Cell segmentation and tracking in microfluidics

Alexandros Papagiannakis, Christine Jacobs-Wagner lab, Sarafan ChEM-H, Stanford University, 2023

This notebook provides an example of cell segmentaion and tracking in microfluidics using the *mother_machine_segmentation* class and the *mother_machine_tracking* class, included in the *microfluidics_segmentation_ghv.py* script. This pipeline is applied on the cropped microdluidic channels, after removing any drifts in the field of view and subtracting the background in the phase contrast channel and inverting the signal for the cells to appear brighter than the background.

The cropping and alignment of the microfluidic channels, as well as the background correction and inversion of the phase contrast images was performed in MATLAB using a previously published pipeline by Wei-Hsiang Lin and Christine Jacobs Wagner (2022): https://doi.org/10.1016/j.cub.2022.07.035

In this example the *particle_tracking_microfluidics* class included in the *particle_tracking_microfluidics_ghv.py* script will also be used to detect the positions of diffraction limited particles in the cells.

In [ ]:
```
import pandas as pd
import pickle
import os
import sys
import warnings
warnings.filterwarnings('ignore')
```

1. Adding the functions/classes to the path

In [ ]:
```
# includes all functions used for the analysis of the Brownian statistics
module_path = os.path.abspath(os.path.join('..'))
if module_path not in sys.path:
    sys.path.append(".../microfluidics_segmentation_ghv.py")
    sys.path.append(".../particle_tracking_microfluidics_ghv.py")

# Check the path to confirm the analysis functions are included
print(sys.path)

%run "....\microfluidics_segmentation_ghv.py"
%run "....\particle_tracking_microfluidics_ghv.py"
```

2. Variable definition

In [ ]:
```
deback_phase_path = ".../1/c1Exp"  # folder with phase contrast tiff images
                                   # after background correction and inverse signal estimation
deback_fluor_path = ".../1/c2Exp"  # folder with fluorescence images (they can be used to assist segmentation)
phase_interval = 1 #min
experiment = 'experiment ID'
save_path = ".../results"
position = 1 # microfluidic channel position
```

3. Initialize classes

In [ ]:
```
microfluidics=mother_machine_tracking(deback_phase_path, deback_fluor_path, phase_interval, experiment, position, save_path)
```

4. Segmentation of cell labels

In this example, segmentation is not assisted by the fluorescence channel and hence the 'fluor_threshold = 0'.

In [ ]:
```
watershed_dict_fluor = microfluidics.run_segmentation(frames=(0,microfluidics.n_frames), orientation='up', channel_end=50, channel_start=15,
                                                      central_line_window=3, smoothing_factor=5, projection_width=8,
                                                      std_threshold=0.6, adaptive_smooth=4, adaptive_window=5, adaptive_offset=-9,    #0.8, -8
                                                      erosions=3, distance_threshold=2, area_threshold=50,   # 4
                                                      show_labels=False, show_threshold=False, fluor_smoothing=2, fluor_threshold=0)
```

In [ ]:
```
# save segmentation labels
with open(save_path + '/' +experiment+'_pos'+str(position)+ '_watershed_dict_fluor', 'wb') as handle:
    pickle.dump(watershed_dict_fluor, handle, protocol=pickle.HIGHEST_PROTOCOL)
```

In [ ]:
```
# load segmentation labels
with open(save_path + '/' +experiment+'_pos'+str(position)+ '_watershed_dict_fluor', 'rb') as handle:
    watershed_dict_fluor = pickle.load(handle)
```

5. Curate segmentation labels

Rules:

To split the label into two: single left click near the splitting point (the optimal splitting point will be detected)

To merge two labels: single left click on each label. The y-coordinates of the two clicks should not be more than 10 pixels apart.

In [ ]:
```
# 1st curation round
curated_watershed_dict = microfluidics.segmentation_curation(watershed_dict_fluor, length=180, height=3, edge=15, orientation='up', merging_threshold=15, cmap_='tab20b')
```

In [ ]:
```
# This curation step can be repeated as many times as needed
curated_watershed_dict = microfluidics.segmentation_curation(curated_watershed_dict, length=180, height=3, edge=15,  orientation='up', merging_threshold=15, cmap_='tab20b')
```

In [ ]:
```
# save curated labels
with open(save_path + '/' +experiment+'_pos'+str(position)+ '_curated_watershed_dict', 'wb') as handle:
    pickle.dump(curated_watershed_dict, handle, protocol=pickle.HIGHEST_PROTOCOL)
```

In [ ]:
```
# load curated labels
with open(save_path + '/' +experiment+'_pos'+str(position)+ '_curated_watershed_dict', 'rb') as handle:
    curated_watershed_dict = pickle.load(handle)
```

6. Cell tracking

A curation is applied removing small cell labels and cells near the exit of the microfluidic channel.

In [ ]:
```
# tracking of cell labels (with area > 40px in this example)
cell_id_df = microfluidics.track_cells(curated_watershed_dict, min_area=40, frame_range=(0,microfluidics.n_frames), max_distance=15, area_ratio_range=(0.6,1.4))
# curation
cell_id_df_cur = cell_id_df[cell_id_df.y>10] # removes the cells near the exit
# saving the tracked labels
cell_id_df_cur.to_pickle(save_path + '/' +experiment+'_pos'+str(position)+ '_cell_df', compression='zip')
```

7. DIffraction limited particle detection (GFP-μNS)

In [ ]:
```
# specify the path with the particle tiff images
particle_images_path = ".../images_path"
# initialize the particle tracking class
micro_par = particle_tracking_microfluidics(images_path, save_path, experiment, 1, 2, 8)
# run particle detection
micro_par.getting_the_particles(log_adaptive_parameters=[2.0, 1000, 95.0, 1.0, 9, -5.0, 0],
                                min_particle_size=3, max_particle_size=80,
                                min_particle_aspect_ratio=0.3, post_processing_threshold=90, box_size=7,
                                analysis_range=(0,microfluidics.n_frames), metric='raw pixels', operation='sum',
                                gaussian_fit_show = False)
# The Pandas dataframe with the detected particle positions is saved in the results folder with the suffix "...particles_df"
```