Microfluidics droplet tracking for immunofluorescence research

A. Basseto, F. Cunha, F. Da Dalt, V. Gillioz, S. Jain

ETH Zurich, Switzerland

1 Motivation

Droplet based microfluidics allows for biomedical study of isolated cells. However, droplet displacement in the liquid surrounding makes it difficult to observe the evolution and behaviour of individual cells through time. In our case, the cell samples are hard to acquire, so there is a need to optimise their use. Moreover, the current model requires to spend a lot of time checking droplets by eye. A method for droplet tracking through time would therefore be desirable to increase the analysis capacities in droplet-based microfluidic research.

2 Overview

In this project, we received microscopic images containing thousands of droplets each. Some of the droplets contained cells that the researchers wanted to study. Most of the droplets contained magnetic beads to allow immunofluorescence analysis of the cells' behavior. In the images, droplets are around 40x40 pixels. Images were taken every 30 minutes, starting right after the cover glass filling, for a duration of 4 to 4.5 hours. The images contained 5 channels :

- a Bright-field image where the droplets, cells and beads are visible
- a DAPI channel where the cells are visible
- three channels for the fluorescence analysis that we did not make use of

In this context, three major difficulties needed to be overcome :

- Large displacement between frames due to high time interval (30 minutes)
- Visual cues are not robust due to appearance changes of the droplets and cells
- Hard to distinguish droplets due to high visual similarities

3 Method

Figure 2 shows an end-to-end architecture of the methods used to perform droplet tracking. The detailed description of each of the methods are as follows:

3.1 Preprocessing

1. We suppress very dark and very bright pixels by thresholding at very extreme quantiles.
2. We take a local-median-filtered version of the image and then subtract this filtered image from the original image.

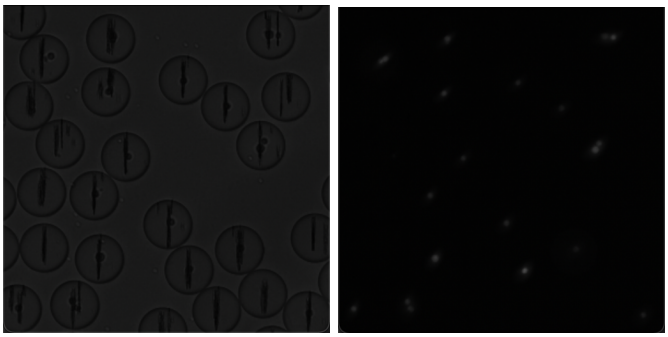

Figure 1. Typical brightfield (left) and fluorescent (right) channels

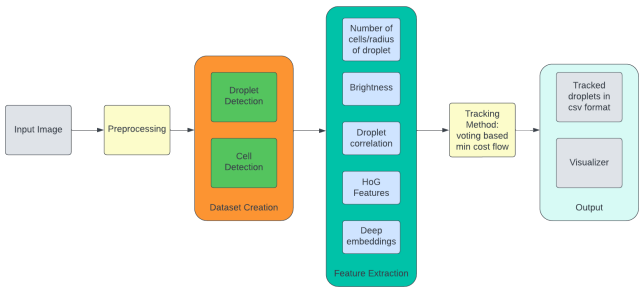

Figure 2. Droplet Tracking Architecture Overview

3. Then we compute a local-rank-equalized version of the image and use it to weight the pixels.
4. Then we apply a threshold that sets the bottom 50% of darkest pixels to zero since typically the 50% of darkest pixels is always background.
5. We finish off by applying a Gaussian sharpening filter.

3.2 Dataset Creation

3.2.1 Droplet detection

Droplets are detected by first using OpenCV's Hough transform[1] and then we refine the detection by a RANSAC algorithm[2] which can be used for robust fitting in noisy environments.

3.2.2 Cell detection

1. First, we create a "hypothesis" of where we have for sure background in the droplet image.
2. Then we find "valleys" in the DAPI channel. We extend the background hypothesis along these valleys by means of morphological operators.
3. Then we do some final morphological closing-operations to fill small gaps in the background hypothesis.

4. To find peaks we simply find all peaks and filter out those peaks that are in the background hypothesis. All peaks that survive this process are considered significant peaks.

3.3 Feature Extraction

The following features were extracted using the created dataset:

- **Basic Droplet Features**: these comprise features like the number of cells/spikes and the radius of the droplet.
- **Correlation Features**: All correlations/similarities between droplets for both the DAPI and Bright-field channels.
- **Brightness**: Difference between integrated/maximum brightness for every pair of droplets.
- **HoG Features**: Histogram of Oriented Gradient is a feature extraction technique in which counts of gradient orientation are computed in a localized portion of the image. [3]
- **Deep Learning Based Feature Extraction**: In order to extract more complex features from the dataset, we use a Masked Autoencoder (MAE) [4].

  The goal of the model is to reconstruct the original signal - an image of a droplet - given its partial observation. In order to achieve this, the original image is divided into a series of non-overlapping patches, and afterwards a random subset of these patches is removed. This will make the task of reconstructing the image from features harder, as it requires the model to have a holistic understanding of the input beyond low-level image statistics. The remaining patches are then fed to an asymmetric autoencoder model.

  Encoder. The chosen encoder is a standard ViT [5]. The fact that the model only operates on a subset of the original images enables it to be trained on local machines in a relatively short time span when compared to the alternative - making this approach quite scalable. As its output, the encoder will produce a series of features, which are afterwards passed downstream to the next step in the pipeline.

  Decoder. Taking into consideration that the latent encodings are the sole reason for the model's usage and that the input's reconstruction should just be a step to achieve this goal, the used decoder is very small (in comparison with the encoder). This promotes the creation of very expressive features, that can be used to recreate an input with very low reconstruction loss even when using a very simple decoder.

3.4 Tracking

A Hierarchical linking approach was followed to track the droplets:

1. **Neighbourhood Selection**: A user-defined neighbourhood was selected for a particular droplet in the next frame.
2. **Feature-based weighted voting**: Based on the computed features, each of the droplets in the neighbourhood received votes; if the features were close then a higher vote was given and vice-versa.
3. **Validity mask creation**: Vote-based thresholding was done to create a validity mask and so as to link the droplet with the most similar droplets.
4. **Linking**: Min-cost flow algorithm based on Google ortools[6] was used to link the droplets using input as a distance matrix (Euclidean distance between every pair of droplets in the neighbourhood) along with some other features and the validity mask.

3.5 Visualizer

Due to the highly sensitive nature of the experiments, it is imperative researchers can be completely confident in the tracking results. The only way to achieve the level of confidence needed is to allow for human interaction to the extent of a verification of each trajectory. In order to make this as efficient as possible, we developed a visualiser to achieve that task.

The visualizer takes two inputs, the images being studied and the tracking results. It draws dots connected by a line for each trajectory in the 2D space of the image. It also allows for the user to display, on top of the trajectories, any frame of their choosing. The user can display multiple frames at a time too, each being somewhat transparent. Each frame's visibility can be toggled on or off. This allows for the verification of tracking results. On top of this, our visualizer allows for quality-of-life image visualisation tools such as a zoom, a pan or a reset to the original view.

To act upon their judgment of the tracking results, we offer researchers more tools. First a lasso selection that will toggle a "keep" state for each trajectory, represented by a different color visually. Second, a cut tool allows to cut a trajectory in two. Third, a swap tool that allows to swap two edges between two trajectories. Those two last tools came from an exploration of our tracker's limitations and recurring mistakes that would be easily fixed by human input.

Once the researcher has selected all trajectories they want to keep, they can press a button to save an updated tracking result keeping only those.

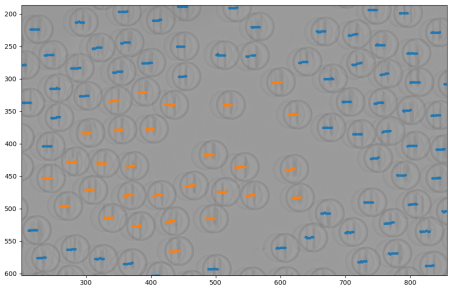

Figure 3. Visualizer example. This shows the visualizer with a zoomed-in view, some trajectories selected, and the first and last frame superimposed.

References

[1] https://en.wikipedia.org/wiki/Circle_Hough_Transform. Accessed: 2023-02-19.

[2] https://sdg002.github.io/ransac-circle/index.html. Accessed: 2023-02-19.

[3] https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f. Accessed: 2023-02-19.

[4] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv:2111.06377*, 2021.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.

[6] https://developers.google.com/optimization/flow/assignment_min_cost_flow. Accessed: 2023-02-19.