IGFL-FR

Authors: Ko Sugawara

Email: ko.sugawara@ens-lyon.fr

Platform: Linux

Prerequisites: Java 8+, CUDA-capable GPU

IGFL-FR: SUMMARY

Our method has originally been developed for the GUI-based semi-automated tracking system called ELEPHANT [1, 2] that implements client-server architecture where the Java-based client [3] is responsible for the data models and the postprocessing algorithms, and the Python-based server provides deep-learning algorithms. In the scripts dedicated to the CTC, Java programs are executed first, in which Python scripts are called, exchanging the data via .json files to mimic the interactions between the client and the server. Our method employs a tracking-by-detection paradigm, which first detects nuclei in 3D and subsequently links them to yield tracks where nuclei are approximated by ellipsoids. We use two independent 3D U-Net [4, 5] CNNs in the segmentation and tracking workflows. We used custom annotations on the **Fluo-N3DH-CE** training sequences using ELEPHANT to train the CNNs. The pretrained models were then applied to the two challenge sequences.

IGFL-FR: PREPROCESSING

In the segmentation step, we corrected the uneven background levels across the *z*-slices by shifting the slice-wise median value to the volume-wise median value.

IGFL-FR: SEGMENTATION

Segmentation of nuclei is performed in the following steps. First, the preprocessed image volumes are fed into the pre-trained U-Net CNN to output the probability maps for nucleus center, nucleus periphery, and background. Second, a post-processing workflow extracts nucleus center voxels as connected components and approximates them as ellipsoids. Finally, these ellipsoids are drawn in 3D to yield the segmentation results, where the overlapping ellipsoids are rendered in a way that they share intersection to the same extent.

IGFL-FR: TRACKING

Tracking is performed by combining optical flow estimation and nearest-neighbour linking in the following steps. First, the two consecutive image volumes are fed into the pre-trained U-Net CNN to output the 3D optical flow. Second, the detected nuclei are linked in two steps: (i) shifting the positions of nuclei in time-point *t+1* based on the estimated optical flow, (ii) looking up the parent nuclei in time-point *t* by nearest-neighbour algorithm. Each nucleus accepts one or two links, determined by the estimated displacements of the nuclei and the actual distances of the links. If there are competing links beyond the allowed maximum of two links, the links with smaller estimated displacements are adopted and the remaining nucleus looks for the next closest nucleus up to three neighbors. This procedure is repeated up to five times to generate the links.

IGFL-FR: POST-PROCESSING

We implement a postprocessing workflow for interpolating missing nuclei in the tracks. In this workflow, each orphan nucleus tries to find its parent up to five time-points back using the estimated coordinate calculated from the flow prediction. If it finds its parent, the links are generated with the interpolated nuclei in between.

REFERENCES

1. Sugawara K, Cevrim C, Averof M. Tracking cell lineages in 3D by incremental deep learning. bioRxiv: 2021.02.26.432552, 2021.
2. https://elephant-track.github.io
3. https://github.com/mastodon-sc/mastodon
4. Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of Medical Image Computing and Computer-Assisted Intervention*, 234-241 (2015).
5. Cicek O, Abdulkadir A, Lienkamp SS, Brox T, Ronneberger O. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. In *Proceedings of Medical Image Computing and Computer-Assisted Intervention*, 424-432 (2016).