KTH-SE

Authors: Klas E. G. Magnusson, Joakim Jaldén, Helen M. Blau

Email: klasmagnus@gmail.com

Platform: Windows

Prerequisites: MATLAB 2019b (x64) or higher

KTH-SE: SUMMARY

We have used a tracking by detection framework with four separate segmentation algorithms and a track linking algorithm based on the Viterbi algorithm. For the **Fluo-N3DL-DRO**, **Fluo-N3DL-TRIC**, and **Fluo-N3DL-TRIF** datasets, we also used a detection pre-processing algorithm based on GM-PHD filtering, which allows us to use dynamic motion models in the track linking step. When possible, we used a search algorithm to optimize the segmentation parameters, but in some cases we got better results by optimizing the parameters manually in a graphical user interface. Automatically optimized parameter values are underlined in the text below as well as in **Parameter Configurations**.

KTH-SE: PREPROCESSING

All images were converted to 64-bit double images with a saturation intensity of 1. Furthermore, to get rid of background features in **Fluo-C2DL-MSC**, **BF-C2DLHSC**, **BF-C2DL-MuSC**, and **PhC-C2DH-U373**, we subtracted background images before we applied the respective segmentation algorithms. In **Fluo-C2DL-MSC**, the background image was computed as the minimum intensity for each pixel position, taken over the time dimension of the sequence. In **BF-C2DL-HSC**, **BF-C2DL-MuSC**, and **PhC-C2DH-U373**, the median intensity was used instead of the minimum intensity. In **BF-C2DL-MuSC**, the appearance of the image changes over time. Therefore, we computed separate background images for different time intervals. To deal with gradual changes of the image, we fitted a linear combination of background images to each image and then subtracted that linear combination. Background images were computed for the frame intervals 1-321, 322-421, 422-875, 876-908, and 909-1376.

KTH-SE: SEGMENTATION

We used four different segmentation algorithms to generate the binary segmentation masks, which we then post-processed to extract cell regions. When possible, we used a search algorithm to optimize the segmentation parameters, but in some cases we got better results by optimizing the parameters manually in a graphical user interface. Automatically optimized parameter values are underlined in the descriptions below. The z-stacks of the **Fluo-N3DL-TRIF** dataset are so large that we decided to break each z-stack into 16 sub-volumes and segment the sub-volumes separately in order to use less memory. The sub-volumes overlapped by 100 voxels in each direction and after they were segmented, the segmented blobs from all sub-volumes were combined into a segmentation result for the entire z-stack.

Bandpass segmentation. To segment all of the **Fluo** datasets and **PhC-C2DL-PSC**, we used the bandpass filtering based segmentation algorithm that we presented in [1]. We performed the filtering by convolving the original image I with two different Gaussian filters G_S and G_B, with covariance matrices $\Sigma_\mathrm{S} = \sigma_\mathrm{S}^2\Sigma$ and $\Sigma_\mathrm{B} = \sigma_\mathrm{B}^2\Sigma$. In two dimensions, Σ is the 2×2 identity matrix, and in three dimensions $\Sigma = \mathrm{diag}(1, 1, 1/r^2)$, where r is the ratio between the voxel height and the voxel width. The two filtered images are given by $I_\mathrm{S} = I * G_\mathrm{S}$ and $I_\mathrm{B} = I * G_\mathrm{B}$, and the bandpass filtered image is computed as $I_\mathrm{BP} = I_\mathrm{S} - \alpha I_\mathrm{B}$, where α is a free parameter. The binary segmentation mask is obtained by applying the threshold τ to I_BP. To avoid under-segmentation of dim objects that are close to bright objects, we preprocessed some of the datasets using intensity clipping, where all pixel values above I_max are set to I_max. In **Fluo-C2DL-MSC** we applied a tophat filter with a radius of 300 pixels to remove background intensity. In **Fluo-N3DH-CE** the noise properties are different in the different image dimensions, and therefore we used a 5×1×3 median filter to reduce the noise before we applied the bandpass filter. In **Fluo-N3DL-TRIC** and **Fluo-N3DL-TRIF** we applied 2D tophat filters with radii of 15 and 18 pixels respectively to each z-slice to remove the high background in regions with densely packed nuclei. In **Fluo-N3DH-CE** and **PhC-C2DL-PSC**, we used different values for σ_S and σ_B for the first and the last image of the sequence, and used linear interpolation to compute different values for each image in between. In the Fluo-N3DH-CE dataset, the same linear functions are used for both sequences, but the sequences have different lengths and therefore they have different settings in the last image.

Variance segmentation. To segment the cells in **BF-C2DL-MuSC** and **PhC-C2DH-U373**, we computed a texture image representing the intensity variance in a region around each pixel in the original image. This technique has been used previously to segment cells in transmission microscopy images [2, 3]. For **BF-C2DL-MuSC**, we computed the variance in a square region of 7×7 pixels. In **PhC-C2DH-U373**, we weighted the surrounding pixels using a Gaussian kernel G with covariance matrix $\Sigma_\mathrm{var} = \sigma_\mathrm{var}^2\mathbf{I_2}$, where $\mathbf{I_2}$ is the 2 × 2 identity matrix. We computed the weighted local variance image V as $(G * I^2)/ (G * \boldsymbol{1})- (G * I)^2/ (G * \boldsymbol{1})^2$, where I^2 is an image with the squared pixel intensities and $\mathbf{1}$ is an image with all ones. The obtained variance image was thresholded using a threshold τ_var to give a binary segmentation mask. We used the parameter values σ_{var} = <u>1.88</u> and τ_{var} = <u>5.57E-5</u> for **PhC-C2DH-U373** and τ_{var} = <u>1.44</u> for **BF-C2DL-MuSC**.

Ridge segmentation. To segment cells in **DIC-C2DH-HeLa**, we developed an algorithm inspired by the algorithm used to segment muscle fibers in [4]. We first applied a ridge detection filter similar to the filter described in [4], to highlight the boundaries between the cells. The ridge detection was done by smoothing the image with Gaussian kernels with standard deviations σ of 5, 6, 7, 8, 9, and 10 pixels and computing the Hessian at each pixel of the six resulting images. The ridge image $v_0(\sigma)$ at the scale σ was then computed from the eigenvalues λ_1 and λ_2, where $\lambda_1 \leq \lambda_2$, of the corresponding Hessians as

$$v_0(\sigma) = \begin{cases} 0 & \text{if } \lambda_1 > 0, \\ \exp\left(\frac{-R_B}{\gamma^2}\right)\left(1 - \exp\left(\frac{-S}{\beta^2}\right)\right) & \text{otherwise,} \end{cases}$$

where $R_B = |\lambda_2|/|\lambda_1|$ and $S = (\lambda_1)^2 + (\lambda_2)^2$. We used γ=1 and β=10. The final ridge image was obtained by taking the pixel-wise maximum of $v_0(\sigma)$ over all σ and smoothing using a Gaussian filter with a standard deviation of 1 pixel. Once we had the ridge image, we transformed the intensities using the function $f(x) = \text{asinh}(20x)$, to enhance dim ridges, and divided by the mean intensity of the transformed image. Then, we thresholded the ridge image at $T = 0.75$, and skeletonized the resulting binary mask to extract cell boundaries. To determine which of the resulting regions were cells and which were background, we computed a local variance image where each pixel value represented the sample variance in a 9×9 neighborhood of the corresponding pixel in the original image. Regions with an average local variance above 0.0005 were considered to be cell regions. To fill in gaps in the skeletonized boundaries, we detected all end points of the skeleton and connected pairs of them by straight lines. End points were connected if they were no more than 50 pixels apart, and if the added line cut through a single segment, without generating a fragment smaller than 7500 pixels. If one of the new regions would become a background region, the size threshold was instead set to 200 pixels, as the operation would not split a cell in two. After joining end points, we removed cracks in regions by erasing all boundary pixels, which were bordering a single region. Then we merged the background regions and the border pixels into a single background region. Finally, we merged cell regions with less than 7500 pixels into adjacent cell regions until all cell regions either had at least 7500 pixels or were surrounded by background pixels.

Template matching. To segment the **BF-C2DL-HSC** dataset, we used a segmentation algorithm based on template matching. In the algorithm, the cells are compared to a template, which in our case is a tightly cropped 23×23 pixel image of a single representative cell in the training data. The segmentation algorithm computes the correlation coefficient between the template and all image regions of the same size as the template. This produces a correlation coefficient image which has local maxima on the centers of the cells. To handle cells of different sizes, the template is scaled to have side lengths of 19, 21, 23, 25, and 27 pixels. The cells are then detected as local maxima in a maximum intensity projection over the different sizes. For each local maximum, the cell size is taken to be the size which has the highest value for the local maximum pixel. The detections with a correlation coefficient above τ_{temp} = 0.45 are then converted into pixel masks. Circular pixel regions of the same sizes as the templates are created around the local maxima. The detections are added in order of decreasing correlation coefficients and detections which are closer than 10 pixels from an already added detection are discarded. Pixels that are inside multiple circles are assigned to the closest local maxima. The local variance segmentation algorithm that was used for **BF-C2DL-MuSC** was used as a secondary segmentation algorithm, to deal with cells that do not fit the template. In this case, the variance was computed in a square region of 5x5 pixels and thresholded with τ_{var} = 2.

Post-processing. To break regions with multiple cells into individual cell regions, we applied a seeded watershed transform (*watersheds*) to the image intensity (wI), the bandpass filtered image (wB), or the distance transform (wS) of the binary segmentation mask. The pixel values in the distance transform are the Euclidean distances to the closest background pixels. For z-stacks, where the voxel height was different from the voxel width, we used the anisotropic distance transform [5], where the distance between z-planes is different from the distance between neighboring voxels in the same plane. In **Fluo-N3DH-CE**, **Fluo-N3DL-DRO** and **Fluo-N3DL-TRIC**, this did however give poor separation boundaries between the watersheds, as the distance between z-planes was too large. To avoid these problems, we inserted virtual z-planes between adjacent z-planes in the distance transform. We assigned values to the virtual z-planes using linear interpolation, ran the watershed transform and then removed the virtual planes. We used nine virtual z-planes for **Fluo-N3DH-CE** and two for **Fluo-N3DL-DRO** and **Fluo-N3DL-TRIC**. For all datasets, the watershed transform was constrained to the foreground pixels of the binary segmentation mask, to speed up the computation, and to avoid getting watersheds, which overlap with multiple cell regions. To avoid over-segmentation, we applied Gaussian smoothing with a standard deviation of σ_W, and/or an *h*-minima transform with an *h*-value of H_{min}. In **Fluo-N2DH-GOWT1**, we also removed watershed seeds with a distance transform value below 10 pixels, to further reduce over-segmentation. In **BF-C2DL-MuSC**, **Fluo-C2DL-MSC**, **Fluo-N3DL-TRIC**, **Fluo-N3DL-TRIF**, and **PhC-C2DL-PSC**, we applied an additional watershed transform, after the first one, to break even more clusters into individual cells. To get rid of regions without cells, we removed regions with fewer than $A_{\min}$ voxels, and regions where the summed voxel intensity was below $S_{\min}$. To compute the summed voxel intensity, we subtracted the minimum value of the image, and summed all voxels inside the segmented region. In **Fluo-N3DL-DRO** and **Fluo-N3DL-TRIC**, we also removed regions larger than 10000 voxels. For some datasets, we applied morphological operators to the extracted cell regions. We filled in holes in the segments of all datasets. In the **Fluo-N2DH-SIM+-02** image sequence and in **FluoN3DH-SIM+**, we added all pixels inside the convex hulls of the original regions. Whenever a pixel was in the convex hull of multiple regions, we did not add it to any of them. In **Fluo-N2DH-GOWT1** there were also pieces missing from the segments, but the true regions were not always convex, so to fill in missing parts, we instead applied morphological closing with a circular structuring element with a radius of 12.2 pixels. The variance-based segmentation of **BF-C2DL-MuSC**, **BF-C2DL-HSC**, and **PhC-C2DH-U373** tends to give too large regions, due to the large kernel size used to compute the variance and the weighted variance. To overcome this problem, we applied morphological erosion with a square structuring element of 7 × 7 pixels for **BF-C2DL-MuSC** and **BF-C2DL-HSC**, and with a circular structuring element of a radius of 8.31 pixels for **PhC-C2DH-U373**. Furthermore, in **BF-C2DL-HSC**, the variance-based segmentation is used to create a binary segmentation mask, the pixels segmented by template matching are removed, and then morphological opening with a circular structuring element with a radius of 6 pixels is used to get rid of thin fragments that have been segmented between the cells detected by template matching. In **DIC-C2DH-HeLa** there was quite a lot of over-segmentation, but in many cases over-segmented regions were correctly segmented in adjacent images. We therefore tried to reduce the over-segmentation by looking for cases where multiple cells overlapped with the same region in an adjacent image. If the fragments were smaller than 15000 pixels and had at least 60 % of their pixels in common with the region in the adjacent image, they were merged into a single region.

Parameter optimization. For many of the datasets we used an automated search algorithm to optimize the segmentation parameters. The search algorithm used a type of coordinate ascent with variable step length to optimize the individual parameters one at a time. The parameters were initialized using manual tweaking, and the step lengths were set to 10 % of the initial values. In each optimization iteration of the optimization, the algorithm goes through the parameters one at a time and tries both increasing and decreasing them by the corresponding step lengths. The parameters are adjusted to the best value if either of the options gives a better result. If a better segmentation is found, the step length is increased by 20 % and otherwise it is decreased by 20 %. We used **SEG** as utility function for the optimization and ran it for 25 iterations. For **Fluo-C2DL-MSC**, **Fluo-N2DH-SIM+**, **Fluo-N3DH-SIM+**, **Fluo-C3DH-A549**, and **Fluo-C3DH-A549-SIM**, the parameters were optimized separately for each image sequence, but for all other datasets, the optimization was performed over all image sequences simultaneously, on the average **SEG**.

KTH-SE: TRACKING

For all datasets except **Fluo-N3DL-DRO**, **Fluo-N3DL-TRIC**, and **Fluo-N3DL-TRIF**, we applied our global track linking algorithm [3] directly to the detected cell regions. For **Fluo-N3DL-DRO**, **Fluo-N3DL-TRIC**, and **Fluo-N3DL-TRIF** we used a detection preprocessing algorithm [6], which takes advantage of the dynamic nature of the nuclei motion by preprocessing the detected locations using a Gaussian Mixture Probability Hypothesis Density (GM-PHD) filter [7]. Once we had preprocessed the locations, we linked them using the track linking algorithm presented in [3]. The algorithm considers the *n* most likely cell migrations to and from each detected cell region in the image sequences. For all datasets except **Fluo-N3DL-TRIC** and **Fluo-N3DL-TRIF**, *n* was set to 3 in order to not exclude true migrations. For **Fluo-N3DL-TRIC** and **Fluo-N3DL-TRIF** however, *n* was set to 1 to decrease the run time and memory requirements.

Global track linking. Our track linking algorithm is global in the sense that it considers all images of the sequence simultaneously when tracks are generated. The algorithm optimizes a probabilistically motivated scoring function by iteratively adding cell tracks to the image sequence. This is done by constructing a state space diagram representing all possible ways in which an additional cell track can be added to the image sequence [3]. The arcs of the state space diagram have scores associated with them, so that we can find the track that increases the scoring function the most by finding the highest scoring path through the state space diagram. Given that the state space diagram is a trellis graph, the highest scoring path can be found by solving a shortest path problem using the Viterbi algorithm. To prevent incorrectly created tracks from blocking the creation of correct tracks in subsequent iterations, the preexisting tracks can be edited using so called swap operations, when new tracks are created [3]. The scoring function is a sum of logarithmic probabilities of tracking events, which describe migration, mitosis, appearance, disappearance, and the number of cells in each detection. The probabilities of migration events are computed as described in [3], using a Brownian motion model where the location of a cell in one image is assumed to follow a Gaussian distribution with covariance matrix $\sigma_V^2\Sigma$, centered around the location of the cell in the previous image. We used the same Σ as in *Bandpass* Segmentation, except for **Fluo-N3DH-CE**, where we used $\Sigma = \mathrm{diag}(1, 1, 1/(4r)^2)$, as there was significantly less motion in the z-dimension than in the other dimensions. The values for σ_V were set manually for all datasets. In **Fluo-N3DH-CE**, we used different values for σ_V for the first and the last image of the sequence, and used linear interpolation to compute a different value for each image in between. The prior probabilities that the segmented regions contain zero, one, or more than one cell are denoted p_0, p_1, and p_2. The probability that a cell undergoes mitosis in a region is denoted p_S, and the probability that a cell appears or disappears randomly in a region is denoted p_A. For all datasets except **BF-C2DL-HSC** and **BF-C2DL-MuSC**, these probabilities were set manually. For **BF-C2DL-HSC** and **BF-C2DL-MuSC**, logistic regression classifiers were used to compute p_0, p_1, p_2, and p_S. The logistic regression classifiers use intensity and shape-features of the segmented regions and were trained on manually corrected tracking results on the training sequences. Details about the classifiers and the features can be found in [3]. Once the Viterbi algorithm has finalized generating tracks, the segmented regions with multiple cells are separated using *k*-means clustering of the pixel coordinates as described in [3], so that each cell gets a region of its own. Then the track linking is updated, to account for the new centroid positions of the individual cells, by solving an assignment problem that maximizes the scoring function. For the image sequences which have *FP*="yes", we included segmented regions in the results even if the track linking algorithm found them to be false positives. This was done to maximize the **TRA** and **SEG** measures, which penalize false negatives more than false positives.

Global track linking with detection preprocessing. The cells in **Fluo-N3DL-DRO**, **Fluo-N3DL-TRIC**, and **Fluo-N3DL-TRIF** form tissues which deform as the embryos develop. Because of this, the nuclei follow smooth and predictable trajectories. The track linking procedure described in the previous section assumes that the nuclei follow Brownian motion, and can therefore not take the velocities of the nuclei into account when it predicts where they are going to be in the next frame. To enable tracking of fast moving nuclei, we therefore used the algorithm described in [6]. It first runs a GM-PHD filter on the centroids of the nuclei and then links the Gaussian components (which include velocity states) of the computed hypothesis densities into tracks using the track linking algorithm in [3]. For the GM-PHD we used the directed linear motion model previously used by us to track simulated microtubules in [6], with a scale factor q = 0.5 for the process noise, and an observation noise covariance of $R = 4\Sigma$. For the remaining parameters described in [6] we used the following values: $p_S = 0.9999$, $p_D = 0.999$, $\kappa = 4\text{E}{-6}$, $w_{\min} = 0.001$, $KLD_{\min} = 1$, $J_{\max} = 10000$, and $\sigma_V = 2$. To get estimates of the probabilities that Gaussian components in the GM-PHD correspond to cells, we trained a multinomial logistic regression classifier on the weights of Gaussian components that were updated using detections that overlapped with ground truth regions in the training dataset. The component with the highest weight in each detection was assumed to be a cell and the others were assumed to not be cells. This was the best we could do with the incomplete ground truth that we were given. The same classifier was used for **Fluo-N3DL-DRO**, **Fluo-N3DL-TRIC**, and **Fluo-N3DL-TRIF**. We first tracked all the nuclei using the algorithm described above, and then we selected the tracks that overlapped with one of the manually marked nuclei in the first image. For manually marked nuclei, which had no overlapping tracks, we selected the closest non-overlapping track. For **Fluo-N3DL-DRO** we expected all of the selected tracks to reach the end of the video. We therefore extended broken selected tracks by linking them to fragments of unselected tracks. This was done by propagating the state of the broken track to the frame after the break, using the directed linear motion model, and then linking it to the closest unselected track in that frame. This was done iteratively until all selected tracks reached the end of the image sequence.

KTH-SE: POST-PROCESSING

To remove segmentation errors that were due to over-segmentation in the watershed transforms, we iteratively merged region fragments without cells into adjacent regions with cells, after the tracking had been completed. We took this idea one step further in the dataset **Fluo-C3DH-A549** and the image sequence **FluoC2DL-MSC-01**, where we also merged region fragments without cells in image t, into cells with which they overlapped in one of the images t − 3, t − 2, t − 1, t + 1, t + 2, and t + 3. The fragments were merged with the regions of the cells in image t, provided that the cells were present in that image. The merging was done iteratively until no more fragments could be merged.

REFERENCES

1. Maška M, Ulman V, Svoboda D, Matula P, Matula P, Ederra C, Urbiola A, Espana T, Venkatesan S, Balak DMW, Karas P, Bolcková T, Štreitová M, Carthel C, Coraluppi S, Harder N, Rohr K, Magnusson KEG, Jaldén J, Blau HM, Dzyubachyk O, Křížek P, Hagen GM, Pastor-Escuredo D, Jimenez-Carretero D, Ledesma-Carbayo MJ, Munoz-Barrutia A, Meijering E, Kozubek M, Ortiz-de-Solórzano C. A benchmark for comparison of cell tracking algorithms. *Bioinformatics* 30, 1609-1617 (2014).
2. Wu K, Gauthier D, Levine MD. Live cell image segmentation. *IEEE Transactions on Biomedical Engineering* **42**, 1-12 (1995).
3. Magnusson KEG, Jaldén J, Gilbert PM, Blau HM. Global linking of cell tracks using the Viterbi algorithm. *IEEE Transactions on Medical Imaging* **34**, 1-19 (2015).

4. Sertel O, Dogdas B, Chiu CS, Gurcan MN. Microscopic image analysis for quantitative characterization of muscle fiber type composition. *Computerized Medical Imaging and Graphics* **35**, 616-628 (2011).
5. Mishchenko Y. A fast algorithm for computation of discrete Euclidean distance transform in three or more dimensions on vector processing architectures. *Signal, Image and Video Processing* **9**, 1-9 (2012).
6. Magnusson KEG, Jaldén J. Tracking of non-Brownian particles using the Viterbi algorithm. In *Proceedings of the 12th IEEE International Symposium on Biomedical Imaging*, 380-384 (2015).
7. Vo BN, Ma WK. The Gaussian mixture probability hypothesis density filter. *IEEE Transactions on Signal Processing* **54**, 4091-4104 (2006).