THE BAXTER ALGORITHMS

Klas E. G. Magnusson

RaySearch Laboratories, Eugeniavägen 18, SE-113 68 Stockholm, Sweden

1. INTRODUCTION

The software uses a tracking by detection framework with 3 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, it also uses a detection preprocessing algorithm based on GM-PHD filtering, which allows it to use dynamic motion models in the track linking step. The software is called the Baxter Algorithms and can be downloaded from https://github.com/klasma/BaxterAlgorithms. The software is written in Matlab and C++. Earlier versions of the software have been used in the ISBI Cell Tracking Challenges of 2013, 2014, 2015, 2019, and 2020. For the Cell Tracking Challenge of 2021, I submitted results for the primary track and updated results for the datasets Fluo-C2DL-Huh7 and Fluo-N3DL-TRIF. In Fluo-N3DL-TRIF, I switched from the setting that had been optimized using coordinate ascent to the settings from Fluo-N3DL-TRIC, as it seemed like there was some overfitting to the training data. The code and the settings for the other secondary track datasets are all unchanged and therefore I found it unnecessary to submit the results for those datasets again. The algorithm descriptions given in the text apply to both the primary and the secondary track, but the settings presented in the text apply only to the secondary track. The training procedure and the minor algorithm changes used in the primary track of the Cell Tracking Challenge of 2021 are described in Section 5.

The Baxter Algorithms were developed in a collaboration between the department of signal processing at KTH and the Blau lab at Stanford University, while I did my PhD at KTH. In the Cell Tracking Challenges of 2013, 2014, and 2015, I participated in a team together with my advisers Joakim Jaldén and Helen Blau. The work that was done for the cell tacking challenges of 2019, 2020, and 2021 was however done at RaySearch Laboratories. It should however be noted that I did most of the work on the BF-C2DL-HSC and BF-C2DL-MuSC during my PhD, as I had access to those datasets through our collaboration with the Blau lab.

Parallel processing of multiple frames is used in the segmentation step. The number of frames that are processed in parallel is decided based on the amount of RAM available but is never made larger than the number of processor cores.

2. SEGMENTATION

The software uses 4 different segmentation algorithms, described in sections 2.2, 2.3, 2.4, and 2.5 to generate binary segmentation masks, which it then post-processes to extract cell regions, as described in Section 2.6. When possible, a search algorithm, described in Section 2.7, was used to optimize the segmentation parameters, but in some cases better results were achieved by optimizing the parameters manually in a graphical user interface. Automatically optimized parameter values are underlined in the text and the tables. In the descriptions below, all images have been converted to 64-bit double images with a saturation intensity of 1.

To reduce the memory usage, each z-stack of the Fluo-N3DL-TRIF is broken into 16 sub-volumes which are segmented separately. The sub-volumes overlap by 100 voxels in each direction and after they have been segmented, the segmented blobs from all sub-volumes were combined into a segmentation result for the entire z-stack. Without separating the z-stacks into sub-volumes, 64 GB of RAM is not enough to perform segmentation.

2.1. Background subtraction

To get rid of background features in Fluo-C2DL-MSC, BF-C2DL-HSC, BF-C2DL-MuSC, and PhC-C2DH-U373, background images were subtracted before the respective segmentation algorithms were applied. In the fluorescence dataset 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 the transmission microscopy datasets 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, separate background images were computed for different time intervals. To deal with gradual changes of the image, a linear combination of background images was fitted to each image and then subtracted. Background images were computed for the frame intervals 1-321, 322-421, 422-875, 876-908, and 909-1376.

2.2. Bandpass segmentation

The bandpass filtering based segmentation algorithm presented in [1] was used to segment all of the Fluo datasets and PhC-C2DL-PSC. The filtering was performed by convolving the original image I with two different Gaussian filters G_{S} and G_{B}, with covariance matrices $\mathbf{\Sigma}_{\mathrm{S}} = \sigma_{\mathrm{S}}^2\mathbf{\Sigma}$ and $\mathbf{\Sigma}_{\mathrm{B}} = \sigma_{\mathrm{B}}^2\mathbf{\Sigma}$. In 2 dimensions, $\mathbf{\Sigma}$ is the 2×2 identity matrix, and in 3 dimensions $\mathbf{\Sigma} = \mathrm{diag}(1, 1, 1/r^2)$, 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 $I_{\mathrm{BP}} = I_{\mathrm{S}} - \alpha I_{\mathrm{B}}$, 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, some of the datasets were preprocessed using intensity clipping, where all pixel values above I_{max} are set to I_{max}.

In Fluo-N3DH-CE, the noise properties are different in the different image dimensions, and therefore a $5 \times 1 \times 3$ ($dx \times dy \times dz$) median filter was used to reduce the noise before the bandpass filter was applied. In the Fluo-C2DL-MSC datset, a tophat filter with a radius of 300 pixels was applied to remove background intensity. In Fluo-N3DL-TRIC and Fluo-N3DL-TRIF 2D tophat filters were applied to remove the high background in regions with densely packed nuclei. The tophat filters had radii of 15 and 18 pixels respectively. In Fluo-N3DH-CE and PhC-C2DL-PSC, different values were used

for σ_S and σ_B for the first and the last image of the sequence, and used linear interpolation was used to compute different values for all images in between. In the Fluo-N3DH-CE dataset, the same linear functions were used for both sequences, but the sequences have different lengths and therefore they have different settings in the last image. The parameter values used for the different datasets are given in Table 1.

Table 1. Bandpass filtering parameters.

Dataset	seq.	σ_S	σ_B	α	τ	I_{max}
Fluo-C2DL-Huh7	1,2	4.65	5.66	0.481	0.0103	0.266
Fluo-C2DL-MSC	1	3.04	14.8	0.582	0.031	0.25
	2	2.58	19.2	0.532	0.0398	
Fluo-C3DH-A549	1	1.5	15.6	1.00	0.0156	1
	2		18.9	1.02	0.00873	
Fluo-C3DH-A549-SIM	1	9.41	59.4	1.29	0.0233	0.185
	2	3.09	46.7	1.38	0.0229	0.182
Fluo-C3DH-H157	1	5	50	0.8	0.01	1
	2				0.019	
Fluo-C3DL-MDA231	1,2	1	NA	0	0.125	1
Fluo-N2DH-GOWT1	1	1.8	22	1	0.005	0.2
	2				0.00157	
Fluo-N2DH-SIM+	1	1.25	13.6	1.01	0.00408	0.440
	2	2.53	23.4	1	9.91E-4	0.371
Fluo-N2DL-HeLa	1,2	2.59	4.6	0.931	9.74E-5	1
Fluo-N3DH-CE	1	15(8)	20(12)	1	0.01	1
	2	15(9.85)	20(14.1)			
Fluo-N3DH-CHO	1,2	3	40	1.5	0.005	1
Fluo-N3DH-SIM+	1	1.27	4.22	1	0.00234	0.6
	2	1.78	12.4		0.00139	0.633
Fluo-N3DL-DRO	1,2	1.5	8	1	0.01	0.3
Fluo-N3DL-TRIC	1,2	0.495	10.6	0.8	0.00234	1
Fluo-N3DL-TRIF	1,2	0.495	10.6	0.8	0.00234	1
PhC-C2DL-PSC	1,2	2.32(0.740)	4.61(2.12)	1	0.00226	1

2.3. 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 the BF-C2DL-MuSC dataset, we computed the variance in a square $w_{\text{var}} \times w_{\text{var}} = 7 \times 7$ pixel region as in [3]. In PhC-C2DH-U373 we weighted the surrounding pixels using a Gaussian kernel G with covariance matrix $\mathbf{\Sigma}_{\text{var}} = \sigma_{\text{var}}^2 \mathbf{I_2}$, where $\mathbf{I_2}$ is the 2×2 identity matrix. We computed pixel n of the weighted local variance image V as

$$[V]_n = \frac{[G * I^2]_n}{[G * \mathbf{1}]_n} - \left(\frac{[G * I]_n}{[G * \mathbf{1}]_n} \right)^2, \quad (1)$$

where I^2 is an image with the squared pixel intensities, $\mathbf{1}$ is an image with all ones, and $[\cdot]_n$ denotes pixel n of an image. The obtained variance image was thresholded using a threshold τ_{var} to give a binary segmentation mask. On PhC-C2DH-U373 we used the parameter values $\sigma_{\text{var}} = \underline{1.88}$, $\tau_{\text{var}} = \underline{5.57\text{E-}5}$. On BF-C2DL-MuSC we used $\tau_{\text{var}} = \underline{1.44}$.

Given that the variance and the weighted variance are computed in large regions, the intensity variations on the boundaries of the cells will contribute to the computed texture of pixels outside of the cell. To avoid getting regions that are too large, we applied morphological erosion with a square $w_e \times w_e = 7 \times 7$ pixel structuring element in BF-C2DL-MuSC and a round structuring element with a radius of $\underline{8.31}$ pixels in PhC-C2DH-U373.

2.4. Template matching segmentation

To segment the BF-C2DL-HSC dataset, I used a segmentation algorithm based on template matching, which is described in [4]. 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 $\tau_{\text{temp}} = 0.45$ 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 on BF-C2DL-MuSC was used as a secondary segmentation algorithm, to deal with cells that do not fit the template. The parameters for the variance segmentation were set to $w_{\text{var}} = 5$, $w_e = 7$, and $\tau_{\text{var}} = 2$. The secondary algorithm is used to create a binary segmentation mask, the pixels segmented by template matching are removed, and then morphological opening with a disk 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.

2.5. Ridge segmentation

To segment cells in DIC-C2DH-HeLa, we developed an algorithm inspired by the algorithm used to segment muscle fibers in [5]. We first apply a ridge detection filter similar to the filter described in [5], to highlight the boundaries between the cells. The ridge detection is 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 6 resulting images. The ridge image $\nu_0(\sigma)$ at the scale σ is then computed from the eigenvalues λ_1 and λ_2, where $\lambda_1 \leq \lambda_2$, of the corresponding Hessians as

$$\nu_0(\sigma) = \begin{cases} 0 & \text{if } \lambda_1 > 0 \\ e^{-R_B/\gamma^2}(1 - e^{-S/\beta^2}) & \text{otherwise} \end{cases} \quad (2)$$

where $R_B = |\lambda_2| \,/\, |\lambda_1|$ and $S = \lambda_1^2 + \lambda_2^2$. We used $\gamma = 1$ and $\beta = 10$. The final ridge image was obtained by taking the pixelwise maximum of $\nu_0(\sigma)$ 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 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 represents the sample variance in a 9×9 pixel 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 end points by straight lines of pixels. 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.

2.6. Post processing

Table 2. Parameters for segmentation post processing.

Dataset	seq.	A_{min}	S_{min}	watersheds	σ_{W}	H_{min}
BF-C2DL-HSC	1,2	40	0	none	NA	NA
Fluo-C2DL-Huh7	1,2	100	5	bandpass(shape)	0	0.00831(5.74)
BF-C2DL-MuSC	1,2	96	0	variance(shape)	5.87(0.162)	3.13(3)
DIC-C2DH-HeLa	1,2	0	0	shape	0	10
Fluo-C2DL-MSC	1 2	0	100 150	intensity(shape) shape(shape)	10 20(10)	10(8) 0(10)
Fluo-C3DH-A549	1,2	100	0	none	NA	NA
Fluo-C3DH-A549-SIM	1,2	100	0	none	NA	NA
Fluo-C3DH-H157	1,2	5000	100	shape	0	10
Fluo-C3DL-MDA231	1,2	100	0	intensity	0	25
Fluo-N2DH-GOWT1	1 2	0	30 5	shape	0	5
Fluo-N2DH-SIM+	1 2	0	10 25	shape	0 7	5 0.1
Fluo-N2DL-HeLa	1,2	0	0.25	shape	0	1
Fluo-N3DH-CE	1,2	0	0	bandpass	0	0.001
Fluo-N3DH-CHO	1,2	0	100	xy-shape	0	5
Fluo-N3DH-SIM+	1 2	500 0	100 500	shape	0	5
Fluo-N3DL-DRO	1,2	100	0	bandpass	0	0.02
Fluo-N3DL-TRIC	1,2	25	0	bandpass(shape)	2.38(0.737)	0.000254(0.427)
Fluo-N3DL-TRIF	1,2	0	5	bandpass(shape)	2.38(0.737)	0.000254(0.427)
PhC-C2DH-U373	1,2	1200	0	none	NA	NA
PhC-C2DL-PSC	1,2	20	0	bandpass(shape)	0(0)	0.0293(0.75)

To break regions with multiple cells into individual cell regions, we applied a seeded watershed transform to the image intensity, the bandpass filtered image, the local variance, or the distance transform of the binary segmentation mask. The distance transform is computed so that the transform value is the Euclidean distance to the closest background pixel. For z-stacks, where the voxel height was different from the voxel width, we used the anisotropic distance transform [6], 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 9 virtual z-planes for Fluo-N3DH-CE and 2 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 with other parameters, 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 Fluo-N3DH-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 PhC-C2DH-U373 tends to give too large regions, due to the large kernel used to compute the variance. To overcome this problem, we applied morphological erosion with a circular structuring element with a radius of 8.31 pixels.

In DIC-C2DH-HeLa there was quiet a lot of over-segmentation, but in many cases over-segmented regions were correctly segmented in adjacent images. We therefore tired 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.

2.7. 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 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 the SEG-measure as utility function for the optimization and ran it for 25 iterations. For Fluo-C2DL-MSC, Fluo-N2DH-SIM+, and 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-measure.

3. TRACK LINKING

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 [7], 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 [8]. Once we had preprocessed the locations, we linked them using the track linking algorithm 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.

3.1. Global track linking

Our track linking algorithm is global in the sense that it considers all images of the image 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 which 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 $\sigma_{\mathrm{V}}^2\mathbf{\Sigma}$, centered around the location of the cell in the previous image. We used the same $\mathbf{\Sigma}$ as in Section 2.2, except for Fluo-N3DH-CE, where we used $\mathbf{\Sigma} = \mathrm{diag}(1, 1, 1/(4r^2))$, 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, and are given in Table 3. 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 segmented regions contain 0, 1, or more than 1 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 probabilities 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, and are given in Table 3. 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 is done 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 which maximizes the scoring function of the track linking problem. For the image sequences which have "yes" in the last column of Table 3, 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 a lot more than false positives.

Table 3. Track linking parameters.

Dataset	seq.	σ_{V}	p_0	p_1	p_2	p_{S}	p_{A}	FP
BF-C2DL-HSC	1,2	2	NA	NA	NA	NA	0	no
BF-C2DL-MuSC	1,2	11.1	NA	NA	NA	NA	0	no
DIC-C2DH-HeLa	1,2	25	0.1	0.7	0.2	0.01	0	yes
Fluo-C2DL-MSC	1 2	20 40	0.1	0.8	0.1	0	0	no yes
Fluo-C3DH-A549	1,2	3	0.2	0.8	0	0	0	no
Fluo-C3DH-A549-SIM	1,2	3	0.2	0.8	0	0	0	no
Fluo-C3DH-H157	1 2	20 40	0.07	0.83	0.1	0	0.001	no
Fluo-C3DL-MDA231	1,2	7.5	0.2	0.7	0.1	0	0	yes
Fluo-N2DH-GOWT1	1,2	5	0.1	0.8	0.1	0.01	0.001	no
Fluo-N2DH-SIM+	1-2	10	0.05	0.94	0.01	0.01	1E-5	no
Fluo-N2DL-HeLa	1,2	5	0.1	0.8	0.1	0.01	1E-6	yes
Fluo-N3DH-CE	1 2	20(10) 30(12.3)	0.01	0.99	0.01	0.01	1E-6	no
Fluo-N3DH-CHO	1,2	10	0.1	0.89	0.01	0.01	1E-6	no
Fluo-N3DH-SIM+	1,2	12	0.2	0.7	0.1	0.01	0	no
PhC-C2DH-U373	1-2	12	0.1	0.8	0.1	0	0	no
PhC-C2DL-PSC	1,2	2	0.05	0.9	0.05	0.01	0	no

3.2. Global track linking with detection preprocessing

The cells in Fluo-N3DL-DRO, Fluo-N3DL-TRIC, 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 3.1 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 [7]. That algorithm 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 that we used to track simulated microtubules in [7], with a scale factor $q = 0.5$ for the process noise, and an observation noise covariance of $R = 2^2\mathbf{\Sigma}$. For the remaining parameters described in [7] we used the following values: $p_{\mathrm{S}} = 0.9999$, $p_{\mathrm{D}} = 0.999$, $\kappa = 4\mathrm{E}{-6}$, $w_{\min} = 0.001$, $KLD_{\min} = 1$, $J_{\max} = 10000$, 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* of the nuclei using the algorithm described above, and then we selected the tracks which overlapped with one of the manually marked nuclei in the first image. For manually marked cells 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.

4. POST TRACKING SEGMENTATION

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 Fluo-C2DL-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 to 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.

5. CTC 2021 PRIMARY TRACK

Results for the primary track of the ISBI 2021 Cell Tracking Challenge were generated using bandpass segmentation and the global track linking algorithm without detection preprocessing with a GM-PHD. The segmentation parameters were optimized using coordinate ascent and the track linking parameters were the same for all datasets. Both the segmentation and tracking performance would have benefited from optimization of the track linking parameters, but that kind of optimization had not yet been implemented in the software in 2021. The segmentation performance would probably have been slightly higher if raw segmentation results had been submitted without running tracking.

Bandpass segmentation is only meant to be used on the fluorescence microscopy datasets and PhC-C2DL-PSC. The watershed transforms used for post-processing allow it to segment many of the cells in the other datasets too, but there are other algorithms which can handle those datasets much better.

5.1. Segmentation

For the primary track submission, all datasets were processed using the bandpass segmentation described in Section 2.2. The same pre-processing and post-processing was used on all datasets. All other dataset specific pre-processing and post-processing, such as tophat filtering of the input image and morphological closing of segmented regions was removed. For preprocessing, intensity clipping with an intensity threshold of I_{max} was used. For post-processing, two seeded watershed transforms were used. The first was applied to the bandpass filtered image and the second was applied to the distance transforms of the binary segmentation masks. To avoid over-segmentation, h-minima transforms with h-values of H^b_{min} and H^d_{min} were applied to the bandpass filtered image and the distance transform respectively. Regions with fewer than A_{min} voxels, and regions with a summed voxel intensity below S_{min} were removed to get rid of false positives.

5.2. Intensity normalization

To make it easier to achieve good segmentation results on all datasets with a single set of parameters, the image sequences were rescaled to have a minimum intensity of 0 and a maximum intensity of 1 according to

$$I^{\text{norm}}_n(t) = \frac{I_n(t) - \min_{m,\tau} I_m(\tau)}{\max_{m,\tau} I_m(\tau) - \min_{m,\tau} I_m(\tau)}, \quad (3)$$

where $I_n(t)$ and $I^{\text{norm}}_n(t)$ is voxel n in time point t of the original and normalized image sequences respectively.

5.3. Segmentation parameter optimization

The parameters I_{max}, H^b_{min}, H^d_{min}, A_{min}, and S_{min} were optimized using the coordinate ascent algorithm described in Section 2.7. The coordinate ascent algorithm was modified so that the step length was increased by 40 % if the utility function was not affected by neither increasing nor decreasing the scoring function. Previously, the step length was decreased by 20 % in that case, but that often made parameters get stuck on values where they had no effect on the segmentation.

The scoring function was changed from SEG to $0.9\text{SEG}_{\text{R}} + 0.1\text{DET}$, where SEG_{R} is a relaxed version of the SEG measure, introduced in [4]. The relaxed SEG measure is different from the traditional SEG measure in that it associates the ground truth regions with the computer generated regions which give the highest Jaccard indices, without requiring that more than half of the ground truth region is covered. The relaxed measure is easier to optimize as it allows segmentation improvements to guide the optimization even if the segmented regions do not yet cover more than half of a ground truth region. The term 0.1DET was added to the scoring function to penalize false positives. In the secondary track, where false positives were not penalized in the segmentation parameter optimization, the initial optimization often resulted in a lot of false positives which were removed by setting either A_{min} or S_{min} manually. In the primary track, segmented regions were included in the results even if the track linking algorithm found them to be false positives, as there were very few real false positive detections given that DET was included in the utility function in the segmentation optimization.

The initial segmentation settings can be found in Table 4. All settings were optimized using 25 iterations of coordinate ascent. To reduce the computation time in the ST scenario, 32 time points were used to optimize the settings of image sequences with more than 32 time points. In the GT+ST scenario, the set of time points was instead limited to the time points with GT segmentations plus 32 other time points. All ST-regions associated with a tracking marker also associated with a GT-region were removed in the GT+ST and allGT+allST scenarios. The SEG_{R} measure was then computed as the average of the SEG_{R} measures of the GT and filtered ST ground truths.

Table 4. Initial guess for segmentation optimizations.

σ_S	σ_B	α	τ	I_{max}	H^b_{min}	H^d_{min}	A_{min}	S_{min}
3	10	1	0.01	1	0.01	2	100	5

5.4. Track linking

The same tracking settings were used on all datasets. The settings were adjusted manually by looking at the training data with segmentation settings optimized using the GT-ground truth. The TRA measure was never evaluated, as that would not have been allowed for the scenarios where only the ST-ground truth is available. The tracking settings can be found in Table 5.

Table 5. Track linking parameters used in the primary track.

σ_{V}	p_0	p_1	p_2	p_{S}	p_{A}	FP
15	0.2	0.7	0.1	0.01	0.001	yes

6. CTC 2024 TRACK 3

Results were submitted to track 3 of the linking-only benchmark in the ISBI 2024 Cell Tracking Challenge. Results were also generated for the simulated datasets Fluo-C3DH-A549-SIM, Fluo-N2DH-SIM+, and Fluo-N3DH-SIM+ in track 4, using the same approach.

The linking-only benchmark is a great initiative, as it is now possible to compare linking algorithms that are not coupled to a segmentation algorithm. It would however be even better if the precomputed segmentations to be linked came directly from a real segmentation algorithm, as that would make the problem more realistic. For example, the segmentation from the highest performing segmentation algorithm in the segmentation-only benchmark could be used. In the current linking-only benchmark, there are no false positives and it seems like segmented regions have been artificially removed to make the linking problem harder. In real cell tracking problems, there are usually false positives and cells are almost never missing from the segmentation. In general, the biggest challenge is associated with cells that are segmented jointly in clusters.

6.1. Track linking

The pre-segmented regions were passed directly to the track linking without any segmentation post-processing. The global track linking algorithm without detection preprocessing with a GM-PHD was used. Detection pre-processing with a GM-PHD, which is used for the embryonic datasets, does give the Baxter Algorithms the ability to handle cells that are missing from the segmentation. It could therefore have improved the results, but it was not used because it would increase the complexity and probably decrease the performance for users who wish to apply the algorithms to real-world data. The parameter FP, which converts tracks that the algorithm considered to be false positives into real tracks, was used for all datasets as the performance measures do not penalize false positives.

6.2. Track linking parameter optimization

The parameters σ_{Vxy}, σ_{Vz}, p_0, p_1, p_2, p_{S}, p_{A}, and p_{D} were optimized using 10 iterations of the coordinate ascent algorithm described in Section 2.7. The parameter p_{D} is the probability that a cell disappears. In prior challenges, that value was always set equal to p_{A}, but in the linking-only benchmark, the two parameters were allowed to have different values. Similarly, the standard deviations of the motion model were allowed to be different in the xy-plane and in the z-direction by using $\mathbf{\Sigma} = \mathrm{diag}(\sigma^2_{\mathrm{Vyx}}, \sigma^2_{\mathrm{Vxy}}, \sigma^2_{\mathrm{Vz}}/r^2)$ as the covariance matrix of the Gaussian distribution in the Brownian motion model. The probabilities p_0, p_1, p_2 should normally sum to 1, but they were optimized individually for simplicity, as none of the algorithms rely on the probabilities summing to 1. The parameter p_0 could have been set to 0 as there are no false positives in the linking-only benchmark. It was however kept in the optimization as it can be important for the performance in real-world applications.

In the optimization, σ_{Vxy} and σ_{Vz} had a lower bound of 0.5 voxels. The other parameters had a lower bound of 0 and an upper bound of 1. The linking accuracy measure (LNK) was used as scoring function in the optimization. The initial guess used for all parameter optimizations is given in Table 6 and the optimized parameters are given in Table 7.

Table 6. Initial guess for track linking parameter optimizations.

σ_{Vxy}	σ_{Vz}	p_0	p_1	p_2	p_{S}	p_{A}	p_{D}
15	15	0.2	0.7	0.1	0.01	0.001	0.001

Table 7. Optimized track linking parameters.

Dataset	σ_{Vxy}	σ_{Vz}	p_0	p_1	p_2	p_{S}	p_{A}	p_{D}
BF-C2DL-HSC	2.43	15	0.018	1	0.132	0.0115	0.00133	3.38E-4
BF-C2DL-MuSC	22.4	15	0.018	0.875	0.045	0.0072	0.00159	0.00125
DIC-C2DH-HeLa	18.9	15	0.018	0.875	0.099	0.01	0	0.00139
Fluo-C2DL-MSC	35.1	15	0.046	0.945	0.00396	0.01	0.001	0.001
Fluo-C3DH-A549	15	15	0.2	0.7	0.1	0.01	0.001	0.001
Fluo-C3DH-A549-SIM	15	15	0.2	0.7	0.1	0.01	0.001	0.001
Fluo-C3DH-H157	28.7	15	0.2	0.7	0.1	0.00314	0.001	0.001
Fluo-C3DL-MDA231	18.8	15	0.09	0.875	0.125	0.01	0.001	0.001
Fluo-N2DH-GOWT1	11.3	15	0.13	0.7	0.1	0.01	0.001	0.001
Fluo-N2DH-SIM+	13.6	15	0.24	0.875	0.1	0.012	0.00116	0.001
Fluo-N2DL-HeLa	11.3	15	0.2	0.84	0	0.0113	0.00202	8.98E-4
Fluo-N3DH-CE	15.1	15	0.09	0.562	0.156	0.0105	9.36E-4	0.001
Fluo-N3DH-CHO	11.3	15	0.2	0.7	0.1	0.01	0.001	0.001
Fluo-N3DH-SIM+	11.3	15	0.09	0.7	0.1	0.01	7.5E-4	0.001
PhC-C2DH-U373	15	15	0.2	0.7	0.1	0.01	0.001	0.001
PhC-C2DL-PSC	2.43	15	0.018	1	0	0.0337	6.41E-4	0.00195

7. REFERENCES

[1] M. Maška, V. Ulman, D. Svoboda, P. Matula, P. Matula, C. Ederra, A. Urbiola, T. España, S. Venkatesan, D. M. W. Balak, P. Karas, T. Bolcková, M. Štreitová, C. Carthel, S. Coraluppi, N. Harder, K. Rohr, K. E. G. Magnusson, J. Jaldén, H. M. Blau, O. Dzyubachyk, P. Křížek, G. M. Hagen, D. Pastor-Escuredo, D. Jimenez-Carretero, M. J. Ledesma-Carbayo, A. Muñoz-Barrutia, E. Meijering, M. Kozubek, and C. Ortiz-de Solorzano, "A benchmark for comparison of cell tracking algorithms," *Bioinformatics*, vol. 30, no. 11, pp. 1609–1617, 2014.

[2] K. Wu, D. Gauthier, and M. D. Levine, "Live cell image segmentation," *IEEE Transactions on Biomedical Engineering*, vol. 42, no. 1, pp. 1–12, 1995.

[3] K. E. G. Magnusson, J. Jaldén, P. M. Gilbert, and H. M. Blau, "Global linking of cell tracks using the Viterbi algorithm," *IEEE Transactions on Medical Imaging*, vol. 34, no. 4, pp. 911–929, 2015.

[4] K. E. G. Magnusson, *Segmentation and tracking of cells and particles in time-lapse microscopy*, Ph.D. thesis, KTH Royal Institute of Technology, 2016.

[5] O. Sertel, B. Dogdas, C. S. Chiu, and M. N. Gurcan, "Microscopic image analysis for quantitative characterization of muscle fiber type composition," *Computerized Medical Imaging and Graphics*, vol. 35, no. 7, pp. 616–628, 2011.

[6] Y. Mishchenko, "A fast algorithm for computation of discrete Euclidean distance transform in three or more dimensions on vector processing architectures," *Signal, Image and Video Processing*, vol. 9, no. 1, pp. 19–27, 2015.

[7] K. E. G. Magnusson and J. Jaldén, "Tracking of non-Brownian particles using the Viterbi algorithm," in *Proceedings of the IEEE International Symposium on Biomedical Imaging (ISBI)*, 2015, pp. 380–384.

[8] B.-N. Vo and W.-K. Ma, "The Gaussian mixture probability hypothesis density filter," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, 2006.