Immunostained Image Analysis Documentation

Release 1.3

Created August 10, 2022

Last updated January 15, 2024

Boise State University

Mechanical Adaptations Laboratory

Table of Contents

General Information and Overview

1.1 License

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

1.2 System Requirements

This program and the software necessary to run it have been tested on the following configuration:

Operating System: Windows 10 64-bit

A variety of separately installed programs and code packages are also necessary; These will be discussed in the following installation procedure

Installation

2.1 Required Software

Install the following programs following the provided instructions, and preferably in the given order. Note that all given version numbers represent what was used at the time and is known to work – other versions may work but are not guaranteed

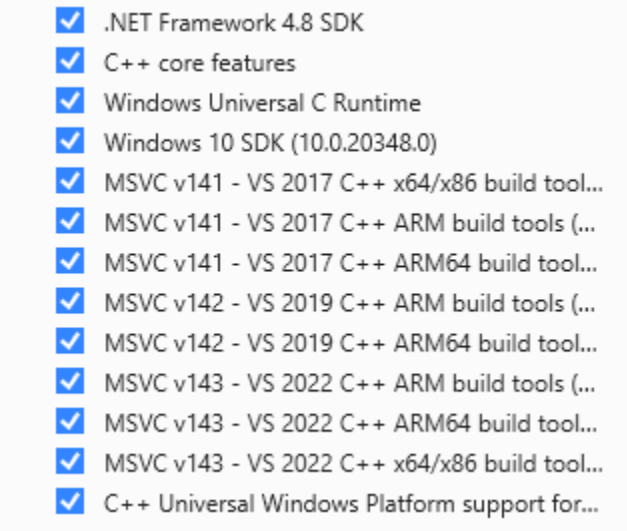
1. Java Developlent Kit (JDK)
    a. https://www.oracle.com/java/technologies/downloads/
    b. Version: JDK 19 (may be compatible with JDK 17)
    c. An installer will walk you through the installation

2. Python
    a. https://www.python.org/downloads/
    b. Version: 3.10.6
    c. A brief installer will walk you through the installation
        i. Run default options, unless otherwise desired

3. Git Bash
    a. https://git-scm.com/download/win
    b. Version: 2.38.0 (compatible with 2.37.1)
    c. An installer will walk you through the installation
        i. When asked about the default editor used by Git, go with the default "Vim" option if you don't intend to modify the code on your device. (img)
        ii. Run default options, unless otherwise desired

4. Microsoft Visual C++
    a. https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170
    b. Version: Most recent version (x86 download from above link); Version 14+ should also be compatible
    c. ***An installer will walk you through the installation

Once in the installer, go to “Individual components”. Then search for and install each of the components listed in the following image (next page). Nothing needs to be installed under Workloads or the other tabs. If prompted for Workloads, continue without them.



***There may be issues obtaining all the necessary components, specifically C++ tools for python-bioformats installation. Will be addressed later

d. It is necessary to restart your device to complete the installation

5. Miniconda
    a. https://docs.conda.io/en/latest/miniconda.html
    b. Version: 4.12.0
    c. Follow the Miniconda setup with default options – should be short, and nothing needs to be changed

2.2 Optional Software

The following software isn’t needed for the program to function, but may be helpful for connected tasks

1. ZEN Blue

   a. https://www.zeiss.com/microscopy/us/products/microscope-software/zen/free-60-day-version-of-zen-blue-edition.html
   b. A digital imaging suite provided by ZEISS. Can be used to open and manipulate .czi files from a Zeiss microscope – the primary image file type utilized in our lab

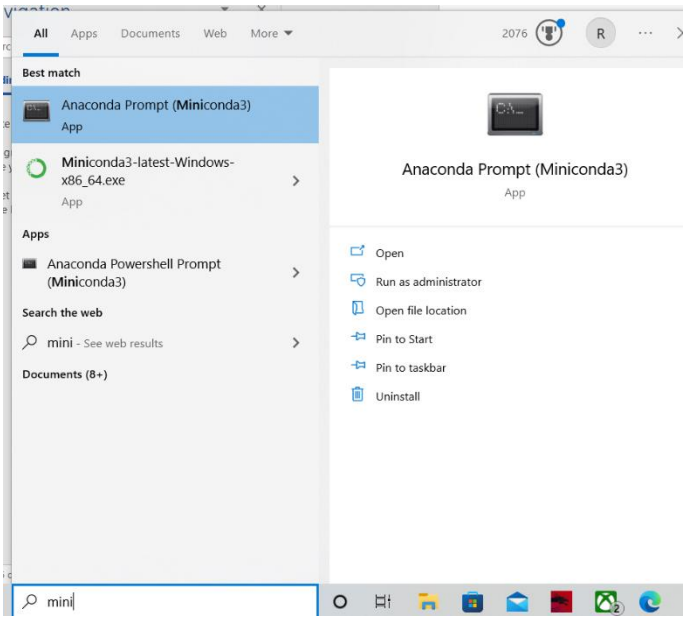
2. ImageJ
   a. https://imagej.nih.gov/ij/
   b. An image processing toolkit that provides a wide range of manual image manipulation functionality

3. Pytorch
   a. NOTE: can only be done after fully setting up the program with Miniconda
   b. This enables the program to use the computer's GPU, which may speed up analysis depending on CPU quality
   c. https://pytorch.org/
   d. To install, click the link and scroll down to "Install Pytorch", then select the most recent version of CUDA and copy the command shown in the "Run this Command" section of the page
      1) At the time of the most recent documentation update, this command should appear as:
         conda install pytorch torchvision torchaudio pytorch-cuda=11.7 -c pytorch -c nvidia
   e. Take the command from the pytorch website or above, and use it to install pytorch onto your device in the same way the other code packages and libraries are stored (see steps 1-5 of program setup w/Miniconda to see this process)

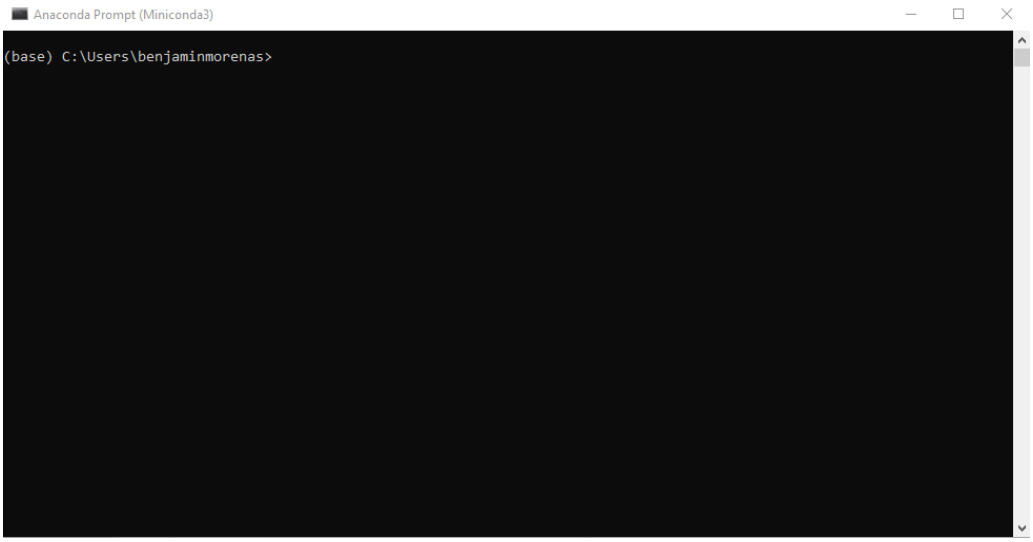
2.3 Program Setup w/ Miniconda

1) Make sure that you've installed all necessary software as described in Section 2.1

2) Run the Miniconda application, which will open an Anaconda command window



Locate the Miniconda app in whatever way is most comfortable. Windows search, shortcut (if you make one), etc.



Should open a window that looks like this, directed towards whatever your current user directory is

3) In the Miniconda prompt, create a new Anaconda environment via the following steps:

    a) Navigate to the directory where you'd like the environment to be stored

        i) This can be anywhere on the device, as long as all users of the program can access the location freely.

ii) Navigation is done using the *cd (name)* command, as in the following:

```
(base) C:\Users\benmo>cd C:/

(base) C:\>cd BioLabMain
```

cd C:/ navigates from the default path to the C: drive, and cd BioLabMain navigates to an existing folder in the C: drive named BioLabMain

b) Once in the desired directory, type *conda create --name (environment name)*

i) In this case, the (environment name) can be whatever you want it to be. In the following example, I name it "Immunostained_Image_Analysis_env"

```
(base) C:\BioLabMain>conda create --name Immunostained_Image_Analysis_env
```

c) Hit enter to run the script

d) Once the script runs, something similar to the following lines should be a part of the output:

```
#
# To activate this environment, use
#
#     $ conda activate Immunostained_Image_Analysis_env
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

i) This information relays how to activate and deactivate the newly created environment. It is recommended to save this info in some way, as the environment will need to be activated every time you want to use the program

4) In Miniconda, download the Immunostained Image Analysis program from GitHub

a) To do this, first copy the program's GitHub url (https://github.com/ninanikitina/Immunostained_Image_Analysis)

b) Through the Miniconda script, navigate to the folder where you want the program files to be installed. Again, this can be anywhere on the device as long as all potential users can access the location. In the following example, I navigate to the public documents folder on my device.

```
(base) C:\Users\benjaminmorenas>cd c:/users/public/documents

(base) c:\Users\Public\Documents>
```

c) Type *git clone (copy-paste GitHub url)* to clone the GitHub repository into this folder

```
(base) c:\Users\Public\Documents>git clone https://github.com/ninanikitina/Immunostained_Image_Analysis
```

Copy-paste can by done by right-clicking or doing Ctrl+V in the script

d) Hit Enter, and wait for the download to complete

5) Activate the environment and install all necessary packages

a) Following the instructions given in step 3-c, activate the environment by typing *conda activate (environment name)*. You can check for success by seeing if (base) at the right of each new line is replaced with the name of the activated environment.

```
(base) C:\BioLabMain>conda activate Immunostained_Image_Analysis_env

(Immunostained_Image_Analysis_env) C:\BioLabMain>
```

b) Install all the necessary packages and libraries, following the listed syntax exactly and preferably following the listed order:

i) conda install numpy
ii) pip install opencv-python==4.5.5.64 (MUST be version 4.5.5.64)
iii) conda install pandas
iv) conda install matplotlib
v) pip install trackpy
vi) pip install python-bioformats
vii) pip install scikit-image
viii) pip install customtkinter
ix) pip install torch
x) pip install torchvision
xi) conda install tqdm

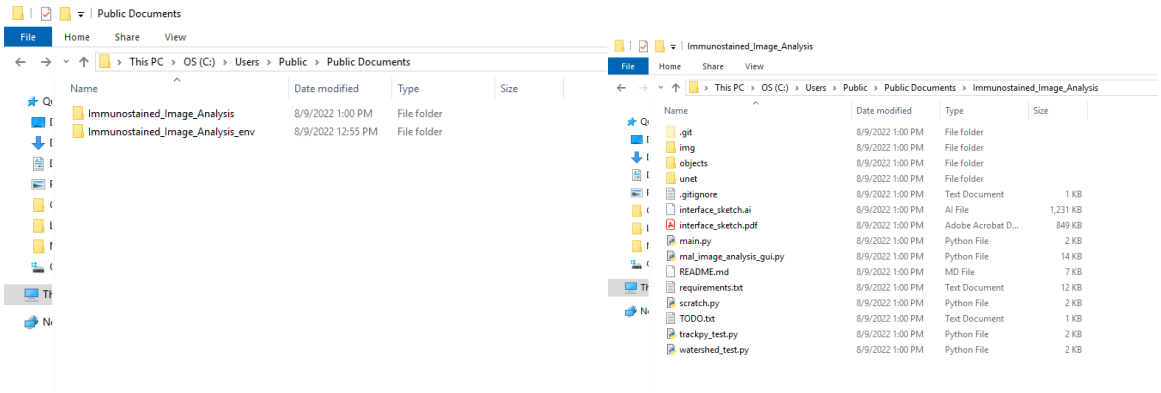
As an example, the line for installing numpy is below:

```
(c:\users\public\documents\Immunostained_Image_Analysis_env) c:\Users\Public\Documents>conda install numpy
```

Note that in the above example, the stuff boxed in red is the environment name, and the stuff boxed in yellow is your current location in the directory. The names for these will vary from user to user, but the structure will be the same.

Also note that some of the installations (the ones using "conda" rather than "pip") will require a yes/no user input to proceed. Just type *y* and hit Enter to complete these installations

6) Once steps 1-5 are complete, the Immunostained Image Analysis program will be ready to run. You can close out of the Miniconda prompt, or keep it open if you wish to run the program immediately. The following images show an example of file locations after the setup:



Program User Guide

3.1 Launching the Program

Running the program currently requires the user to execute several commands in Miniconda. In the future, it is planned to simplify this down to a single executable file

1) Run the Miniconda application, opening the script window

2) In the script, activate the environment by typing *conda activate (environment name)* and hit Enter. See the following example:

```
conda activate c:\users\public\documents\Immunostained_Image_Analysis_env
```

3) Navigate to the folder where the program is downloaded by typing *cd (path to directory)* and hit Enter. Like in the following example, but go one step deeper to
cd c:/users/public/documents/Immunostained_Image_Analysis

```
(base) C:\Users\benjaminmorenas>cd c:/users/public/documents

(base) c:\Users\Public\Documents>
```

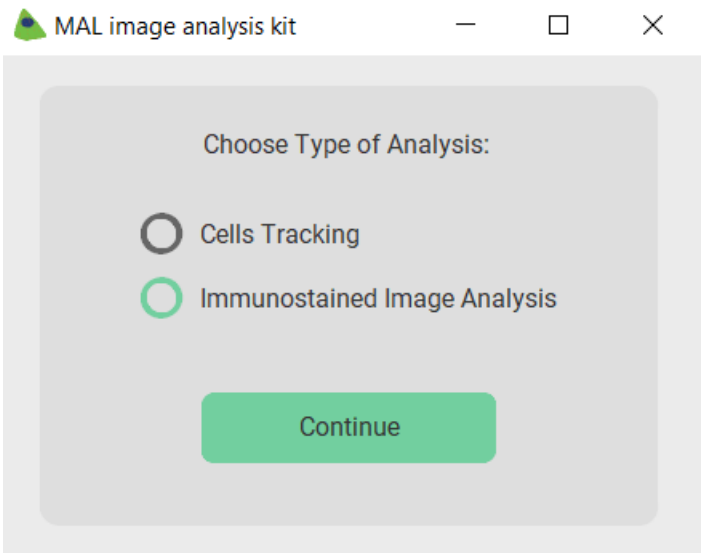
4) Run the program by typing *python mal_image_analysis_gui.py* and hit Enter

```
ined_Image_Analysis_env) c:\Users\Public\Documents\Immunostained_Image_Analysis>python mal_image_analysis_gui.py
```

It may take a few moments for the program to start running, but once it does a GUI menu will pop up that will walk you through the rest of the run

3.2 Navigating the GUI

1. Upon launching the program, the following window should pop up after a brief pause:



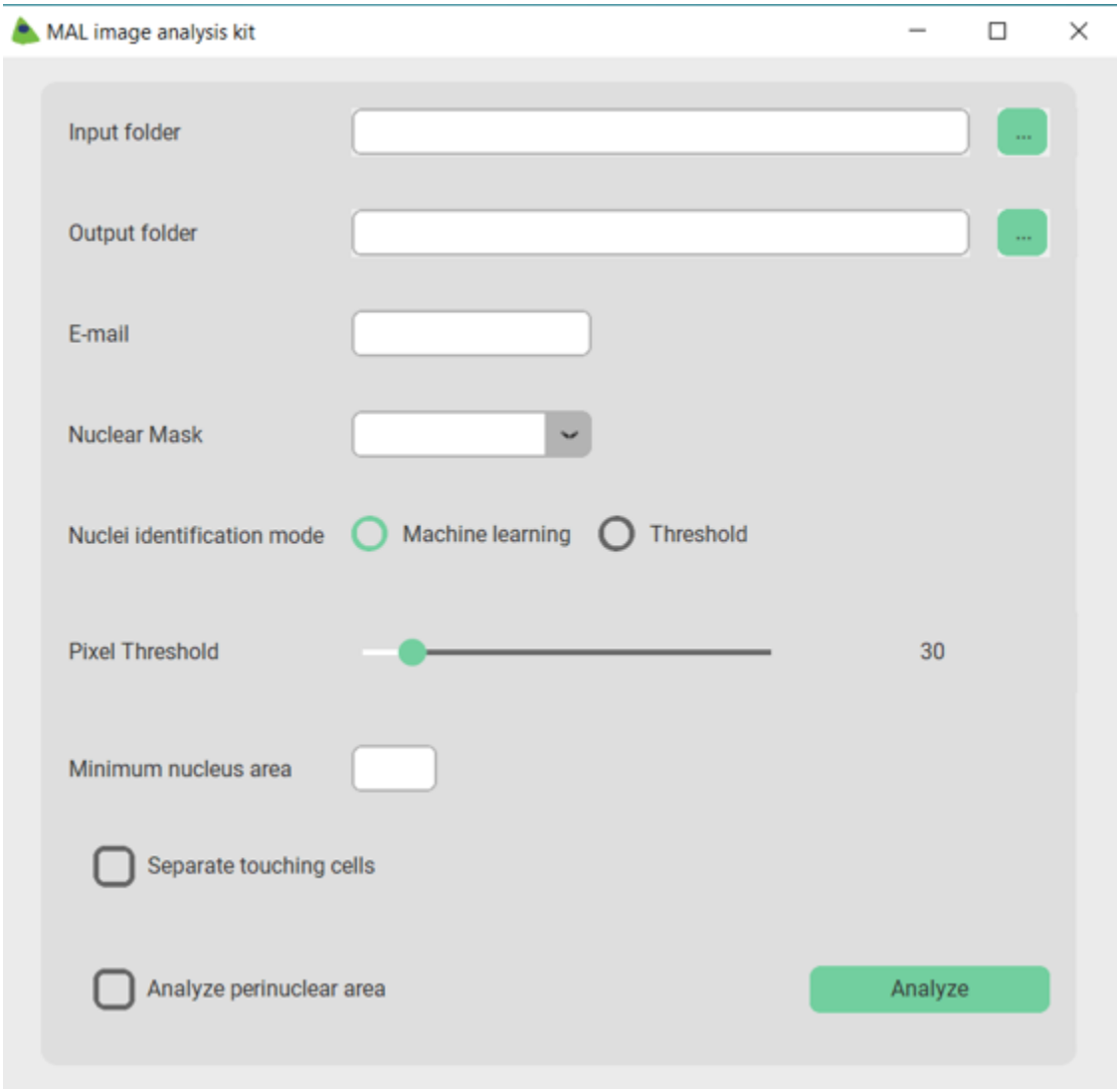
2. The user must select one of the two types of analyses:

Cells Tracking – For a .czi timelapse file (multiple frames taken over a period of time in a single file), tracks nuclei locations and movement over time. Produces a final image of the last frame in the timelapse with total movement paths drawn for each cell, as well as an Excel sheet recording every cell's location at each point in time

Immunostained Image Analysis – For either a czi still-image or timelapse, identifies nuclei and quantifies the amount of immunofluorescent stain in each nucleus across each stain channel. Produces a csv sheet recording various attributes for each nucleus in each image, including stain quantity per channel

Both analysis types also provide nuclear verification images, which are taken after each image (still image) or frame (timelapse) with white contours drawn around all objects the program recognizes as nuclei. Lastly, both analysis types produce simple Excel sheets and line graphs tracking cell population.

3. After selecting one of the options and clicking "Continue", the following window should pop up:



4. Explaining each parameter:

    a. Input Folder – The folder from which .czi files are taken. User can copy/paste the address of the folder manually or click on the three dots to the right and navigate to the folder that way.

    Note: Only have .czi files in the folder – folders and files of any other data type will break the program.

b. Output Folder – The folder where the *analysis_data* folder is created, which will hold all outputs from the program, including verification images and Excel/csv files. Can be navigated to in the same manner as the Input Folder. It also does not have the same restrictions as the Input Folder – any folder can host *analysis_data*

c. E-mail – A valid email address of the user's choice. When the program finishes running either normally or due to an error, a notification will be sent to this email address informing of the program's completion

d. Nuclear Mask – The nuclear stain the program uses to identify nuclei. Select an option through the drop down menu. DAPI is the only option at the moment, but more can be added in the future if desired

e. Nuclei identification mode – The algorithm used by the program to define and draw contours around nuclei. There are two options:

    1) Machine learning: Uses a UNet machine learning model. Takes longer to run, but generally provides more accurate nuclei identification with less user input needed. *This is the preferred algorithm for image analysis.*

    2) Threshold: Uses a simple pixel thresholding method plus some filtering techniques to identify nuclei. Runs faster than machine learning, but is generally worse at drawing contours and requires an additional user input, described below.

f. Pixel threshold – The threshold used by the *Threshold* nuclei identification mode described above. The slider bar allows the user to select a value between 0 and 255. In each image, all pixel values above the set value will be included in the final nuclear mask, while all pixel values below it will be excluded. 50 is generally a good starting point, although this can vary greatly between images.

    Note: The slider bar will be present regardless of the chosen algorithm, but it will not do anything if *Machine learning* is chosen

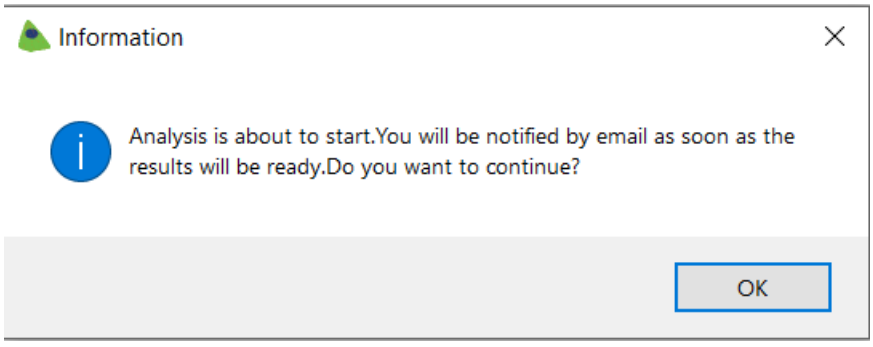
g. Minimum nucleus area – The minimum number of pixels required in a contiguous "object" for it to be considered a nucleus. The ideal value for this may vary between cell sizes and microscope magnifications, and it is planned to implement optional default values in the future.

h. *Separate touching cells – Provides the option to activate a watershed algorithm that gives the program the ability to separate different cells that are in contact with one another. Checking this box activates the functionality, if desired

    - *The current implementation of this functionality provides mixed results, so it may be practical to not use this for now

i. Analyze Perinuclear Area – Provides the option to analyze an area around nuclear regions that is slightly larger than what the program identifies. May be useful when trying to analyze for signals just beyond the nuclear regions. Note that this can also have a significant impact on calculated signal densities, etc.

5. Once all parameters are set, click "Analyze" to run the program. The following window should pop up:



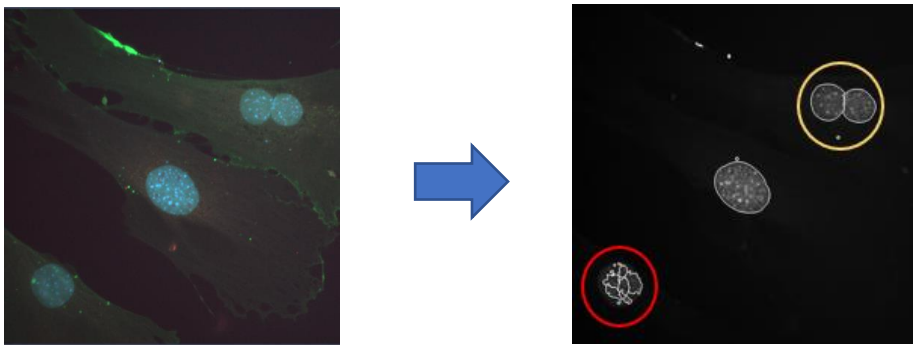
6. Click OK to run the program. Once the program terminates – whether normally or due to an error – an email notification will be sent to the provided address.

Program Features

4.1 Nuclei Identification

This project was initiated with the goal of being a quick and efficient tool for analyzing immunofluorescent stain quantities within the nuclear regions of cells, for the purpose of streamlining the research process in our lab. As such, proper nuclear identification is central to the program's success, and two main approaches were implemented to this end

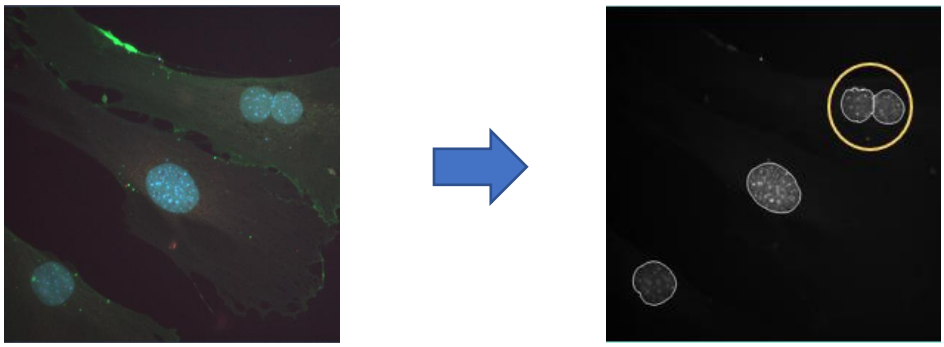
1. Pixel Thresholding - Works by simply filtering out all pixels in the provided Nuclear Mask channel (DAPI in all of our lab's samples) below a certain pixel intensity threshold, with the threshold value (between 0 and 255) being provided by the user. The upside to this approach is a lower runtime than the machine learning alternative. Downside is greater inaccuracy – see region circled in red as an example.



With pixel thresholding, the program takes the nuclear signal from the raw czi image (left) and produces a binary black-and-white mask with the nuclei (right). The area circled in red shows an example of a nucleus that this approach struggles with

2. UNet machine learning model - Works by training a UNet machine learning model with images in which cells are already marked out manually. After training, it can recognize cells in new images without the need of pre-made contours. Different models can be trained for different images differentiated by microscope magnification, treatment conditions, etc. This approach is

generally more accurate at nuclei identification. The downside is a ~3 times longer runtime per image compared to the pixel thresholding approach. However, due to the UNet model's greater accuracy, it is the method of choice for most analyses.



With the UNet approach, the program takes the nuclear signal from the raw czi image (left) and produces a binary black-and-white mask with the nuclei (right). Unlike the thresholding approach, this model was able to successfully identify all nuclei

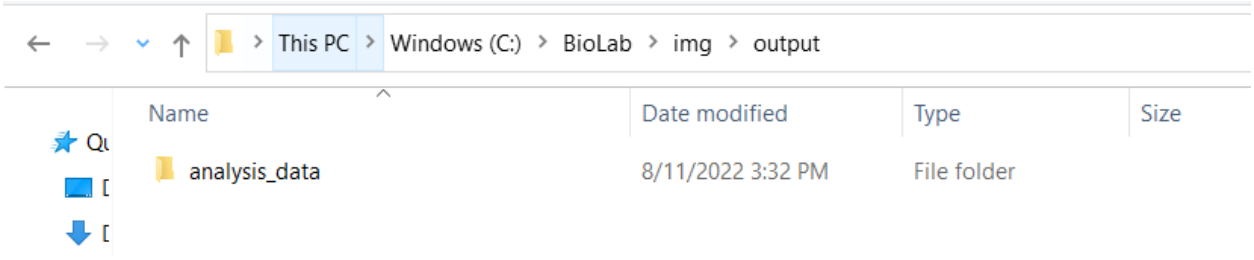
To choose which of these methods to use, select either Machine learning or Threshold in the GUI

Separating Touching Cells

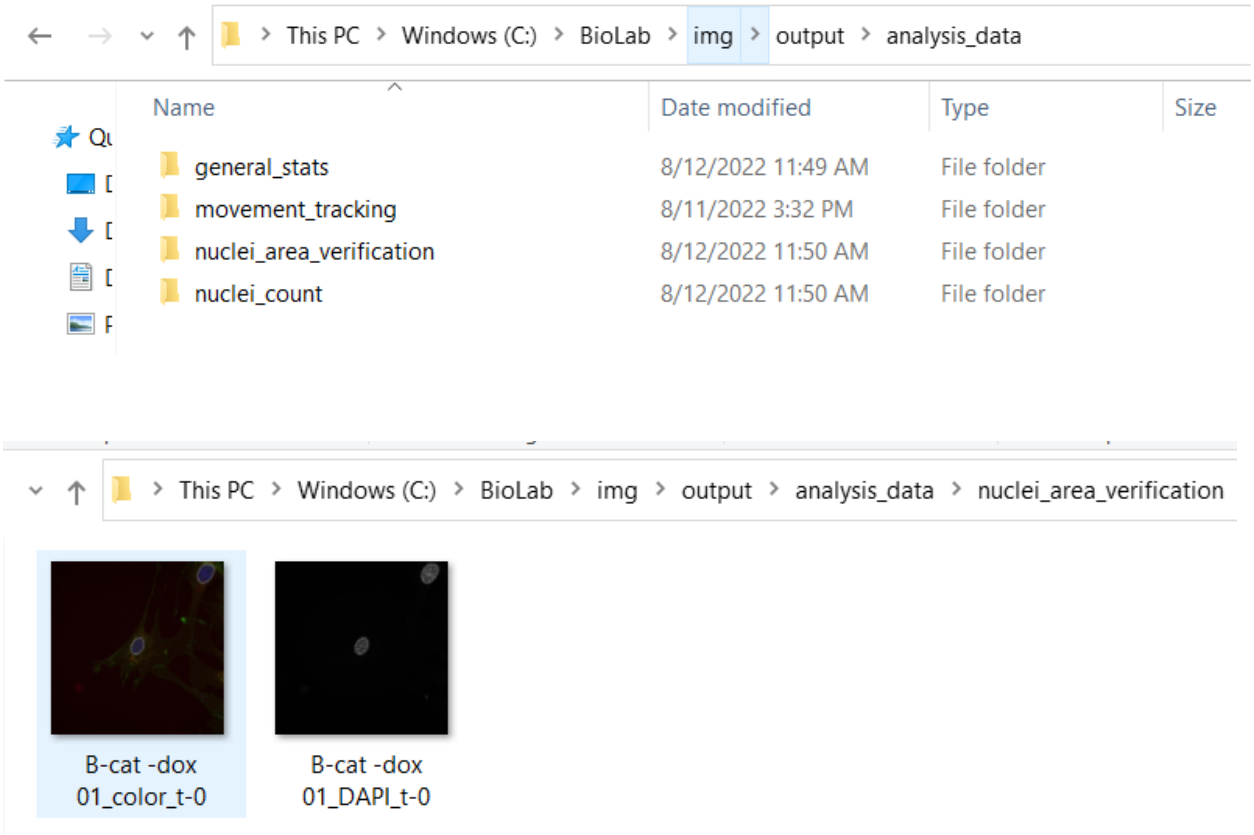
Also note that in the above images, a *watershed* algorithm is applied to separate suspected distinct cells that are in contact with one another. Such cells are circled in yellow in the examples. This functionality can vary greatly in its effectiveness due to scaling with the minimum nucleus area input. To activate/deactivate it, select or deselect the Separate touching cells parameter in the GUI.

Returned Data

Verification images with contours drawn by the program are produced in the *nuclei_area_verification* folder inside the *analysis_data* folder that's created inside the folder indicated by the Output Folder parameter. Both binary and color versions are made and can be examined visually to see what the program sees. Images for reference are on the next page.



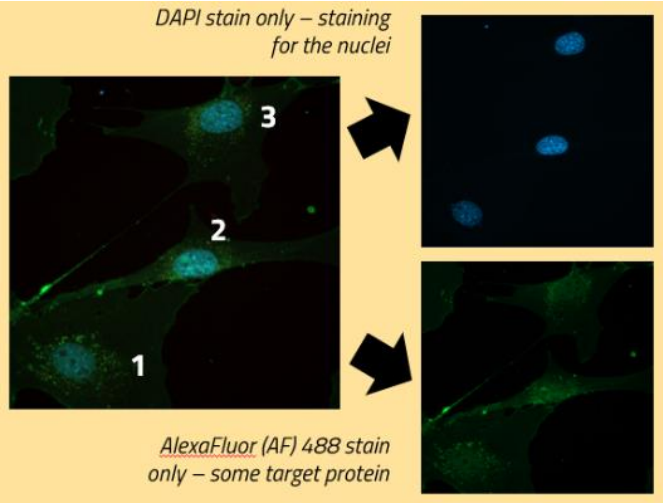
The analysis_data folder, which is created inside the designated output folder (C:/BioLab/img/output in this case)



Two verification images are produced per input image – one in color and one in black/white. Each pair is created immediately after the program finishes analyzing the image in question – no need to wait for it to finish running altogether.

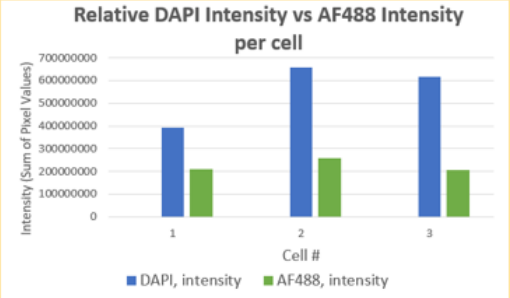
4.2 Immunofluorescent Stain Quantification

Once the program identifies the nuclear regions via the chosen method, it will then quantify the amount of each stain channel in each nucleus, which is the data we really care about. These values are calculated as a summation of each pixel's intensity within each nucleus, separated based on channel. The czi files are analyzed as 16-bit images, which range from 0 to 2^16 = 65536 in terms of value. As such, total summations can be relatively large compared to analyses of 8-bit images, but the greater amount of data means the results are more accurate.



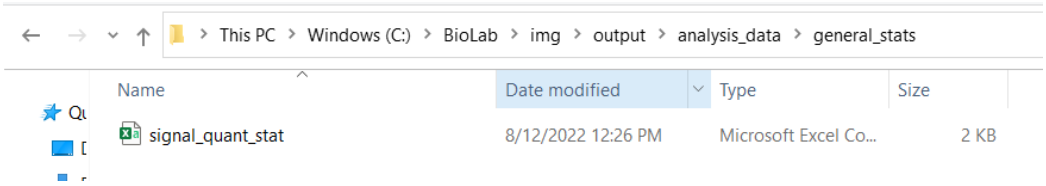
The program analyzes for stains channel-by-channel. In this example, the blue DAPI signal and the green AF488 signal will be quantified separately, each within the nuclear regions identified by the nuclei identification algorithm that is used

Cell #	DAPI Intensity	AF488 Intensity
1	3.94 x 10^8	2.1 x 10^8
2	6.59 x 10^8	2.6 x 10^8
3	6.17 x 10^8	2.07 x 10^8



Figures 1 and 2 *– Numerical quantification of stain intensities in the nuclear regions. DAPI can be consistently observed in higher quantities*

The numerical values calculated by the program for each stain channel is given in the table, while the bar graph below provides a visual representation of this data.



The signal_quant_stat file in the general_stats folder in analysis_data. This is a csv file and it is only created after every file in the input folder is analyzed.

A1 Frame

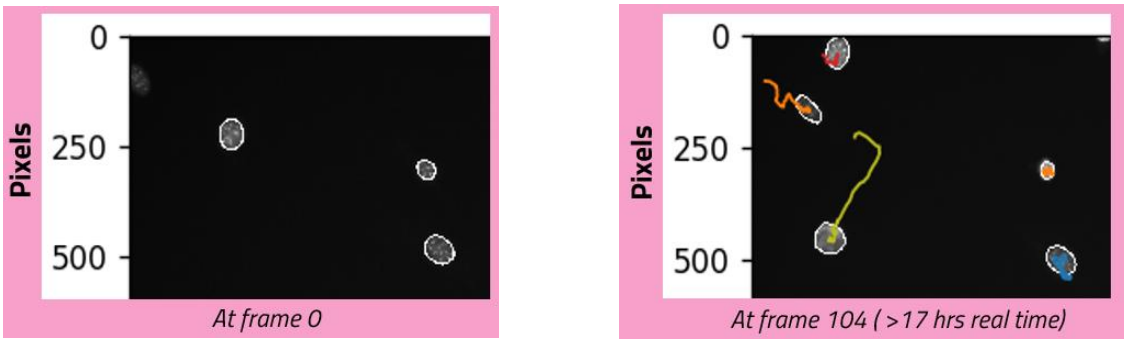
	A	B	C	D	E	F	G	H	I
1	Frame	Image nan	Cell id, #	Cell center	Nucleus ar	DAPI, inter	AF488, inte	mCher, intensity	
2	0	B-cat -dox	0	(1834, 143	32357.5	5.29E+08	1.65E+08	12279312	
3	0	B-cat -dox	1	(1040, 977	14509	2E+08	83797221	5654623	
4	0	B-cat -dox	2	(1011, 106	7294	1.02E+08	41719385	2856094	
5									
6	0	B-cat -dox	0	(1034, 974	14378.5	2.01E+08	68481014	5774505	
7	0	B-cat -dox	1	(996, 1064	12080.5	1.57E+08	58105756	4733021	
8	0	B-cat -dox	2	(1400, 115	12268.5	3.05E+08	71361470	5574729	
9									
10	0	B-cat -dox	0	(1086, 191	41736	6.03E+08	1.98E+08	15125982	
11	0	B-cat -dox	1	(581, 251)	20820.5	2.84E+08	87927056	7506257	
12	0	B-cat -dox	2	(747, 894)	22608	3.13E+08	92687548	8700303	
13	0	B-cat -dox	3	(1478, 134	21030.5	2.36E+08	67357312	7600936	
14	0	B-cat -dox	4	(1392, 140	27050	3.29E+08	88224848	9645878	
15	0	B-cat -dox	5	(471, 1492	4721.5	72765307	15626043	1768405	
16	0	B-cat -dox	6	(533, 1494	4200	52796998	12798929	1529567	
17	0	B-cat -dox	7	(1359, 153	1532.5	16845237	6869834	579492	
18	0	B-cat -dox	8	(552, 1561	5682.5	75120529	16635837	2049603	
19	0	B-cat -dox	9	(462, 1561	5849.5	1.21E+08	19130523	2184422	
20	0	B-cat -dox	10	(502, 1667	20048	2.46E+08	66211195	7270303	
21									

Some sample images (not the one in the example on the previous page) after analysis. Frame represents the "frame" in a timelapse and will always be 0 for still images. Image name is the file name of each image. Cell id, # is the designation given to each individual nucleus in each image. Cell center gives the (x,y) coordinates of each nuclear center in pixels. Nucleus area gives the area of each nucleus in pixels. Each of the channel intensity columns provides the sums of each stain type per cell. If multiple image files were analyzed, a blank row will separate the data for each one.

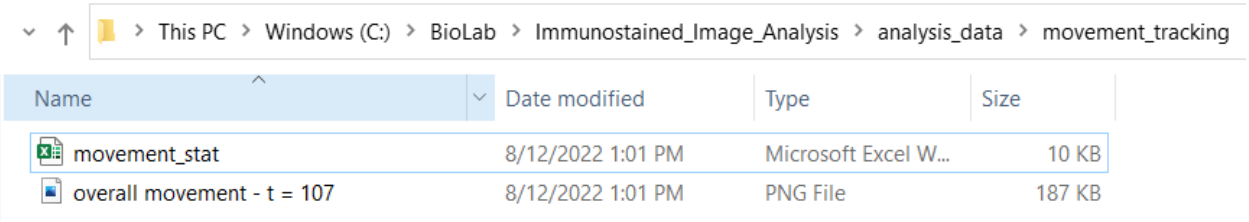
Note that there is no user input for the channels in the provided image files. As long as they are .czi files, the program can automatically detect the stain channels and quantify them separately.

4.3 Nuclei Motion Tracking

The program's newest functionality involves tracking cell movements in a timelapse and visually representing movement via path markers, which the program draws at the end of the timelapse analysis. Nucleus center data is recorded and used as the basis for cell location over time. To consistently designate which cell is which over the course of the timelapse, the program uses a combination of proximity detection and "memory" to keep track of each cell across each frame.



The above images were taken from a czi timelapse file, for which a ZEISS microscope was set to take images of the provided cell culture every 10 minutes for >17 hours overnight. Parts of the start and end frames are provided, with the latter having different colored paths drawn by the program following each cell's movements.



The movement_tracking folder in analysis_data provides two files – an Excel file providing each cell's precise location per frame, and a PNG image showing the final frame of the timelapse with movement paths drawn.

The user can choose either this functionality or the immunofluorescent image analysis in the first GUI window that pops up after running.