Clathrin Mediated Endocytosis Image Analysis Pipeline

Part 1: Spot Detection

Spot detection is done using the pylattice library which can be accessed and virtual environment with all necessary dependencies can be setup using the following github repository

<https://github.com/pylattice/pyLattice_tutorials/tree/master>

The repository contains instructions on setting up pylattice. Please note that there are some bugs in pylattice source code and edited version of source code can be found here.

*(add link)*

PyLattice Background:

1. Uses Scikit-Image function peak local maxima to find local maxima’s (based on intensity of pixels) in 3-D which returns coordinates in 3-D
2. Drops the local maxima which are within a certain defined radius (hyper parameter)
3. Then we fit a gaussian function over each axis of the returned coordinates to measure for noise in each axis (this returns standard deviation values). Points with very high standard deviations or very low standard deviations are noise and are expected to be dropped. We have to give expected standard deviation in each axis.

Movie metadata:

1. Pixel radius:
2. x-axis std dev:
3. y-axis std dev:
4. z-axis std dev:

Step 1:

Prepare separate channels of the movie in fiji for input into pylattice

Note: add how to handle large movies in fiji

Go to

1. Import File
2. Import as **Bio Formats**
3. In **memory management** click use **virtual stack**
4. Click OK

Step 2:

Refer to spots\_detection notebook in the main source repository. The notebook takes in a single channel movie and returns all the detected spots for all frames within that movie. The threshold intensity of spots and the distance between spots can be manually adjusted and to optimally decide the best threshold a single timeframe or a few time frames can be viewed using pyLattice\_demo\_pythonSegmentation notebook.

The output of spots\_detection notebook is coordinates in 3-D (x,y,z) and their standard deviations in pickle format and csv format. Each time frame is returned as a separate pickle/csv file. Which will be later on combined into one for one channel.

Step 3:

Visualize the output using napari/fiji and overlay with the raw image in background to see whether the detections make sense or not.

USES LAP\_TRACK REPOSITORY FROM THE FOLLOWING STEP AND ONWARDS

Step 4:

Now combine all the spots detected from different time frames in different dataframes into one single large dataframe. Add one column for frame number which will be used later on for tracking.

The code can be found in **01.data\_preprocessing** notebook.

Step 5:

Once we have all the detections and the standard deviations of each spot we have to filter and drop the spots which are noise. It can be done in the following ways

1. Visualize, Visualize, Visualize
2. Drop spots with very high/low sigma values. The cutoff can be determined through visualization and using statistical techniques like box and whisker plots
3. Drop spots with very high amplitude/intensity.

All of the relevant code for this part is in **02.detected\_spots\_cleaning.** This notebook also contains visualization of high and low std dev spots.

The input for this notebook comes from the output of data\_preprocessing notebook which is the large combined dataframe of all detected spots across timeframes.

Step 6:

Once we have the filtered spots from the above notebook detected\_spots\_cleaning then we can run tracking on it. The tracking configuration can be set up according to need, however, the parameters used can be found in the notebook **03.cleaned\_spots\_c3\_tracking**.

This notebook is a simple implementation of the laptrack algorithm. The virtual environment for using laptrack can be easily set up using the following GitHub repository:

<https://github.com/yfukai/laptrack/tree/main>

This repository also contains demonstrations on various types of options contained within laptrack.

Step 7:

Now that we have tracks which were obtained from running the above notebook. We need to retrieve the intensity for the other channels for the same spots. For this we can use the **04.extracting\_alt\_channel\_intensity** notebook. It allows us to extract different features from the other channel for the same spot. Like mean intensity, maximum intensity, minimum intensity etc. Moreover, we can also extract the voxel sum. The relevant needed features are appended to the main tracking dataframe.

Step 8:

Now we have the corresponding channel intensity as well. We need to filter for the tracks. The notebook with all relevant steps is **05.filtering\_tracks**. The filters that are applied are

1. Tracks with dynamin peak in first 3 frames are dropped
2. Tracks with dynamin peak in the last frame is dropped
3. Tracks with clathrin peak in the last frame is dropped

Note that this notebook returns a new dataframe which contains filtered tracks and will be used in conjunction with the original tracking dataframe for plots in next steps

Step 9:

Finally we divide the tracks into cohorts based on lifetime and plot them in the notebook **06.lifetime\_cohorts**. For plotting purposes mean intensity is used for both channels. This notebook also contains the cumulative plots for lifetime cohorts showing a general averaged trend.