SPF-CellTracker: Tracking Multiple Cells with Strongly-Correlated Moves Using a Spatial Particle Filter

Osamu Hirose, Shotaro Kawaguchi, Terumasa Tokunaga, Yu Toyoshima, Takayuki Teramoto, Sayuri Kuge, Takeshi Ishihara, Yuichi Iino, and Ryo Yoshida

Abstract—Tracking many cells in time-lapse 3D image sequences is an important challenging task of bioimage informatics. Motivated by a study of brain-wide 4D imaging of neural activity in *C. elegans*, we present a new method of multi-cell tracking. Data types to which the method is applicable are characterized as follows: (i) cells are imaged as globular-like objects, (ii) it is difficult to distinguish cells on the basis of shape and size only, (iii) the number of imaged cells in the several-hundred range, (iv) movements of nearly-located cells are strongly correlated, and (v) cells do not divide. We developed a tracking software suite that we call SPF-CellTracker. Incorporating dependency on the cells' movements into the prediction model is the key for reducing the tracking errors: the cell switching and the coalescence of the tracked positions. We model the target cells' correlated movements as a Markov random field and we also derive a fast computation algorithm, which we call spatial particle filter. With the live-imaging data of the nuclei of *C. elegans* neurons in which approximately 120 nuclei of neurons were imaged, the proposed method demonstrated improved accuracy compared to the standard particle filter and the method developed by Tokunaga et al. (2014).

Index Terms—Particle filter, Markov random field, automatic cell tracking, 4D live-cell imaging data

1 Introduction

Developments in imaging technologies such as confocal microscopes and fluorescent proteins have increased the demand for computational techniques to process live-cell imaging data, including automatic cell tracking. Many algorithms for cell-tracking tasks have been developed owing to their variety in shape, motion and density [see [1], [2], and [3] for comprehensive surveys]. In the review by Maška et al., they classified these tracking algorithms into two categories, namely, *detection and linking* and *contour evolution* according to their algorithm designs. Methods in the first category track cells by using a two-step procedure: (1) detection of cells in all frames of the video and (2) finding the corresponding links of the cells in successive frames [4], [5], [6]. In the second category, segmentation and tracking of cells are simultaneously executed by predicting the cell positions and evolving the contours of the cells in the previous frame to those in the current frame [8], [9]. In addition to the categories, there is another category called *particle filter*. Tracking algorithms in this category track cells by evolving the probabilistic distribution of the cell positions based on the Bayesian recursive formula, instead of estimating the cell positions only [10]. All the three categories have their own advantages. A major advantage of methods in the first category is the capability to track new cells entering the field of view since the cell positions are determined before the tracking. In the second category, robustness for morphological change of living cells is a main advantage. An advantage of particle filters is the capability to handle nonlinear and non-Gaussian motion of cells.

Our motivated datasets were 4D live-cell imaging data that capture the nuclei of *C. elegans*. The locations and Ca^{2+} activity levels of the neurons can be visualized by incorporating various fluorescent proteins such as mCherry, CFP, and YFP [11]. In order to study the information processing in *C. elegans*, their neural activities must be measured accurately. For accurate measurements, it is essential to track multiple neurons since the neurons of a live nematode move frame by frame according to the movements of the nematode itself. In this study, we aim to track more than a hundred neurons in brain-wide 4D live-cell imaging data of *C. elegans*. Data types to which the method is applicable are

- *O. Hirose is with the Institute of Science and Engineering, Kanazawa University, Kakuma, Kanazawa, Ishikawa 920-1192, Japan. E-mail: hirose@se.kanazawa-u.ac.jp.*
- *S. Kawaguchi is with the Graduate School of Natural Science and Technology, Kanazawa University, Kakuma, Kanazawa, Ishikawa 920-1192, Japan. E-mail: zeb79637@gmail.com.*
- *T. Tokunaga is with the Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology 680-4 Kawazu, Iizuka, Fukuoka 820-8502, Japan. E-mail: tokunaga@ces.kyutech.ac.jp.*
- *Y. Toyoshima and Y. Iino are with the University of Tokyo, Building 3, 2-11-16 Yayoi, Bunkyo-ku, Tokyo 113-0032, Japan. E-mail: {ytoyo, iino}@bs.s.u-tokyo.ac.jp.*
- *T. Teramoto, S. Kuge, and T. Ishihara are with Kyushu University, 6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan. E-mail: {teramoto.takayuki.873, kuge.sayuri.449}@m.kyushu-u.ac.jp, takeiscb@gmail.com.*
- *R. Yoshida is with the Institute of Statistical Mathematics, Research Organization of Information and Systems, 10-3 Midori-cho, Tachikawa, Tokyo 190-8562, Japan. E-mail: yoshidar@ism.ac.jp.*

Manuscript received 21 Sept. 2015; accepted 25 Oct. 2017. Date of publication 11 Dec. 2017; date of current version 6 Dec. 2018.
(Corresponding author: Osamu Hirose.)
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCBB.2017.2782255

characterized as follows: (i) cells are imaged as globular-like objects, (ii) hundreds of cells are present, (iii) movements of nearly located cells are strongly correlated with one another, and (iv) cells do not divide. We employ the particle filter as a basis of our tracking algorithm focusing on its scalability to large datasets.

Many tracking methods have been developed on the basis of the particle filter in the field of digital image processing. These methods have succeeded in tracking various objects such as human bodies [12], [13], human faces [14], [15], cars [16], [17], and so forth. Applying methods directly to our data often causes problems such as cell switching, i.e., mistaking a cell of interest for one of the other cells. This is mainly due to the following reasons; (1) the neuronal nuclei to be imaged are usually ellipsoidal shapes and it is difficult to distinguish them from visual information only, (2) the neurons are severely jammed in some areas of the image, and (3) the movements of a nematode itself are irregular and sometimes sudden and rapid.

To improve the accuracy of multi-object tracking based on particle filters, Khan *et al.* proposed a particle filter combined with a Markov random field (MRF) that models the dependency on targets' movements [18]. Smal et al. proposed a method specialized for tracking the multiple edges of microtubules during polymerization based on an MRF to avoid collisions of multiple edges [10]. The MRF-based tracking framework is promising but needs to be designed according to the characteristics of the dataset. The key feature of our data is that many cells move rapidly, but the nearly located cells' movements are strongly correlated since the change in the cell positions occurs owing to the change in body posture of a nematode. Properly modeling such covariation of cells might be the key to reducing the tracking errors.

More recently, we proposed a novel multi-cell tracking method which is based on an MRF model and an optimization technique in order to address the task of tracking hundreds of cells. The proposed method is a novel variant of our previous method [19]. We employ a more sophisticated MRF and the optimization technique is replaced by a sampling-based algorithm motivated by the particle filter algorithm. The use of the particle-based tracking combined with the MRF offers two advantages; (i) a great reduction in computational time and (ii) a significant improvement of tracking accuracy due to the incorporation of more sophisticated spatial information into the MRF. Through applications to synthetic data and 4D images of the neuronal nuclei of *C. elegans*, we demonstrate that the proposed method indeed outperforms our optimization-based method in terms of tracking performance.

2 Methods

Here, we review the existing tracking methods with the particle filter. We introduce the particle filter for single object-tracking and its variants for the simultaneous tracking of multiple objects. We then describe our tracking method.

2.1 Existing Tracking Methods with the Particle Filter

2.1.1 Particle Filter

We begin with a brief introduction to the particle filter (PF), commonly used as a standard object-tracking method. Let y_t be the observed image data at time t, and let x_t denote the *state* of the target at time t. The state includes the target's information such as the location, velocity, volume and shape. The unknown state variables are usually estimated as the *filtering distribution* $p(x_t|y_{1:t})$, where $y_{1:t} = \{y_1, \ldots, y_t\}$. With the dynamics of the state $p(x_t|x_{t-1})$ and the likelihood $p(y_t|x_t)$, the filtering distribution can be computed according to the following Bayesian recursive formula:

$$p(x_t|y_{1:t}) \propto p(y_t|x_t) \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}, \quad (1)$$

If the dynamics $p(x_t|x_{t-1})$ and the likelihood $p(y_t|x_t)$ are linear and Gaussian, these distributions can be exactly calculated by the Kalman filter [20]. These assumptions, however, are too restrictive and unrealistic for object-tracking from video data. Therefore, particle approximation of the distribution is commonly used for dealing with the nonlinearity and non-normality of the distribution. The particle filter approximates the filtering distribution by using a set of point masses, i.e., *particles*. Suppose N is the number of particles, and $\{x_t^{(n)}\}_{n=1}^N$ is the set of particles which approximates the distribution $p(x_t|y_{t-1})$ for *one-step-ahead prediction*. The particle weight $w_t^{(n)}$ is defined as the probability proportional to the likelihood $p(y_t|x_t^{(n)})$ and can be intuitively interpreted as how much a particle captures the target in the current frame. We also suppose $\{w_t^{(n)}\}_{n=1}^N$ is the set of particle weights. The filtering distribution is approximated as follows:

$$p(x_t|y_{1:t}) \approx \sum_{n=1}^{N} w_t^{(n)} \delta(x_t - x_t^{(n)}),$$

where δ denotes the Dirac delta measure with the mass at 0. Under this particle approximation, object-tracking is conducted by the following procedure:

1) *Prediction*: move particles according to the dynamics $p(x_t|x_{t-1})$.
2) *Filtering*: calculate particle weights according to the likelihood $p(y_t|x_t)$.

In Equation (1), the prediction and filtering steps correspond to the calculation of the integral and to the multiplication by the likelihood, respectively. We note that particles are usually resampled according to the probabilities proportional to the particle weights since the resampling procedure often avoids the degeneracy of the algorithm [21], meaning that the resampling procedure stabilizes the tracking quality. We therefore assume that resampling is always conducted in the filtering step, and we refer to a set of resampled particles as a *filter ensemble*, denoted by $\{x_{t|t}^{(n)}\}_{n=1}^N$.

2.1.2 Joint Particle Filter

The standard particle filter has been extended to the tracking of multiple targets. The most basic method is a joint particle filter (JPF), also known as mixture tracking [22]. Suppose $x_{t,k}$ denotes the location of target k at time t. We hereafter denote the set of locations for all targets by x_t. In JPF, the dynamics of the multiple targets is defined as follows:

$$p(x_t|x_{t-1}) = \prod_{k \in V} p(x_{t,k}|x_{t-1,k}),$$

where V is the index set of all targets. That is, the movements of the targets are assumed to be independent of each

other. Therefore, JPF can be computed by an independent run of the standard PF for each target. JPF is a reasonable option for the multi-object tracking if (1) the number of objects to be tracked is moderate, and (2) the visual characteristics of the targets are clearly different. Otherwise, JPF easily fails by tracking an incorrect target located close to the target of interest.

2.1.3 Motion Model with a Markov Random Field

Next, we introduce a framework of MRF-incorporated motion models [18]. In many cases, the movements of a target are dependent on those of other targets. For example, a fish usually changes its direction when it is going to hit another fish. In this approach, JPF is extended to model such dependencies among the targets movements based on the MRF, which is plugged into the dynamics of the JPF as follows:

$$p(x_t|x_{t-1}) \propto \prod_{k\in V} p(x_{t,k}|x_{t-1,k}) \prod_{(k_1,k_2)\in E} \psi(x_{t,k_1}, x_{t,k_2}),$$

where E is the set of edges in the MRF and ψ is the energy function that represents the dependency on movements of targets k_1 and k_2. The filtering distribution $p(x_t|y_t)$ can be obtained by moving particles according to the dynamics $p(x_{t,k}|x_{t-1,k})$ and computing the particle weights that is defined as the product of the likelihood $p(y_t|x_{t,k})$ and the energy function $\psi(x_{t,k}, x_{t,k'})$ for all k' such that $(k, k') \in E$. This algorithm is called the joint MRF particle filter. The tracking performance of the joint MRF particle filter, however, strongly depends on the number of particles N, since the generated particles based on the dynamics $p(x_t|x_{t-1})$ do not include any information about the energy function and tend to be inaccurate. Instead, the filtering distribution $p(x_t|y_t)$ can also be computed by Markov chain Monte Carlo (MCMC) sampling. One issue of the motion model with MRFs is the trade-off between tracking accuracy and computational efficiency. MCMC sampling stabilizes the tracking performance, but its computational cost can be prohibitive due to iterative computation until convergence for each frame. On the contrary, joint MRF-PF is computationally efficient but less accurate. In the next section, we propose a novel sampling method that is a better compromise between the joint MRF particle filter and MCMC sampling.

2.2 Spatial Particle Filter

We propose a novel tracking algorithm, which we call a spatial particle filter (SPF). Our idea to address the sampling issue of MRF-PF is to design a better proposal distribution that shares information about cells' dynamics and dependent movements. How do we construct such a better proposal distribution? The sampling inefficiency in the standard particle filter is due to the difficulty in predicting a cell position based only on temporal information since the movements of a nematode itself are sometimes sudden and rapid. The key idea to overcome the sampling inefficiency is to incorporate the spatial information such as the correlated movements among cells into a proposal distribution. If we already know the position of a cell near the target cell in the current and previous frames and the cells' movements are strongly correlated, the position of the target cell can be efficiently estimated from such correlated movements. We therefore construct a proposal distribution that predicts the target cell position from *spatial* information, unlike the motion models in the standard particle filter. The remaining issue is how we determine positions of *all* cells since all the cell positions are unknown before the tracking. A solution is to determine the position of a cell from a nearly located cell, sequentially. To this end, we transform the joint distribution of the state variables for multiple targets at time t into a recursive formula in the spatial domain, as is similarly done in the temporal domain for the standard PF. Suppose the graph structure of an MRF is restricted to a *tree*. Let $y_{t,k}$ be the region of the kth target image at time t, $u(k)$ the parent node of node k, $1 \rightarrow k$ the path from the root node to node k, and $y_{1\rightarrow k}$ the set of regions of target images corresponding to nodes on the path $1 \rightarrow k$. Then, the conditional distribution $p(x_{t,k}|y_{1\rightarrow k})$ is represented as the following recursive formula:

$$p(x_{t,k}|y_{t,1\rightarrow k}) \propto p(y_{t,k}|x_{t,k}) \int p(x_{t,k}|x_{t,u(k)}) p(x_{t,u(k)}|y_{t,1\rightarrow u(k)}) dx_{t,u(k)}.$$

That is, a target can be tracked from the location of a corresponding parent node by the standard PF-like computation for each path for each frame, meaning that all targets except for the root node can be tracked by a sweep of the MRF tree as shown in Fig. 1c. In this formula, we design $p(x_{t,k}|x_{t,u(k)})$ as a proposal distribution that shares information about the dynamics of the kth target and the energy function between k and $u(k)$. To deal with the nonlinearity and non-normality, we approximate the conditional distribution by a set of particles $\{x_{t,k}^{(n)}\}_{n=1}^N$ and particle weights $\{w_{t,k}^{(n)}\}_{n=1}^N$ as follows:

$$p(x_{t,k}|y_{t,1\rightarrow k}) \approx \sum_{n=1}^{N} w_{t,k}^{(n)} \delta(x_{t,k} - x_{t,k}^{(n)}).$$

We note that the recursive formula is valid only when the MRF is restricted to a tree. We therefore automatically construct an MRF tree from the initial locations of the targets, as described later. We also note that the target corresponding to the root node in the MRF tree cannot be tracked by this recursive formula, and, therefore, we track the root target by the standard PF.

2.2.1 Detection of the Initial Locations and Construction of an MRF Tree

We detect the initial cell locations of by clustering the voxels with local peaks and computing the centers of the clusters. In order to reduce computational costs, we collect the voxels with *strong* fluorescent intensities. Here, a *strong* intensity is defined as a local maximum of the fluorescent intensities in a region narrower than the size of a cell nucleus, assuming that the local peaks of intensities concentrate on the center of a cell nucleus. To cluster the voxels, we use the DP-means algorithm [23], which is an extension of the k-means algorithm, which does not need to specify the number of clusters k. The DP-means algorithm is suitable for our data since (1) it automatically counts the number of clusters, i.e., the number of cells; (2) the computation is considerably fast; and (3) the parameter λ that controls the cluster radii is determined by prior knowledge of the radii of a cell. To construct a graph structure of an

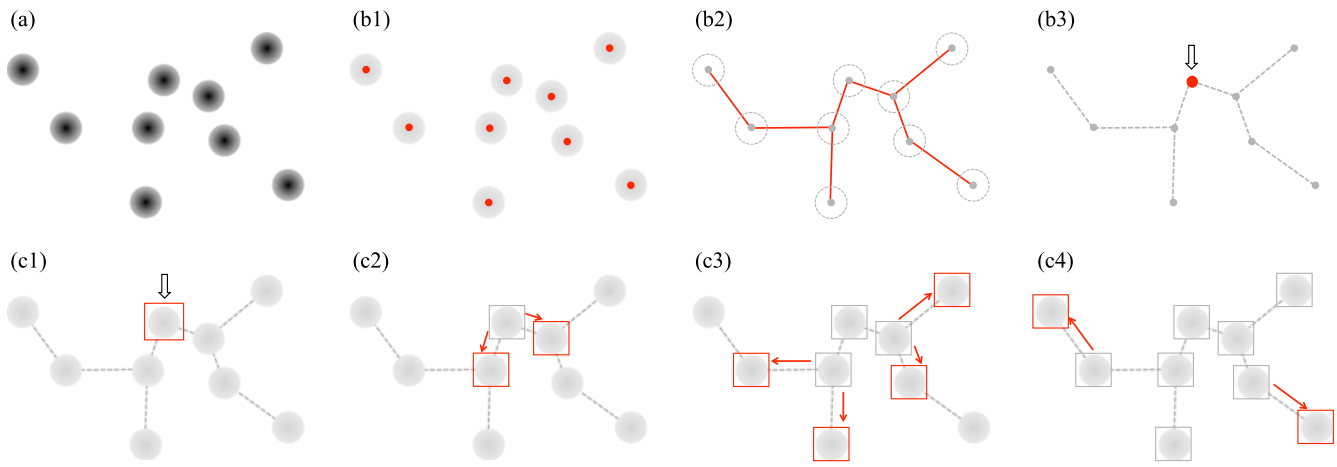


Fig. 1. Outline of the spatial particle filter. (a) An illustrative example of a 4D image dataset for the nuclei of *C. elegans* neurons. (b) Preparation of the algorithm using the initial frame of the dataset: (b1) Detection of the cell centroids by the DP-means algorithm. (b2) Construction of a Markov random field that defines The correlated movements of the cells. Its graph structure is constructed by the minimum spanning tree that spans all the cell centroids in the initial frame. (b3) Determination of the cell corresponding to the root node of the tree. (c) Tracking scheme at time t. (c1) Tracking of the cell corresponding to the root node by the standard particle filter. (c2), (c3), and (c4) Sequential tracking of the remaining cells.

MRF that models the correlated movements among cells, we construct a minimum spanning tree that spans all the detected locations of the cell nuclei by using Kruskal's algorithm [24]. The outline of the spatial particle filter is summarized in Fig. 1.

2.2.2 Particle Generation in the Spatial Domain

We here design a proposal distribution $p(x_{t,k}|x_{t,u(k)})$ that serves as both the dynamics of target k and the energy function between target k and target $u(k)$. Our key ideas for constructing the proposal distribution are to (1) simulate the covariation with neighboring cells, (2) maintain the relative positions of cells in the initial frame, and (3) avoid collisions with neighboring trackers. Let $\{x_{t,k|k}^{(n)}\}_{n=1}^{N}$ be the ensemble set corresponding to the filtering distribution $p(x_{t,k}|y_{1\to k})$, and let $\overline{x}_{t,k}$ be the mean vector averaged over the ensemble set. We hereafter abbreviate the parent node $u(k)$ simply to u if no confusion is likely. One approach to taking the covariation information into account is to generate particle $x_{t,k}^{(n)}$ such that the following approximation holds:

$$x_{t,k}^{(n)} - \overline{x}_{t-1,k} \approx x_{t,u|u}^{(n)} - \overline{x}_{t-1,u},$$

i.e., the movement of target k between two successive frames is roughly the same as the movement of its parent node u. In fact, this is equivalent to preserving the relative position of k for u in the previous frame since transposition of the second and third terms yields

$$x_{t,k}^{(n)} - x_{t,u|u}^{(n)} \approx \overline{x}_{t-1,k} - \overline{x}_{t-1,u}.$$

The covariation of the cells is utilized by this approach, but the relative positions of the cells in the initial frame are not conserved, i.e., the distance between a pair of cells can be infinite in the long run. We therefore consider another constraint on particle $x_{t,k}^{(n)}$, such that the relative positions of the cells in the initial frame are roughly conserved.

$$x_{t,k}^{(n)} - x_{t,u|u}^{(n)} \approx \overline{x}_{1,k} - \overline{x}_{1,u}.$$

We note that the relative positions to be conserved do not have to be those in the initial frame and can be arbitrarily chosen from those in all frames if the tracking is offline. Therefore, if we find the frame when the movement of a nematode itself is slower and more stable than that in the initial frame, the frame is a better choice for defining the relative positions of the cells. Let $\eta_t^{k_1,k_2} := \overline{x}_{t,k_1} - \overline{x}_{t,k_2}$. We construct a proposal distribution $p(x_{t,k}|x_{t,u(k)})$ such that the above two constraints simultaneously hold:

$$x_{t,k}^{(n)} = x_{t,u|u}^{(n)} + \alpha\eta_{t-1}^{k,u} + (1-\alpha)\eta_1^{k,u} + v_{t,k}^{(n)}, \tag{2}$$

where $v_{t,k}^{(n)}$ is the noise vector that follows the normal distribution with mean vector 0 and covariance matrix Σ, and α is a tuning parameter whose range is $0 < \alpha < 1$ and controls the importance between the covariation and the relative positions. By this construction, the relative positions of the cells in the initial frame are conserved in the long run since the time series generated from the proposal distribution is stationary. We see this fact by taking the average of Equation (2) over the index set of particles $\{n|1\cdots,N\}$ to derive the following autoregressive model:

$$(\eta_t^{k,u} - \eta_1^{k,u}) = \alpha(\eta_{t-1}^{k,u} - \eta_1^{k,u}) + \overline{v}_{t,k},$$

where $\overline{v}_{t,k}$ is the average of the noise vectors over the index set of particles. Since the autoregressive coefficient α satisfies $0 < \alpha < 1$, $\eta_t^{k,u}$ is a stationary process with mean $\eta_1^{k,u}$, meaning that the relative positions of the cells in the initial frame are conserved in the long run.

The remaining task to construct the procedure for generating particles is to avoid collisions with neighboring trackers. To this end, we combine a rejection sampling with the proposal distribution. After generating a particle based on the proposal distribution, we reject the particle if the distance between k and u is small compared with the radii of a cell nucleus. Formally, we reject the particle with a probability proportional to $\exp\{-||\eta_t^{k,u}||^2/\lambda^2\}$, where λ denotes the predefined radius of the cell nuclei, and accept it otherwise.

2.2.3 Calculation of the Particle Weights

We next describe how we evaluate the importance of a particle that tracks a cell nucleus. The point-spread function

(PSF) is typically used for evaluating the similarity of objects with ellipsoidal shapes [2]. The cell nuclei in our motivated datasets are roughly ellipsoidal but often slightly deformed. To evaluate the similarity between cell nuclei in the presence of a deformation, we define the importance of a particle as a similarity between 3D subimages around the kth cell centroid in the initial and current frames. Suppose $y_t^{(w)}(x)$ is an augmented vector that corresponds to a 3D subimage at time t with center $x \in \mathbb{R}^3$ and window width parameter $w = (w_1, w_2, w_3) \in \mathbb{N}_0^3$. Here, the window is defined as a set of voxels in the cuboid with a center voxel including x and edge lengths $2w_1 + 1$, $2w_2 + 1$, and $2w_3 + 1$. We also suppose that $\hat{x}_{1,k}$ is the initial position of the kth cell centroid detected by the DP-means algorithm. We then define a likelihood function of particle $x_{t,k}^{(n)}$ as follows:

$$p(y_{t,k}|x_{t,k}^{(n)}) \propto \exp\left\{ -\frac{||y_t^{(w)}(x_{t,k}^{(n)}) - y_1^{(w)}(\hat{x}_{1,k})||^2}{2\sigma^2 W} \right\},$$

where σ^2 is the parameter that controls the variance of the 3D-subimage similarity and W is the number of non-zero elements for the vector defined as the element-wise sum of $y_t^{(w)}(x_{t,k}^{(n)})$ and $y_1^{(w)}(x_{t,k}^{(n)})$. We note that the difference is averaged over nonzero elements in the elementwise sum of the subimages in order to avoid an unintended similarity increase derived from the voxels within the window but outside the nucleus region. We also note that the window parameter w should be chosen to be larger than the maximum size of the cell nuclei so that all the information about shapes and intensities are included in the window.

2.2.4 Dealing with the Disappearance of the Targets to be Tracked

Another difficulty in tracking cells is that some neurons go out of the image space because of the movement of the nematode. We track the cells that are out of the image space by the following scheme. If a generated particle in the prediction step goes out of the image space, we add the particle to a member of the filter ensemble without resampling. The reason why we skip the filtering step for such a particle is that the likelihood of the particle cannot be evaluated whereas the particle should be kept considering the possibility that the target truly goes out of the image space.

2.2.5 Software

We developed a software suite called SPF-CellTracker. Our software is composed of three executables, namely, *convert*, *track*, and *view*. The first software *convert* converts a set of 2D images that comprise 4D live-cell imaging data into a single file encoded as our original binary format. During the conversion, the average subtraction and the 3D median filter can be optionally applied for each 3D image in order to remove background noise and salt-and-pepper noise. The second software *track* is the main software for detecting and tracking multiple cells based on the SPF algorithm from the converted 4D image file. This software was implemented purely in the C language considering the running speed, which will be presented in the following section. The third software *view* is for visualizing the 4D image data with the tracking result. An interesting feature of this visualization software is the capability to zoom in, zoom out, and rotate a 4D image during playing the 4D image data. Our software is freely available at the supplementary website (S1) listed in Appendix.

3 Experiments

We here present the performance comparisons related to cell tracking and detection using synthetic data and real 4D live-cell imaging data. All of the supplementary videos described in this section are available at the supplementary website (S2) listed in Appendix.

3.1 Comparison of the Tracking Performance

3.1.1 Application to Synthetic Data

We begin with how we generated the synthetic datasets. We constructed a model that simulates real 4D imaging data with the following characteristics: (1) the relative positions of the cells obtained from the real 4D imaging data are preserved in the long run, (2) the movements of the nearly located cells strongly correlate, (3) the cells are imaged as globular-like objects and it is difficult to discriminate the cells by shape and size, and (4) the cells do not divide but occasionally disappear. We assumed the use of confocal microscopes for imaging the cell nuclei. Typical confocal microscopes generate 3D images with a longer step size along the z-axis than the edge length of a pixel in the xy-plane. We therefore set the step size along the z-axis to be three times larger than those in the xy-plane. To be reasonably consistent with our real 4D live-cell imaging data, we set the resolution and the number of frames to $512 \times 256 \times 20$ and 500, respectively. We used the results of the cell detection by the DP-means algorithm as the initial positions of cells for each of the three real 4D imaging data. For the three datasets, 114, 120 and 115 cells were detected. We used the prediction model in the SPF as a simulation model of cells' correlated moves. The locations of the root nucleus after the second frame were not changed in order to stabilize the position of a simulated nematode itself. The position of the remaining non-root nuclei after the second frame were generated by Equation (2) with $\alpha = 0.6$ and $\Sigma^{-1/2} = \mathrm{diag}(0.6, 0.6, 0.03)$ to simulate the correlated movements of the nearly located cells and the preservation of the relative positions among the cells. These parameters were determined on the basis of the visual similarity between the generated data and the real imaging data through trial and error. Especially, we set the variance of the noise along the z-axis to nearly zero since the movement of a nematode in the dorsal-ventral axis is considerably small even if the movement in the anteroposterior axis is relatively large. To generate globular-like objects as images of cells, we used an anisotropic PSF. Suppose $d_\Lambda(x, \mu)^2 = (x - \mu)^T \Lambda^{-1} (x - \mu)$ is the square of the Mahalanobis distance between $x \in \mathbb{R}^3$ and $\mu \in \mathbb{R}^3$ with the covariance matrix $\Lambda \in \mathbb{R}^{3\times3}$. We define a PSF with the nucleus center μ and the shape parameter Λ as follows:

$$f(x; \mu, \Lambda) = \begin{cases} c \exp\left\{ -\frac{1}{2} d_\Lambda(x, \mu)^2 \right\} & \text{if } d_\Lambda(x, \mu) < 1 \\ 0 & \text{otherwise,} \end{cases}$$

where c denotes the intensity at the nucleus center μ. For all the nuclei, we used the same intensity parameter c and shape parameter Λ, which led to the most severe condition for tracking so that the cell nuclei were not distinguishable from the intensity and shape information. We set the shape

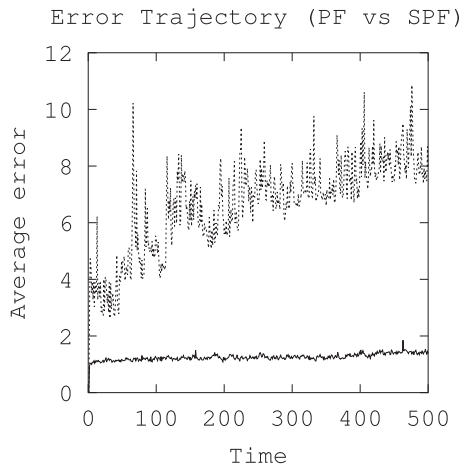


Fig. 2. Time series plots of the average tracking errors per cell nucleus per trial of the standard PF (dotted line) and SPF (solid line) for synthetic dataset 1. Errors were calculated as the root mean square error (RMSE) from the true positions of the synthetic nuclei. The unit distance was defined as the edge length of a voxel in the xy-plane. The error along the z-axis was expanded to three times of that in the xy-plane considering the difference in the physical length between the xy-plane and the z-axis.

parameter $\Lambda = \mathrm{diag}(9, 6, 3)$, which corresponded to an ellipsoid where its principal axes were parallel to the x, y, and z axes and those lengths were $9, 6$, and 3 voxels. A nucleus image was deleted with a probability of 0.03 for each nucleus for each frame in order to simulate its disappearance. Supplementary videos 1, 2 and 3 show the resulting synthetic datasets. The videos show similar motions of the cell nuclei in the real datasets although the large deformation of a nematode body such as the S-curve is not reproduced.

Next, we proceed to the evaluation of the tracking performance for the synthetic datasets described above. For the three datasets, we conducted 20 trials for each PF and SPF. We set the same parameters to those used for generating the synthetic image data. We then computed the root-mean-square errors (RMSEs) and counted the number of tracking failures. In computing the RMSEs, the distance along z-axis was expanded to three times its initial length since the physical length of a voxel along the z-axis was assumed to be three times larger than that in the xy-plane in order to simulate the anisotropy of the 3D imaging data. Fig. 2 shows the time series plots of the average RMSE of the PF and SPF per nucleus per trial for synthetic dataset 1. The error of the PF gradually increased as time t increased, whereas the error of the SPF was considerably small and stable. Table 1 summarizes the failure counts of the whole applications to the three datasets. The tracking failure was defined as the case where the distance between the true position and its estimate was greater than the radius of the synthetic nucleus, i.e., 4.5 voxels. The failure count was *not* incremented if a tracking failure was inherited by the next frame, but it *was* incremented if the tracking failure occurred again after the recovery from the tracking failure. As shown, the SPF drastically reduced the number of tracking failures in comparison with those of PF.

3.1.2 Application to Real 4D Live-Cell Imaging Data

We evaluated the tracking performance of the SPF and the state-of-the-art method reported by Tokunaga et al.[19]. The data we used were D1, D2, and D3 in Data II reported by them, i.e., is, 4D live-cell imaging data obtained by imaging

TABLE 1
Average Number of Tracking Failures Per Nucleus Per Trial in 500 Frames

Data	PF	SPF
1	37.36 (0.234)	1.43 (0.038)
2	41.60 (0.179)	0.78 (0.016)
3	37.22 (0.218)	1.14 (0.013)

PF and SPF were applied to three synthetic datasets. The numbers of tracking failures were averaged over the number of nuclei and the number of trials. The estimated standard errors are shown in parentheses.

the nuclei of *C. elegans* neurons (Supplementary video 4-6). Each of the datasets is composed of 500 frames of 3D images with a resolution $512 \times 256 \times 20$, i.e., 20 z-slices of 2D images with a resolution 512×256. Since the background noise level of the datasets changes according to the depth of the 3D image, we approximately estimated the background noise level as the average over the fluorescent intensities of a 2D image and subtracted it from the 2D image. We then applied the median filter a window size of $3 \times 3 \times 1$ to remove the salt-and-pepper noise. To obtain the tracking results independent from the quality of the cell detection, we used the same starting positions for both methods. The starting positions were obtained by using the repulsive hill-climbing (RPHC) algorithm [19]. All the parameters required for the SPF were manually-tuned and the same parameters were used for all of the three imaging datasets. The parameter set used for the SPF and the resulting tracking animations of the SPF (Supplementary video 7-9) and the method of Tokunaga et al. [19] (Supplementary video 10-12) are available at the supplementary website. Fig. 3 shows tracking results of both methods in the final frame of D3 in Data II. The top panel of the figure shows the starting positions of the trackers. Five trackers that did not track the cell centroids in the bottom of the image space were derived from the false positives of the cell-detection in the initial frame. The middle and bottom panels in Fig. 3 show the positions of trackers in the final frame for [19] and SPF, respectively. The figure suggests that, at least, large inconsistencies between the cell centroids and the positions of the trackers for both methods did not exist.

We evaluated the tracking performance using the final frame of the original datasets by manual verification. To reduce the errors of manual verification as much as possible, we implemented the following features for our 4D image viewer: (1) focusing only on one cell and on the corresponding tracker, (2) starting and stopping at an arbitrary frame, (3) rotation of the 4D image, and (4) adjustment of the intensity threshold that determines whether or not a voxel is displayed. These features of the 4D image viewer are introduced in Supplementary video 13. By using the features of our 4D image viewer, we calculated the success rates of both methods according to the following principles:

- A tracker was counted as a *success* if the distance from the corresponding cell centroid was within five voxels.
- A tracker was excluded from the calculation if it was an incorrectly detected cell centroid in the initial frame.
- A tracker was also excluded from the calculation if the corresponding cell centroid was out of view in the final frame.

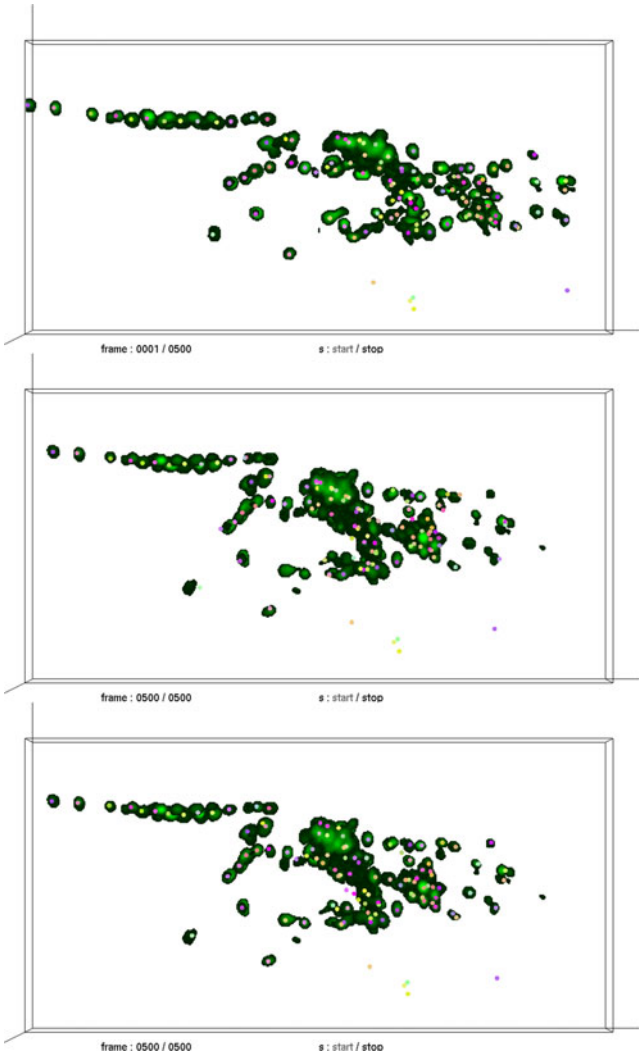


Fig. 3. Comparison of the tracking performance for D3 in DATA II. (Top) Starting positions of the trackers for both methods. The trackers are depicted by colored circles. (Middle) Tracking result of the method of Tokunaga et al. [19] in the final frame. (Bottom) Tracking result of the SPF in the final frame.

Table 2 shows the success rates of both methods based on the manual verification. The SPF considerably improved the tracking performance compared with the method of Tokunaga et al. [19] for all the three datasets D1-D3 in Data II. Among the three datasets, the tracking performance in D3 was noticeably lower than that in the other datasets for both methods. This was due to the differences of nematodes' body postures and movements in the final frame. The body postures in the final frame of D1 and D2 are roughly straight, and the movements of the nematode are moderate (Supplementary video 4 and 5). On the contrary, the nematode in the final frame of D3 is largely shrunk along the anterior-posterior axis and the nematode's movement is rapid (Supplementary video 6).

TABLE 2
Rate of Trackers that Correctly Tracked the Corresponding Cells in the Final Frame $t = 500$ for D1-D3 in Data II

Data	Tokunaga et al. (2014)	SPF
D1 (120)	0.8000	0.9524
D2 (124)	0.6860	0.9669
D3 (111)	0.4467	0.8679

The number of cells in the initial frame detected by the RPHC algorithm is shown in parentheses in the first column. The consistency of the tracking between the initial and the final frames was manually checked for each cell using our 4D image viewer.

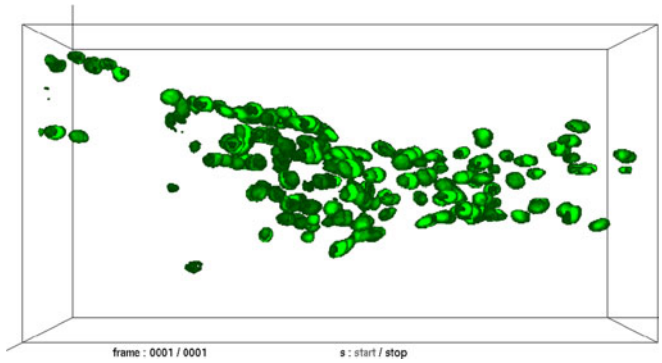


Fig. 4. A 3D still image in DATA I that captured the nuclei of *C. elegans* neurons after the noise removal.

3.2 Comparison of the Cell Detection Performance

Next, we evaluated the detection performance of the DP-means algorithm by comparing it with that of the RPHC algorithm [19]. The datasets we used was DATA I, which was reported by them. DATA I was obtained by imaging the nuclei of the nematode's neurons, but it was different from Data II in that (1) it is composed of distinct 3D still images instead of 4D images and (2) the cell centroids were manually annotated in order to obtain the ground truth of the cell detection problem. The resolution of the xy-plane of all the 10 datasets was 512×256, and the numbers of z-slices for the datasets ranged from 119 to 203. A 3D image in Data I after the noise removal is shown in Fig. 4. To evaluate the performance of both algorithm, we counted the number of true positives (TP), the number of false positives (FP), and the number of false negatives (FN). A position detected by an algorithm was defined as a true positive if the position was within five voxels from a manually identified position, and was defined as a false positive otherwise. An annotated position that was not detected by an algorithm was defined as a false negative. We then computed the true positive rates and the false discovery rates defined as TP/(TP+FN) and FP/(FP+TP), respectively. Table 3 shows the false discovery rates and the true positive rates of both algorithms. The radius parameter λ of the DP-means algorithm was set to eight voxels, which were close to the maximum radius of the cells. The true positive rate of the DP-means algorithm is roughly the same as RPHC while its false discovery rate were higher than RPHC. Fig. 5 shows an effect of the radius parameter λ on the cell detection performance of the DP-means algorithm, suggesting that the DP-means algorithm achieved the best for $\lambda = 8$ and $\lambda = 9$, which were close to the maximum size of the cells.

TABLE 3
False Discovery Rate and True Positive Rate of the Cell Detection

	RPHC	DP-means ($\lambda = 8$)
True positive rate	0.8041 (0.0362)	0.8004 (0.0710)
False discovery rate	0.0301 (0.0305)	0.1362 (0.1060)

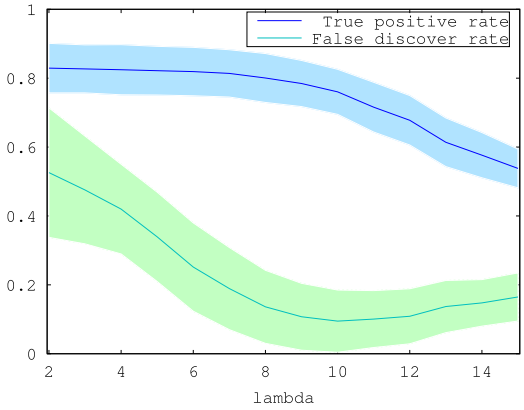


Fig. 5. The radius parameter λ and the cell detection performance of the DP-means algorithm. The accuracy was averaged over the results of ten 3D still images in DATA I. Ranges within 1 standard deviation are indicated by shaded regions.

3.3 Computational Time

We computed the CPU time and real time for D1 in DATA II, which was composed of 500 frames of 3D images with a resolution $512 \times 256 \times 20$. We used an iMac (27-inch, Late 2013, OS X 10.9.5) with a 3.2 GHz Intel Core i5 CPU and 8 GB of RAM as our computational environment. OpenMP implemented in GCC 4.8 was used as a tool for parallelization. We used the same parameter set as that used in the evaluation of the tracking performance for the real 4D live-cell imaging data. Especially, we set the number of particles for tracking a cell to 1000, i.e., the total number of particles for tracking 100 cells was 10^5. To measure the scalability of our method, we randomly selected 20, 40, 60, 80, and 100 cells without replacement among 114 cells. Fig. 6 shows the CPU time and real time consumed for tracking 500 frames in the dataset D1 in DATA II. The CPU time and real time were averaged over 20 trials. The CPU time in the most severe condition, i.e., $K = 100$, was less than 120 seconds, suggesting that our software is sufficiently of practical use.

3.3.1 Discussion

We here summarize the results suggested by the numerical studies.

- The tracking performance of the standard PF was substantially improved by the proposed method for synthetic datasets (Table 1).
- The tracking errors of the PF were sequentially inherited by the next frame, whereas those of the SPF were not severely inherited. This result suggests a recovery from tracking failures (Fig. 2).
- Tracking performance can be evaluated by using the features implemented in our 4D image viewer (Supplementary video 13).
- The SPF considerably outperformed the method of Tokunaga et al. in terms of the success rates of the tracking in the final frame for all the three datasets D1-D3 in Data II (Table 2).
- The parameters of the SPF were set to the same ones for all the three datasets D1-D3 in DATA II, suggesting a robustness of the SPF (Section 3.1).
- For the method of Tokunaga et al., some trackers were merged with other trackers and did not return to the cells to be tracked again (Supplementary video 10-12).

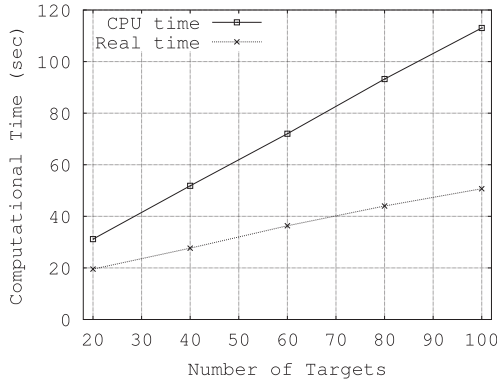


Fig. 6. Average time consumption of the SPF and its scalability to the number of targets. The CPU time and real time were evaluated for completing the tracking of all 500 frames of the dataset D1 in DATA II.

- For the SPF, some trackers often recovered from tracking failures and merged trackers were rarely observed, probably due to the integration of the spatial information such as the covariation, relative positions, and collision avoidance among cells (Supplementary videos 7-9).
- The true positive rates of the DP-means algorithm and the RPHC algorithm for detecting cell centroids were roughly the same, whereas the false positive rate of the DP-means algorithm was higher than that of the RPHC algorithm. This suggests an advantage of the RPHC algorithm over the DP-means algorithm for the detection of cell centroids (Table 3).
- The CPU times of the SPF were proportional to the number of targets, suggesting a favorable property in terms of the scalability to the data size. (Fig. 6).

4 Conclusion

In this study, we aimed at the tracking of more than a hundred cells in 4D live-cell imaging data. One important characteristic of live-cell imaging data is that the cells to be tracked are densely scattered and visually similar. For these imaging data, the particle filter often mistakes a cell of interest for the other cells since the visual similarity among the cell nuclei makes their discrimination difficult. Fortunately, our 4D live-cell imaging data share a characteristic that is useful for accurate tracking: cells' movements in the 4D live-cell imaging data are strongly correlated and the relative positions among the target cells are roughly conserved.

To address the tracking issue, we designed an MRF that models the covariation and preservation of the relative positions among cells. To avoid the inefficiency of JPF and MCMC sampling, we also proposed a novel sampling algorithm, which we call spatial particle filter. The proposal distribution in the prediction step draws more accurate particles than those generated by dynamics since the proposal distribution shares both temporal and spatial information about a cell's movements. The SPF tracks cells by a sweep of an MRF tree in the spatial order for each frame; this allows for an effective simultaneous tracking. The MRF tree is automatically constructed by computing an MST among the initial locations of all targets. We applied the proposed method to synthetic data and to our 4D live-cell imaging data of *C. elegans*. The results showed that our algorithm required less computation yet achieved higher accuracy than our previous algorithm.

Future work includes performance comparisons with tracking methods such as detection-and-linking and contour-evolution methods. Another direction is to expand the applicability of SPF to a wider class of 4D imaging data, for example, 4D images of chromosome arrangements during the cell division. Tracking chromosomes is a challenging problem since they duplicate, split, and change their shapes according to the state of cell division. We expect the proposed method to be a reasonable candidate for tracking multiple targets in a wider class of 4D live-cell imaging data.

Appendix

The source codes of SPF-CellTracker are available at the following github repository (S1). All the supplementary videos (1-13) described in Experiments section are available at the following supplementary website (S2).

(S1) https://github.com/ohirose/spf
(S2) https://sites.google.com/site/webosamuhirose/spf

Conflict of interest

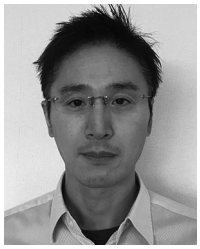
None declared.

Acknowledgments

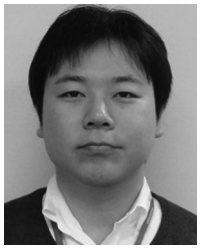
This study is supported in part by the CREST program "Creation of Fundamental Technologies for Understanding and Control of Biosystem Dynamics" of the Japan Science and Technology Agency (JST).

References

[1] M. Maška, et al., "A benchmark for comparison of cell tracking algorithms," *Bioinf.*, vol. 30, no. 11, pp. 1609–1617, 2014.
[2] E. Meijering, et al., "Methods for cell and particle tracking," *Methods Enzymol.*, vol. 504, no. 9, pp. 183–200, 2012.
[3] N. Chenouard, et al., "Objective comparison of particle tracking methods," *Nat. Methods*, vol. 11, no. 3, pp. 281–289, 2014.
[4] O. Al-Kofahi, et al., "Automated cell lineage construction: A rapid method to analyze clonal development established with murine neural progenitor cells," *Cell Cycle*, vol. 5, no. 3, pp. 327–335, 2006.
[5] N. Chenouard, et al., "Multiple hypothesis tracking for cluttered biological image sequences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2736–2750, Nov. 2013.
[6] F. Li, et al., "Multiple nuclei tracking using integer programming for quantitative cancer cell cycle analysis," *IEEE Trans. Med. Imaging*, vol. 29, no. 1, pp. 96–105, Jan. 2010.
[7] E. Wait, et al., "Visualization and correction of automated segmentation, tracking and lineaging from 5-D stem cell image sequences," *BMC Bioinf.*, vol. 15, no. 1, pp. 328–340, 2007.
[8] A. Dufour, et al., "3-D active meshes: Fast discrete deformable models for cell tracking in 3-D time-lapse microscopy," *IEEE Trans. Image Process.*, vol. 20, no. 7, pp. 1925–1937, Jul. 2011.
[9] M. Maška, et al., "Segmentation and shape tracking of whole fluorescent cells based on the Chan-Vese model," *IEEE Trans. Med. Imaging*, vol. 32, no. 6, pp. 995–1006, Jun. 2013.
[10] I. Smal, et al., "Rao-Blackwellized marginal particle filtering for multiple object tracking in molecular bioimaging," in *Proc. Int. Conf. Inf. Process. Med. Imaging*, 2007, vol. 20, pp. 110–121.
[11] D. Clark, et al., "Temporal activity patterns in thermosensory neurons of freely moving Caenorhabditis elegans encode spatial temporal gradients," *J. Neurosci.*, vol. 27, no. 23, pp. 6083–6090, 2007.
[12] J. Deutscher, et al., "Articulated body motion capture by annealed particle filtering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2000, vol. 2, pp. 126–133.
[13] H. T. Chen, et al., "Multi-object tracking using dynamical graph matching," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2001, vol. 2, pp. 210–217.
[14] Y. Rui, et al., "Better proposal distributions: object tracking using unscented particle filter," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2001, vol. 2, pp. 786–793.
[15] C. Gomila, "Graph-based object tracking," in *Proc. Int. Conf. Image Process.*, 2003, vol. 2, pp. 41–44.
[16] K. Nummiaro, et al., "Object tracking with an adaptive color-based particle filter," in *Proc. Joint Pattern Recognit. Symp.*, 2002, pp. 353–360.
[17] J. Wang, et al., "Online selecting discriminative tracking features using particle filter," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, vol. 2, pp. 1037–1042.
[18] Z. Khan, et al., "An MCMC-based particle filter for tracking multiple interacting targets," in *Proc. 8th Eur. Conf. Comput. Vis.*, 2004, pp. 279–290.
[19] T. Tokunaga, et al., "Automated detection and tracking of many cells by using 4D live-cell imaging data," *Bioinf.*, vol. 30, pp. i43–i51, 2014.
[20] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Fluids Eng.*, vol. 82, no. 1, pp. 35–45, 1960.
[21] A. Doucet, et al., "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics Comput.*, vol. 10, pp. 197–208, 2000.
[22] J. Vermaak, et al., "Maintaining multimodality through mixture tracking," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, 2003, vol. 2, pp. 1110–1116.
[23] B. Kulis, et al., "Revisiting k-means : New algorithms via Bayesian nonparametrics," in *Proc. 29th Int. Conf. Mach. Learning*, 2012, pp. 513–520.
[24] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proc. Amer. Math. Soc.*, vol. 7, no. 1, pp. 48–50, 1956.



Osamu Hirose is an assistant professor in the Faculty of Electrical and Electronic Engineering, Kanazawa University. His recent research interests focus on developing machine learning techniques for automating the analysis of digital images and movies, especially those related to biology. His research interests also include 3D shape modeling and registration.



Shotaro Kawaguchi was a graduate student in the Graduate School of Natural Science and Technology, Kanazawa University, when he was engaged in this work. His research was centered on developing automating tools for analyzing bioimages. Currently, he works at NTT DoCoMo, Inc.

Terumasa Tokunaga is an associate professor in the Department of Design and Informatics, Faculty of Computer Science & Systems Engineering, Kyushu Institute of Technology. His recent research interests focus on the development of spatiotemporal pattern mining techniques for analyzing image time series.

Yu Toyoshima is a research associate in the Department of Biological Sciences at the Graduate School of Science, University of Tokyo. His current research interests focus on bio-image informatics for whole-brain activity measurement. He is also interested in understanding how living nervous systems perform computations by combining biological experiments, and mathematical analyses.

Takayuki Teramoto is an associate professor in the Department of Biology, Kyushu University. His research focuses on how the neuronal network works applying Ca2+ imaging techniques on a small organism, C. elegans as a model system.

Sayuri Kuge is a postdoctoral fellow in the Department of Biology, Kyushu University. Her main research is the development of calcium probes.

Takeshi Ishihara is a professor in the Department of Biology, Faculty of Sciences and Graduate School of Systems Life Sciences, Kyushu University (Japan). He is interested in sensory processing in the central nervous system and mechanisms of active forgetting.

Yuichi Iino is a full professor in the Department of Biological Sciences, Graduate School of Science, University of Tokyo, and is currently a department chair. His research has focused on the molecular, cellular, and neural circuit mechanisms of behaviors, especially those of the plasticity of chemotaxis in the nematode C. elegans. He and his group now aim at answering this question by looking at the activity of the whole nervous system and its dynamics.

Ryo Yoshida is an associate professor in the Department of Statistical Modeling and the head of the Data Science Center for Creative Design and Manufacturing, Institute of Statistical Mathematics, an invited researcher in the National Institute for Materials Science. His research interests focus on the development of machine learning algorithms for bioinformatics and the discovery of innovative functional materials.