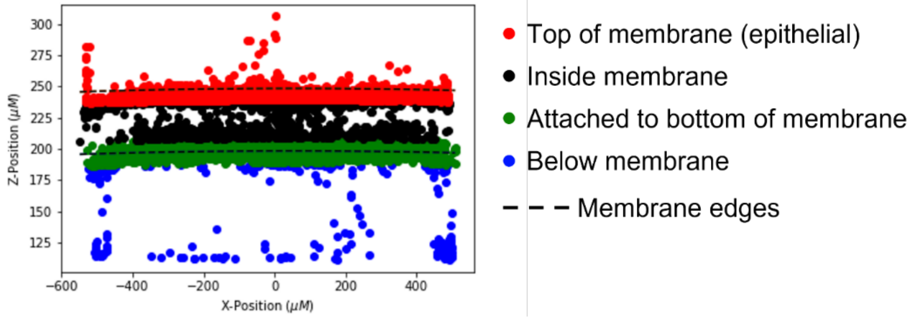


Figure 21. A scatterplot in the xz-plane that displays the locations of object centroids, their recruitment characterizations, and membrane estimates.

Continued on next page

Output Data, Continued

Feature Extraction

After cell objects are classified, the next step is feature extraction, which identifies key object features for each class. The purpose is quantification of stains by identifying features, including **number of objects (counts), total volume of objects, mean intensity of objects, and sum intensity of objects**. The counts of migrated objects and the percentage of migration are two other features calculated across all classes in each FOV. These pieces of information are extracted from the Excel spreadsheets provided by the ImageJ script in the 3D object counting step. This process, on a per-class basis, is repeated for the overall FOV and for each class in each FOV, for all FOV. The four primary features are:

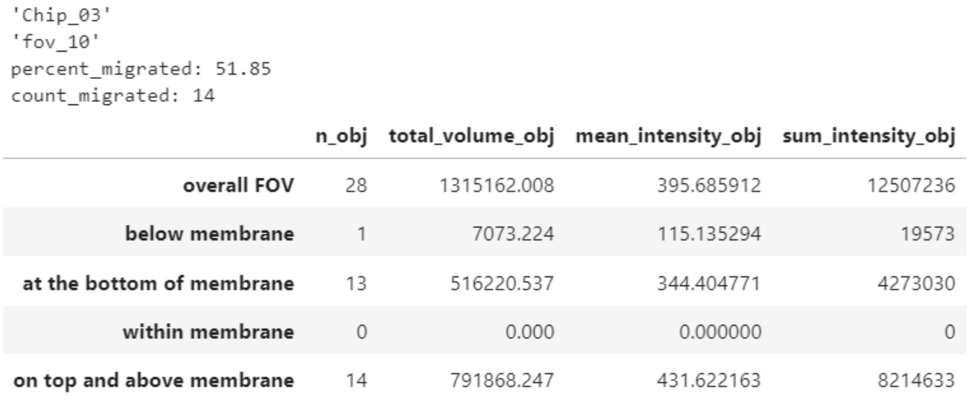
- **The number of objects** is calculated from the number of rows in the Excel spreadsheet of object statistics from the ImageJ script.
- **The total volume of objects** is calculated as the sum of volume for all objects in that class. It is calculated from the “Volume (micron^3)” column in the Excel spreadsheet of object statistics from the ImageJ script.
- **The sum intensity of objects** is the sum of all pixel intensities of all objects, calculated from the “IntDen” column in the Excel spreadsheet of object statistics from the ImageJ script.
- **The mean intensity objects** is (sum intensity) / (sum of all object volume) in that class. The sum of all object volume is taken from the “Nb of obj. voxels” column in the Excel spreadsheet of object statistics from the ImageJ script.
- **Note:** The volume here is different from the total volume of objects above. However, the “Volume (micron^3)” values are just “Nb of obj. voxels” multiplied by a scalar.
- **Additional feature:** The percentage of migration is calculated as the counts (above membrane + within membrane) / (above membrane + within membrane + at the bottom of membrane) * 100. The denominator excludes counts below membrane—that is, counts in the bottom channel. This value is calculated on a per-FOV basis.

Continued on next page

Output Data, Continued

Percentage of Migration

$$\frac{\left(N_{\text{top of membrane}} + N_{\text{inside membrane}}\right)}{\left(N_{\text{top of membrane}} + N_{\text{inside membrane}} + N_{\text{bottom of membrane}}\right)} \times 100$$



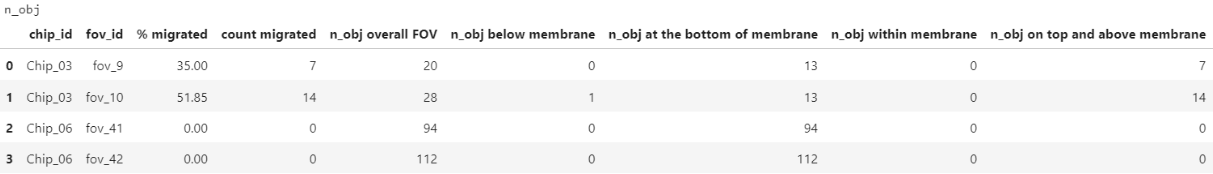
```
'Chip_03'
'fov_10'
percent_migrated: 51.85
count_migrated: 14
```

	n_obj	total_volume_obj	mean_intensity_obj	sum_intensity_obj
overall FOV	28	1315162.008	395.685912	12507236
below membrane	1	7073.224	115.135294	19573
at the bottom of membrane	13	516220.537	344.404771	4273030
within membrane	0	0.000	0.000000	0
on top and above membrane	14	791868.247	431.622163	8214633

Feature Analysis

In the previous feature extraction step, the object features are calculated on a per-class basis. In this step, the features are further analyzed on per-FOV and per-chip bases. Features calculated in this step include:

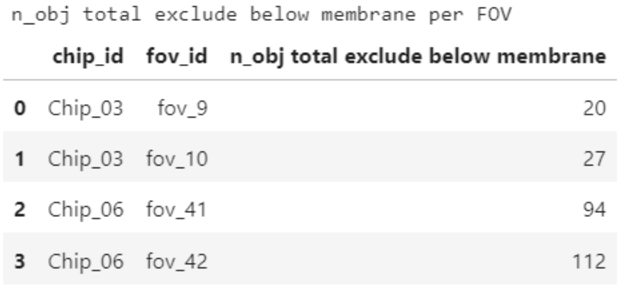
- Summary of features per FOV
  - Comparing features in each class across FOV.
  - There are four tables in total, one for each feature.



n_obj

	chip_id	fov_id	% migrated	count migrated	n_obj overall FOV	n_obj below membrane	n_obj at the bottom of membrane	n_obj within membrane	n_obj on top and above membrane
0	Chip_03	fov_9	35.00	7	20	0	13	0	7
1	Chip_03	fov_10	51.85	14	28	1	13	0	14
2	Chip_06	fov_41	0.00	0	94	0	94	0	0
3	Chip_06	fov_42	0.00	0	112	0	112	0	0

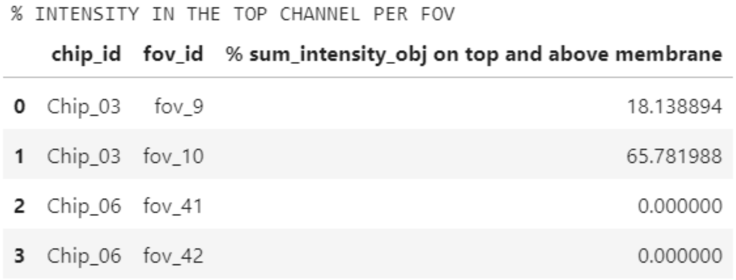
- Number of objects in total exclude below membrane per FOV
  - Sum of number of objects across all classes in a FOV, minus number of objects below the membrane—that is, counts in the bottom channel. This is calculated on a per-FOV basis.



```
n_obj total exclude below membrane per FOV
```

	chip_id	fov_id	n_obj total exclude below membrane
0	Chip_03	fov_9	20
1	Chip_03	fov_10	27
2	Chip_06	fov_41	94
3	Chip_06	fov_42	112

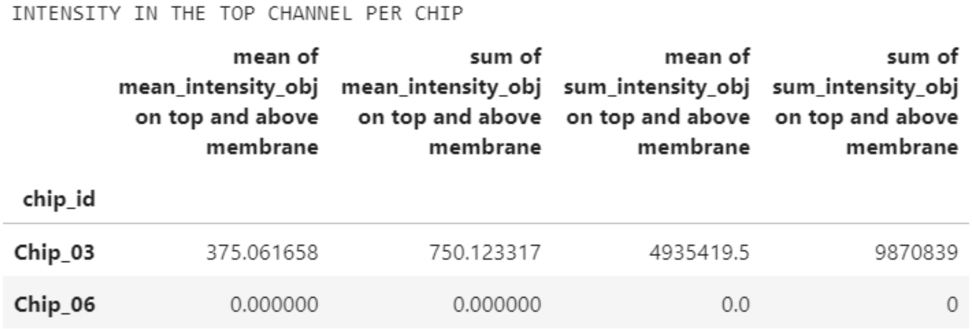
- % intensity in the top channel per FOV
  - The percentage is the sum intensity in the top channel divided by the total sum intensity except the bottom channel
  - (On top and above membrane) / (at the bottom + inside membrane + on top and above membrane) * 100
  - The intensity below the membrane (aka in the bottom channel) is excluded.
  - This is calculated on a per-FOV basis



```
% INTENSITY IN THE TOP CHANNEL PER FOV
```

	chip_id	fov_id	% sum_intensity_obj on top and above membrane
0	Chip_03	fov_9	18.138894
1	Chip_03	fov_10	65.781988
2	Chip_06	fov_41	0.000000
3	Chip_06	fov_42	0.000000

- Intensity in the top channel per chip
  - mean of mean_intensity_obj on top and above membrane across FOV
  - sum of mean_intensity_obj on top and above membrane across FOV
  - mean of sum_intensity_obj on top and above membrane across FOV
  - sum of sum_intensity_obj on top and above membrane across FOV
  - They are calculated on a per-chip basis.

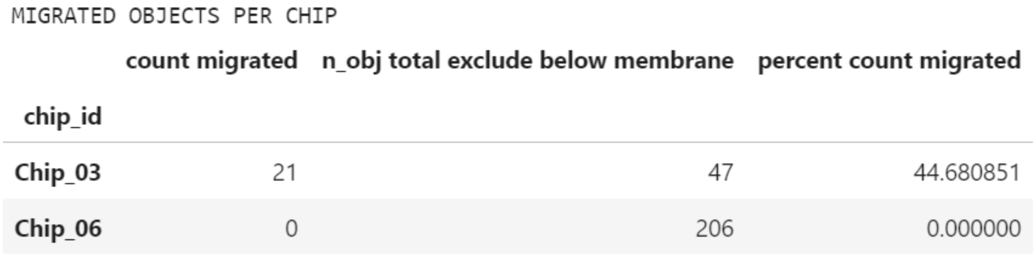


```
INTENSITY IN THE TOP CHANNEL PER CHIP
```

	mean of mean_intensity_obj on top and above membrane	sum of mean_intensity_obj on top and above membrane	mean of sum_intensity_obj on top and above membrane	sum of sum_intensity_obj on top and above membrane
chip_id				
Chip_03	375.061658	750.123317	4935419.5	9870839
Chip_06	0.000000	0.000000	0.0	0

- Migrated objects per chip

- Count of migrated objects per chip
  - The percentage of migration is calculated as the counts (on top and above membrane + within membrane) / (on top and above membrane + within membrane + at the bottom of membrane). The denominator excludes counts below membrane—that is, counts in the bottom channel. This is calculated on a per-chip basis.
- Number of objects in total exclude below membrane per chip
  - Take the number of objects in total excluding below membrane per FOV calculation above and sum those values across all FOV for each chip. This is calculated on a per-chip basis.
- % count of migrated objects per chip
  - Divide the migrated count by the number of objects in total excluding below membrane to get the percentage
  - (Within membrane + on top and above membrane) / (at the bottom + within membrane + on top and above membrane) * 100. The counts below the membrane (that is, the bottom channel) are excluded.



MIGRATED OBJECTS PER CHIP

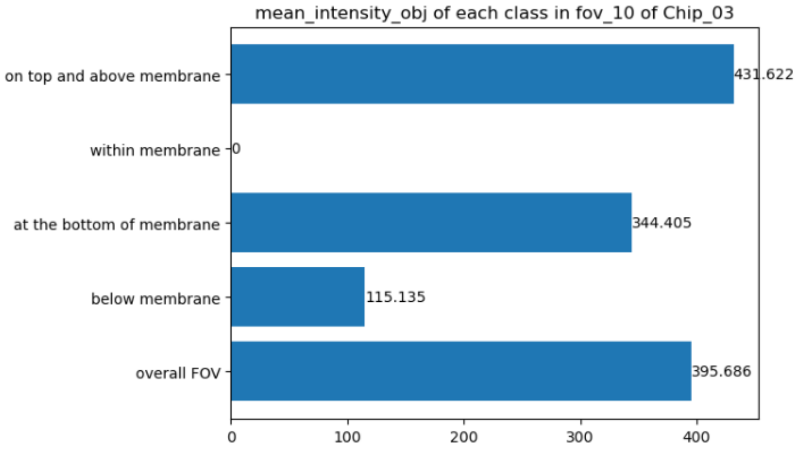
	count migrated	n_obj total exclude below membrane	percent count migrated
chip_id			
Chip_03	21	47	44.680851
Chip_06	0	206	0.000000

Note: The per-chip outputs are generally more useful than the per-FOV outputs for a more holistic view of migration and recruitment. However, both types of outputs are available for the user.

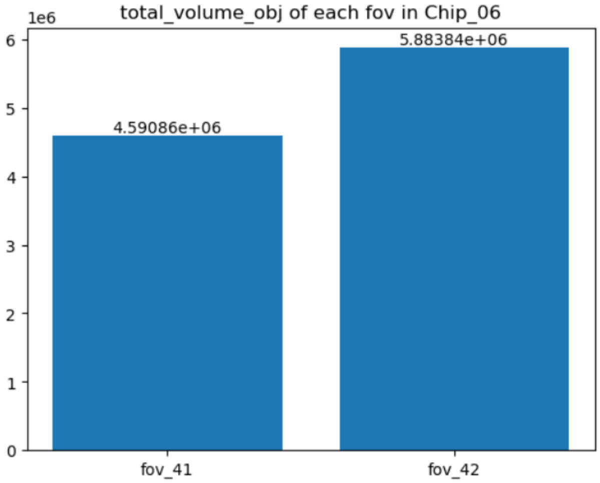
Plotting

After all features are analyzed, the plotting step enables visualization of the results using bar plots and stacked bar plots, which allow the user to inspect the data from different perspectives. There are five sets of plots:

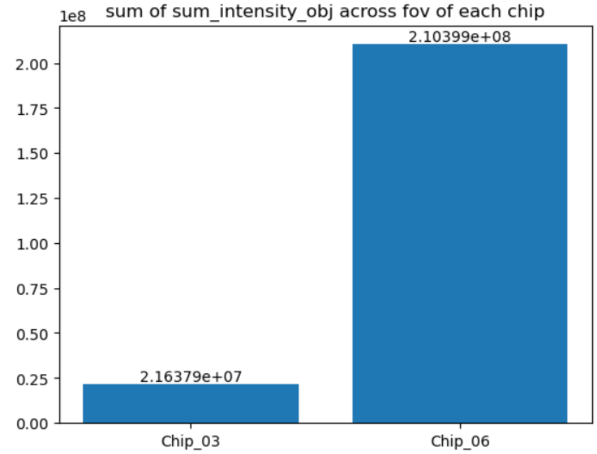
- Bar plots comparing features between classes
  - Bar plots for comparing features between classes, for all FOV of all chips
  - This set contains (4 features * number of FOV * number of chips) plots



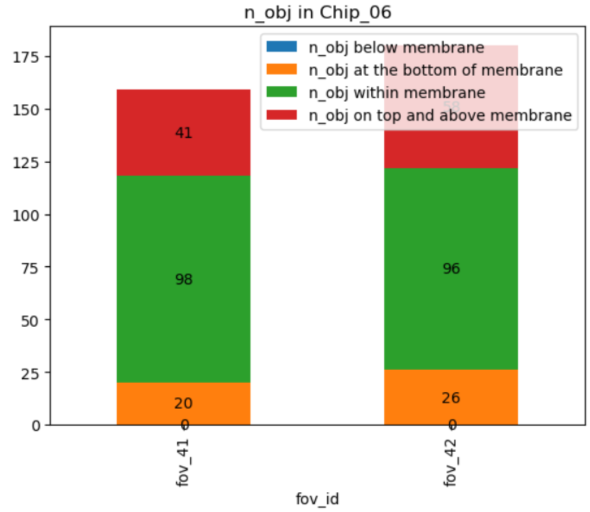
- Bar plots comparing features between FOV
  - Bar plots for comparing the specified feature of the overall FOV between FOV, for all chips.
  - This set contains (4 features * number of chips) plots



- Bar plots comparing features between chips
  - Bar plots for comparing the mean or sum of specified feature across all FOV in a chip, between chips.
  - This set contains 8 plots (2 methods, mean or sum * 4 features).

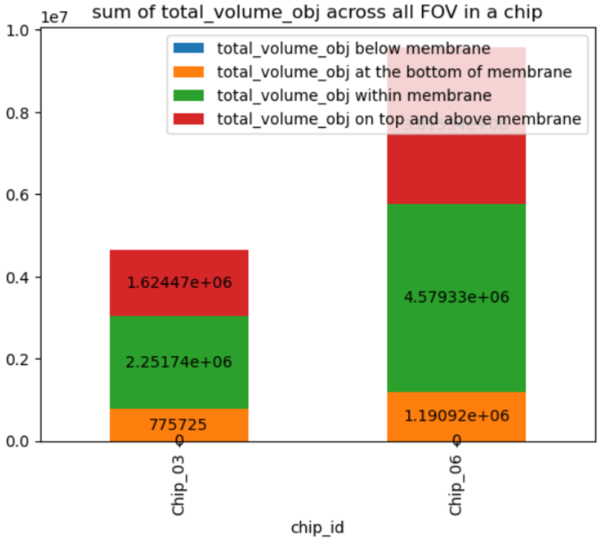


- Stacked bar plots comparing features between FOV
  - A more compact way of visualization.
  - Stacked bar plot for comparing the specified feature between classes and between FOV, for all chips. Features of all classes are stacked in the same FOV.
  - This set contains (3 features * number of chips) plots
  - Mean intensity is excluded here because mean is not stackable. The sum of the means of a subset is not the sum or the mean of the entire set.



- Stacked bar plots comparing features between chips
  - A more compact way of visualization.
  - Stacked bar plots for comparing the sum of specified feature across FOV for all classes, between chips. Features of all classes are stacked in the same chip. The values of the classes are summed across all FOV in the chip.
  - This set contains 3 plots (3 features).

- Mean intensity is excluded here because mean is not stackable. The sum of mean of a subset is not the sum or the mean of the entire set.



Note: *The per-chip outputs are generally more useful than the per-FOV outputs for a more holistic view of migration and recruitment. However, both types of outputs are available for the user.*

Co-Localization for CAR T Killing

Overview

Summary

The aim of co-localization is to capture the event of cell death by correlating the cell presence and cell death spatially. The cell tracker for CAR T stain detects the presence of CAR T cells, while the caspase stain detects cell death. A cell tracker for CAR T object is considered dead when it is co-localized with a caspase object and live when otherwise.

By identifying overlaps of objects from a cell tracker stain and a cell death stain, it is possible to infer potential cell death events taking place at the time of observation, and therefore the cell states (live or dead). Co-localized objects, in the context of caspase and cell tracker, represent dead CAR T. Objects that are not co-localized, in the context of caspase and cell tracker, represent live CAR T.

Note: Co-localization only indicates correlation (there is cell death and CAR T at the same location) and not causation (cell death is a direct result of the presence of CAR T). Therefore, the user needs to be careful in drawing conclusions from the results.

Contents

Algorithm

Description

While there are many approaches to co-localization, the fractional overlap method is the most appropriate for this analysis. The core idea is that, for a pair of objects—one from a cell tracker stain and one from cell death stain—it is possible to measure the fraction of overlap by calculating the ratio of volume of the intersection to volume of the cell tracker object (See Figure 22).

If the ratio is above a pre-defined threshold, then the two objects are considered co-localized. Alternatively, the volume of union can be used as the denominator. The volume of the cell tracker object is used because the presence of cells (CAR T in this case) is the reference, while the cell death stain is just an indicator of the cell state (live or dead).

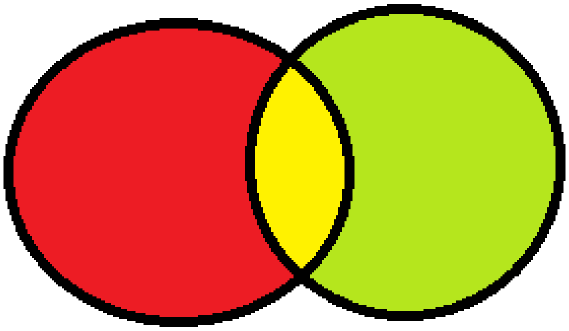
This process is repeated for all pairs of objects from a cell tracker stain (CAR T) and a cell death stain (caspase).

Steps

Algorithm for calculating the ratio of intersection volume over cell tracker object volume. For each object in the cell tracker:

Step	Action
1	Find the intersection
2	Calculate the intersection volume
3	Ratio of overlap = intersection volume / object volume

Figure 22



A diagram illustrating fractional overlap in 2D. The red object is from cell tracker for CAR T, while the green is from caspase stain. The yellow area is the intersection. The ratio of overlap is calculated as (area of yellow) / (area of red).

Input Data

Input Images

The input images to co-localization are the output object maps from the ImageJ script, a natural continuation from that step. The input object maps to the co-localization program have a very specific format. Please format the inputs according to the following descriptions if other programs are used for 3D object counting instead of the “ImageJ script for 3D object counting,” which is described above.

Raw 3D Images

The raw 3D images from each channel (cell tracker and cell death stains) are first segmented to separate objects from the background. Then, the pixel values in each object need to be labeled as the object index, instead of binary values, so the script can track which objects are co-localized. The background pixels have values of 0. All individual objects are labeled with positive integers. A pair of object maps in the form of 3D images (one from cell tracker and one from caspase) are input into the co-localization program.

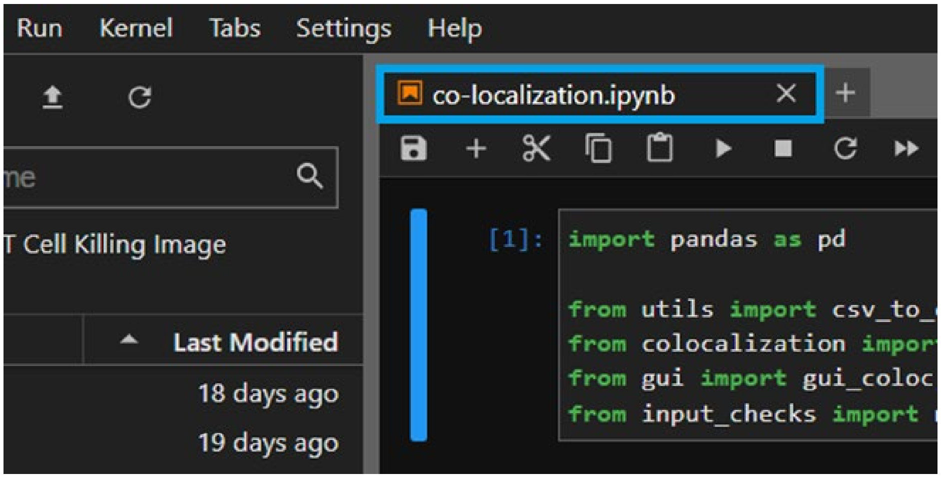
Object Maps

Object maps created by any programs that count and segment 3D objects can be used so long as the object maps are formatted according to the descriptions above. 2D or 3D object maps can be used as inputs, as long as the dimensions match between the pair of input images.

Protocol

Steps

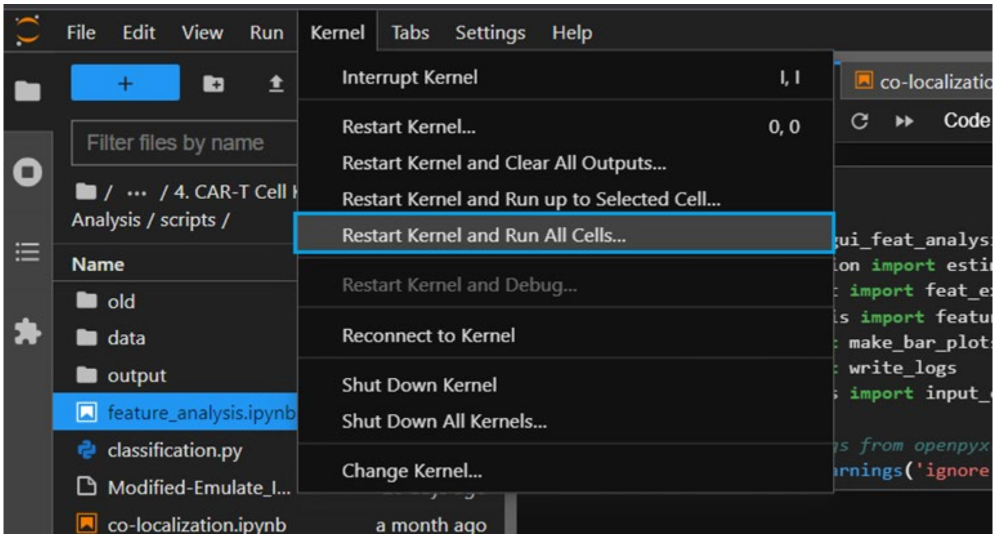
Step	Action
1	In JupyterLab, select the co-localization.ipynb tab.



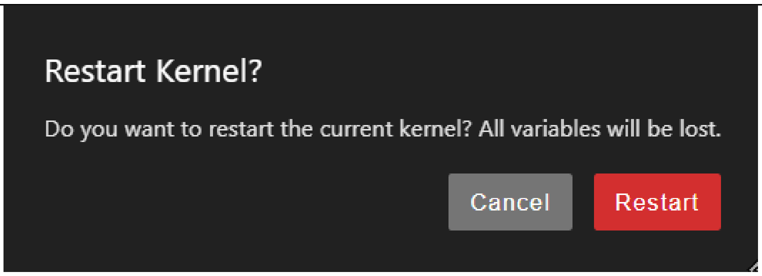
```
import pandas as pd

from utils import csv_to_
from colocalization impor
from gui import gui_coloc
from input_checks import
```

Step	Action
2	At the toolbar, select “Kernel” → “Restart Kernel and Run All Cells…”



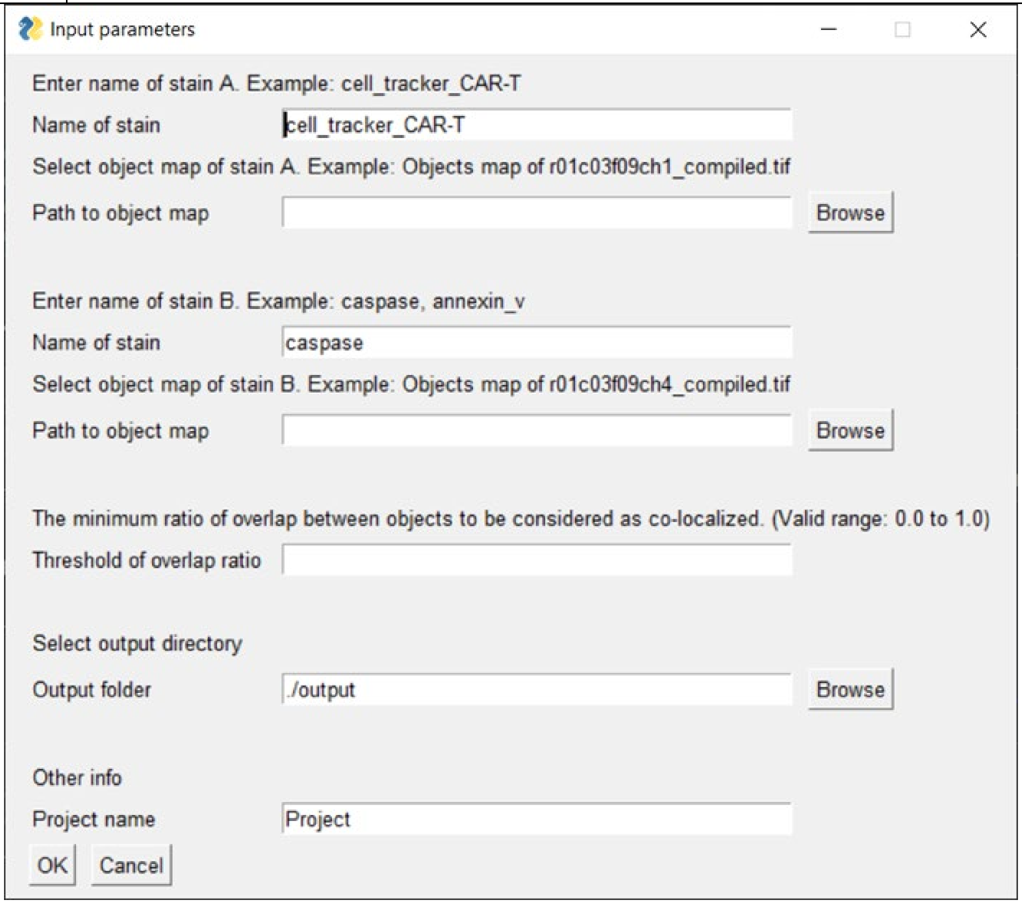
Step	Action
3	If a “Restart Kernel?” dialog pops up, select “Restart.”



Restart Kernel?

Do you want to restart the current kernel? All variables will be lost.

4 A GUI window will pop up asking the user to input parameters.



Enter name of stain A. Example: cell_tracker_CAR-T

Name of stain: cell_tracker_CAR-T

Select object map of stain A. Example: Objects map of r01c03f09ch1_compiled.tif

Path to object map:

Enter name of stain B. Example: caspase, annexin_v

Name of stain: caspase

Select object map of stain B. Example: Objects map of r01c03f09ch4_compiled.tif

Path to object map:

The minimum ratio of overlap between objects to be considered as co-localized. (Valid range: 0.0 to 1.0)

Threshold of overlap ratio:

Select output directory

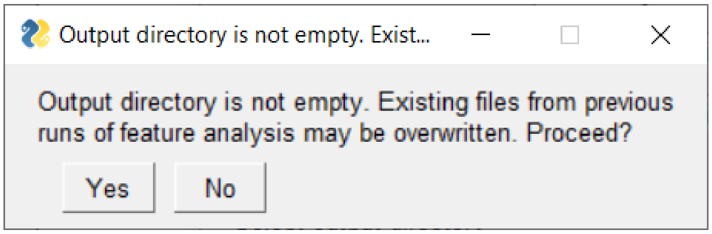
Output folder: ./output

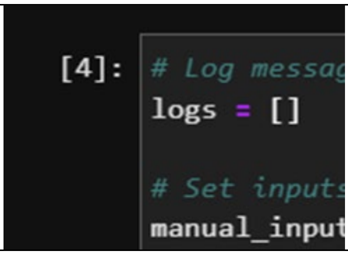
Other info

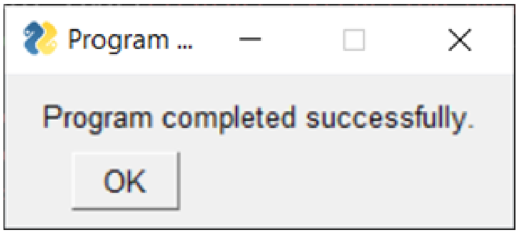
Project name: Project

5 Enter the following parameter values:

Step	Action
1	Enter the name of the stain/channel A (example: cell tracker for CAR T). This is the reference/baseline channel used for co-localization.
2	Click “Browse” to locate and select the object map of stain A, created by the ImageJ Script for 3D Object Counting in the previous section.
3	Enter the name of the stain/channel B (example: caspase).
4	Click “Browse” to locate and select the object map of stain B, created by the ImageJ Script for 3D Object Counting in the previous section.

		Note: Swapping stain A and B may create slightly different results. Be sure to use stain A as the reference stain.
	5	Enter the minimum ratio of overlap between objects to be considered as co-localized. This is a threshold that separates true positives from false positives. **Note:** The ratio of overlap = volume intersection / volume of object from stain A.
	6	Click “Browse” to locate and select the output directory (create one first if one does not already exist). **Note:** A sub-folder “co-localization_output” will be automatically created under this selected folder to store all outputs from this analysis. Another sub-folder will be created under “co-localization_output” using the common Chip ID and FOV ID of the input images (for example: a folder called “r01c03f09” will be created for input images “Objects map of r01c03f09ch1_compiled.tif” and “Objects map of r01c03f09ch4_compiled.tif”). Be sure to use different output directories to store the results of different experiments/projects.
	7	Enter the desired project name. The default value can be used.
	8	Click “OK” when done.
6		If the output directory is not empty, existing files may be overwritten. Click “Yes” to start the analysis and overwrite existing files or “No” to return to the previous step to select/create a different output folder.
		 Output directory is not empty. Exist... Output directory is not empty. Existing files from previous runs of feature analysis may be overwritten. Proceed? Yes No
7		While the analysis is running, the results will start populating in the notebook. Keep scrolling to view and follow the outputs while they are being generated. **Note:** When a code block is running, there is an asterisk on the upper left. When the code block is done, a number (execution order) will replace the asterisk.

`[*]: if run_analy`
`popup_me`
`# Export`  `[4]: # Log messag`
`logs = []`
`# Set inputs`
`manual_input`	

8	Finally, if the program is completed successfully, the following message will appear. If this message does not show up, check the outputs in the notebook to see if the program is still running or has encountered an error.
 Program ... Program completed successfully. 	

Error Message

Messages and Causes

Messages	Possible Causes
Display a warning message “Please fill out all fields.”	Some inputs in GUI are not filled out.
Display a dialog “Do you really want to exit?”	User accidentally clicks “x” or “Cancel.”
Display an info message “Analysis cancelled.”	User decides to cancel.
Display a warning message “FOV is not present or doesn't match in file names. Using file name as FOV ID instead. Program will continue.”	The file name of object maps created by ImageJ script has a specific format. Sample format: "Objects map of r01c03f09ch1_compiled.tif", where “r01c03f09” is the FOV ID, which must be preceded by "Objects map of " and ended in "ch". The channel index is “ch1”. If the user uses object maps with the file name formatted differently than this or two object maps with different FOV ID, the FOV ID cannot be determined.
If the dimensions of both images do not match, display an error message and stop the analysis.	The input object maps have different dimensions.
Error: “FileNotFoundError: [Errno 2] No such file or directory”	Microsoft Windows has a MAX_PATH limit of ~256 characters. When the length of any file path exceeds this limit, the error “FileNotFoundError: [Errno 2] No such file or directory” will occur in the Jupyter Notebook which halts the analysis. To mitigate this, check the lengths of the paths of input and output directories. Reduce the layers of directories or shorten file names as much as possible.

Hierarchical Structure of the Output Directory

Overview

The output is a folder of object maps (in the form of 3D image stacks, TIFF format) and co-localization statistics (in the form of tables, csv format). The folder is named "co-localization_output." The outputs are organized into sub-folders by the chip ID and FOV ID (common ID between the two inputs images, e.g., r01c03f09 is the common ID between Objects map of r01c03f09ch1_compiled.tif and Objects map of r01c03f09ch4_compiled.tif) and the pair of co-localized stains (e.g., cell_tracker_CAR-T_caspase). For example, there is currently one sub-folder in the co-localization_output folder: r01c03f09, under which is another subfolder: cell_tracker_CAR-T_caspase, which are the names of the two stains. Each sub-folder follows the same hierarchical structure.

Hierarchy

- output (folder)
    - feature_analysis_output (folder)
        - (Described in the previous section)
    - **co-localization_output**
        - r01c03f09 (folder) (1 or more)
            - cell_tracker_CAR-T_caspase (folder) (1 or more)
                - cell_tracker_CAR-T_caspase_coloc_stats.csv (table of stats)
                - cell_tracker_CAR-T_caspase_stain_coloc_map.tif (3D object map)
                - cell_tracker_CAR-T_cell_state_obj_map.tif (3D object map)
                - cell_tracker_CAR-T_coloc_obj_map.tif (3D object map)
        - coloc_obj_counts.csv (table of stats)
        - logs_YYYY-MM-DD_HH-MM-SS.txt (logs)
    - max_projection_output
        - (Described in the next section)

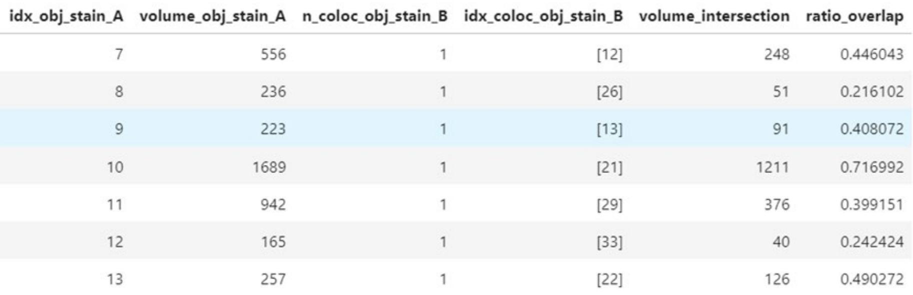
Output Data

Key Outputs

- Table of co-localization statistics
- Stain co-localization map
- Co-localized object map
- Cell state object map

Table of Localization Statistics

After the co-localization process is completed, a table of statistics is generated, which includes information such as indices of co-localized cell tracker objects (CAR T) and cell death stain objects (caspase), volume of intersection, and number of co-localized objects. This table of statistics, especially the indices of co-localized cell tracker objects (CAR T), is then used for creating the object maps.



idx_obj_stain_A	volume_obj_stain_A	n_coloc_obj_stain_B	idx_coloc_obj_stain_B	volume_intersection	ratio_overlap
7	556	1	[12]	248	0.446043
8	236	1	[26]	51	0.216102
9	223	1	[13]	91	0.408072
10	1689	1	[21]	1211	0.716992
11	942	1	[29]	376	0.399151
12	165	1	[33]	40	0.242424
13	257	1	[22]	126	0.490272

Table 6. Co-localization statistics. A type of output from this process.

Stain co-localization map

Besides the table of statistics, objects are also generated. The stain co-localization map (See Figure 23) is a 3D image stack that displays co-localized objects from cell tracker for CAR T, co-localized objects from caspase stain, and their intersections, all labeled with different colors. This map provides an intuitive way of visualizing the co-localization results.

Co-localized object map

The co-localized object map for CAR T (See Figure 24) is another 3D image stack that only displays co-localized objects from the cell tracker for CAR T stain. This map enables the user to visualize the “co-localized (dead) CAR T” only.

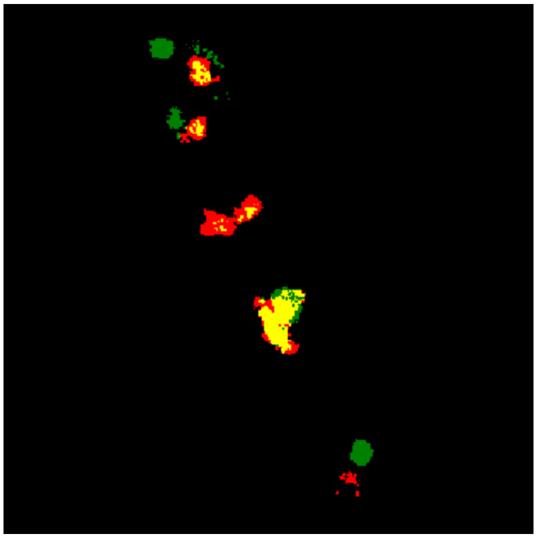
Cell state object map

The cell state object map (See Figure 25) is a 3D image stack that displays not only the co-localized (dead) CAR T but also the non-co-localized (live) ones. This map enables the user to view the cell states (co-localized and non-co-localized) of the CAR T cells.

Continued on next page

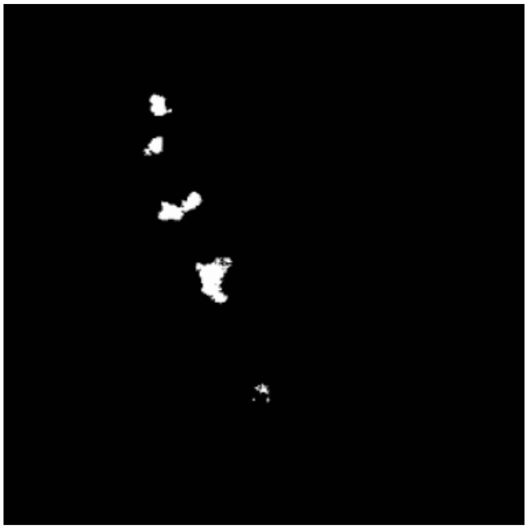
Output Data, Continued

Figure 23



A section of a stain co-localization map between cell tracker for CAR T and caspase, where cell tracker for CAR T is red, "caspase" is green, and the intersection is yellow.

Figure 24

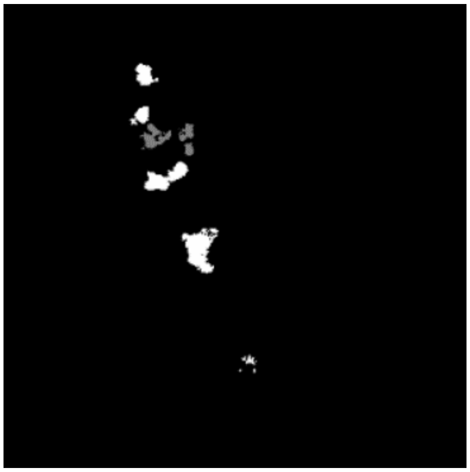


A section of a co-localized object map for CAR T, where the only objects displayed are co-localized (dead) CAR T objects, which are labeled as white (cell tracker for CAR T+ caspase = dead CAR T).

Continued on next page

Output Data, Continued

Figure 25



A section of a cell state object map for CAR T, where co-localized (dead) cells are labeled as white and live cells are labeled as gray.

Maximum Intensity Projection

Overview

Summary

A user may be interested in the sum intensity in the 2D representation of a 3D image stack of each FOV. Maximum intensity projection flattens a 3D image stack into a 2D image by taking the maximum value of intensity along the z-axis for each pixel in the x- and y-coordinates. The sum intensity is calculated by taking the sum of all pixel values in that 2D image.

Contents

Input Data

Summary

The input is a raw 3D image stack which has not been subject to segmentation or 3D object counting in ImageJ. This operation is described independently in this section and separately from the feature analysis section because the inputs are different. The inputs to feature analysis are Excel spreadsheets of object statistics, whereas maximum intensity projection requires raw 3D image stacks as inputs.

Note: Inputs

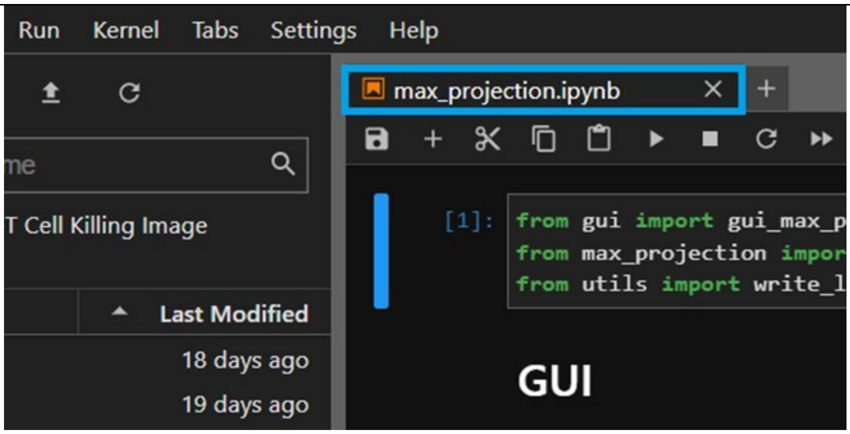
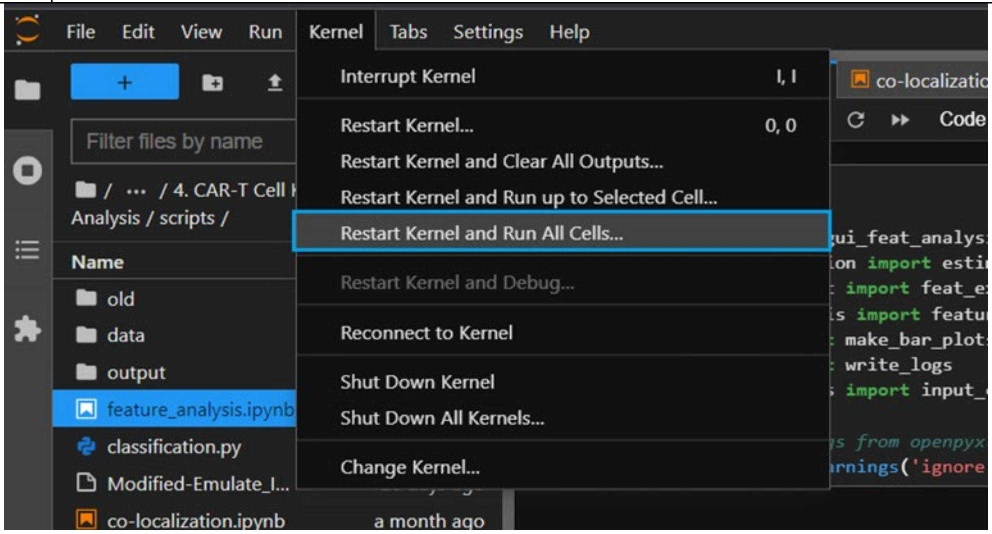
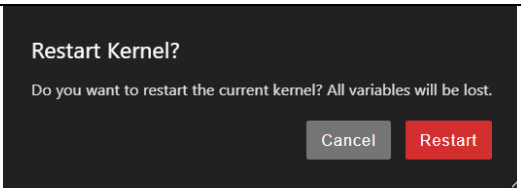
Only 3D images can be used as inputs.

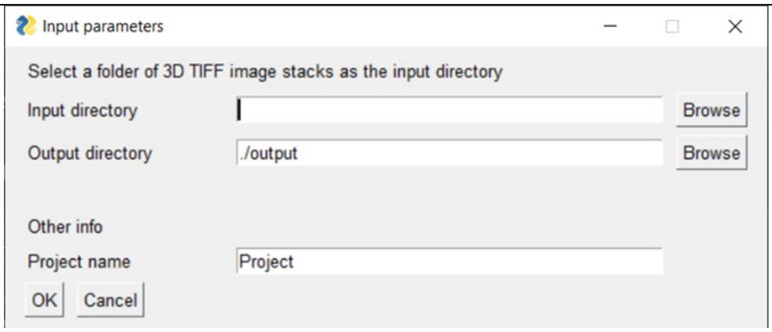
Note: 16-bit images

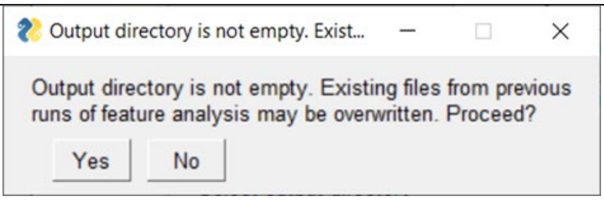
For a 16-bit image (with intensity values up to 2^{16}-1), the largest possible size of an input image is 562,950 GB.

Protocol

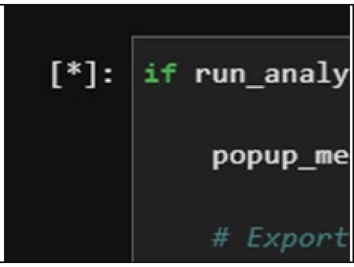
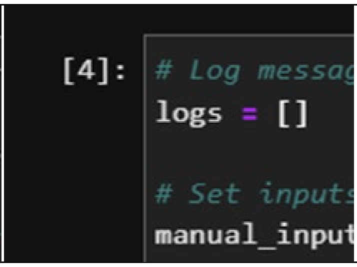
Steps

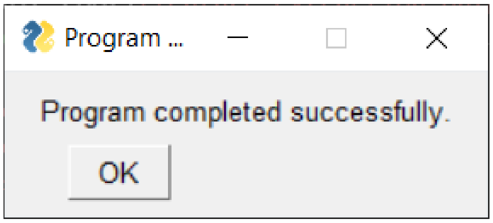
Step	Action
1	In JupyterLab, select the max_projection.ipynb tab.
	 Run Kernel Tabs Settings Help max_projection.ipynb T Cell Killing Image Last Modified 18 days ago 19 days ago [1]: GUI from gui import gui_max_p from max_projection impor from utils import write_l
2	At the toolbar, select “Kernel” → “Restart Kernel and Run All Cells…”
	 File Edit View Run Kernel Tabs Settings Help Interrupt Kernel I, I Restart Kernel... 0, 0 Restart Kernel and Clear All Outputs... Restart Kernel and Run up to Selected Cell... Restart Kernel and Run All Cells... Restart Kernel and Debug... Reconnect to Kernel Shut Down Kernel Shut Down All Kernels... Change Kernel... Filter files by name / ... / 4. CAR-T Cell I Analysis / scripts / Name old data output feature_analysis.ipynb classification.py Modified-Emulate_I... co-localization.ipynb a month ago co-localizatic Code ui_feat_analys: ion import esti : import feat_e: ls import featu : make_bar_plot: : write_logs ; import input_ rs from openpyx arnings('ignore
3	If a “Restart Kernel?” dialog pops up, select “Restart.”
	 Restart Kernel? Do you want to restart the current kernel? All variables will be lost.
4	A GUI window will appear asking the user to input parameters.

<table>
<tr><td colspan="2">

Select a folder of 3D TIFF image stacks as the input directory
Input directory
Output directory ./output
Other info
Project name Project</td></tr>
<tr><td>5</td><td>Enter the following parameter values:
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Click “Browse” to locate and select the input directory that contains the raw image z-stacks (not object maps) to be analyzed. These are the same inputs used for the ImageJ script for 3D object counting.</td></tr>
<tr><td>2</td><td>Click “Browse” to locate and select the output directory (create one first if one does not already exist).
Note: A sub-folder “max_projection_output” will be automatically created under this selected folder to store all outputs from this analysis. Be sure to use different output directories to store the results of different experiments/projects.</td></tr>
<tr><td>3</td><td>Enter the desired project name. The default value can be used.</td></tr>
<tr><td>4</td><td>Click “OK” when done.</td></tr>
</table></td></tr>
<tr><td>6</td><td>If the output directory is not empty, existing files may be overwritten. Click “Yes” to start the analysis and overwrite existing files, or “No” to return to the previous step to select/create a different output folder.</td></tr>
<tr><td colspan="2">

Output directory is not empty. Existing files from previous runs of feature analysis may be overwritten. Proceed?</td></tr>
<tr><td>7</td><td>While the analysis is running, the results will start populating in the notebook. Keep scrolling to view and follow the outputs while they are being generated.
Note: When a code block is running, there is an asterisk on the upper left. When the code block is done, a number (execution order) will replace the asterisk.</td></tr>
</table>

 `[*]: if run_analy`
`popup_me`
`# Export`  `[4]: # Log messag`
`logs = []`
`# Set inputs`
`manual_input`	

8	Finally, if the program is completed successfully, the following message will appear. If this message does not show up, check the outputs in the notebook to see if the program is still running or has encountered an error.
 Program completed successfully.	

Error Messages

Overview

Messages	Possible Causes
Display a warning message “Please fill out all fields.”	Some inputs in GUI are not filled out.
Display a dialog “Do you really want to exit?”	User accidentally clicks “x” or “Cancel.”
Display an info message “Analysis canceled.”	User decides to cancel.
Display a warning message and ask the user if he/she wants to proceed.	If the user selects an output directory that already exists and is not empty, existing files from previous runs may be overwritten.
Display a warning message for the user to use 3D images. The image will be skipped, and the analysis will continue.	If the input image is not a 3D stack, e.g., 2D, 4D, then maximum intensity projection cannot be applied.
Display a warning message about empty input directory.	If directory is empty, contains no TIFF files, or does not exist.
Error “FileNotFoundError: [Errno 2] No such file or directory”	Microsoft Windows has a MAX_PATH limit of ~256 characters. When the length of any file path exceeds this limit, the error “FileNotFoundError: [Errno 2] No such file or directory” will occur in the Jupyter Notebook, which halts the analysis. To mitigate this, check the lengths of the paths of input and output directories. Reduce the layers of directories or shorten file names as much as possible.

Hierarchical Structure of the Output Directory

Overview

The output is a folder of projected images (in the form of 2D image, TIFF format) and a table of sum intensities (csv format). The folder is named “max_projection_output.”

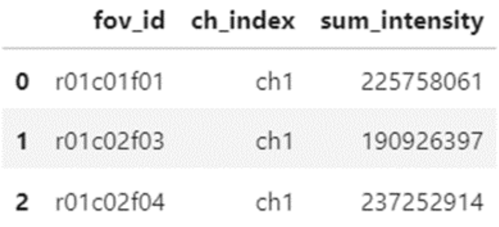
- Output (folder)
  - feature_analysis_output (folder)
    - (Described in previous section)
  - co-localization_output
    - (Described in previous section)
  - **max_projection_output**
    - max_projected_r01c01f01ch1.tiff (2D image) (1 or more)
    - sum_intensity_max_projection.csv (table of sum intensities)
    - logs_YYYY-MM-DD_HH-MM-SS.txt (logs)

Output Data

Summary

- 2D images created through maximum intensity projection
- A table of sum intensities

Table 7



	fov_id	ch_index	sum_intensity
0	r01c01f01	ch1	225758061
1	r01c02f03	ch1	190926397
2	r01c02f04	ch1	237252914

Sum intensities.

References

Overview

References

The **ImageJ Script for 3D Object Counting** utilizes the ImageJ plugin *Cordelières, F (2006) 3D Objects Counter (Version 2.0.1) [Source code]. https://imagej.net/plugins/3d-objects-counter#license*

The original plugin was published here: S. Bolte & F. P. Cordelières, A guided tour into subcellular colocalization analysis in light microscopy, Journal of Microscopy, Volume 224, Issue 3: 213-232

Link to the 3D Object Counter Plugin Manual: https://imagejdocu.list.lu/lib/exe/fetch.php?media=plugin:analysis:3d_object_counter:3d-oc.pdf