Probabilistic Tracking of Particles in Fluorescence Microscopy Time-lapse Images using Kalman Filtering

Tae Yeon Yoo TYOO@FAS.HARVARD.EDU
School of Engineering and Sciences, Harvard University, Cambridge, MA 02138 USA

Abstract

Accurate particle tracking algorithm is essential in a variety of biological and biomedical research where fluorescent particles are utilized to learn about biological fluid dynamics or mechanical properties. Fluorescent particle tracking is challenging because fluorescent microscopy images often have limited signal-to-noise ratio, and the trajectories of particles are unexpectedly truncated and overlapped. In this project, I modeled particle dynamics as Hidden Markov Chain, and use Kalman filter to find posterior on true state of particles. In this scheme, trajectories are determined in a probabilistic manner, rather than through hard linking. At the end, the probabilisic algorithm is evaluated and compared to the deterministic algorithm that is being widely used.

1. Introduction

Fluorescent particles are widely used in physics and biomedical research. The application ranges from the study of hydrodynamics of fluid ([1], [2]) to the detection, tracking and chracterization of cancer cells and inflammation.[3], [4] Moreover, in cell biology and single-molecule biophysics research, many important biological mechanisms are studied by labeling a protein of interest with a fluorescent protein or dye and observing the dynamics of individual or a group of the proteins through fluorescence microscopy.[5][6],[7] In such studies, accurate tracking of the spots in fluorescence microscopy image is essential. However, accurate particle tracking is often challeging, especially when the motion of particles is not understood, images are noisy, and the density of particle is high. A variety of particle tracking methods and softwares have been developed and actively used for the analysis of fluorescent particle time-lapse images. A paper published in 2014 [10] presented an objective comparison of particle tracking methods based on an open competition held in 2012. (http://www.bioimageanalysis.org/track/) A total of 14 teams took up the challenge. Among 14 different methods, only two adopted probabilistic algorithms to predict the trajectories of particles, and one of them was place in top three in the challenge. The method takes trajectory of particle as Hidden Markov Chain and uses Kalman filter to make inference. In the final project, I decided to reproduce their method, described in [8] and [9], not only because I actively use particle tracking methods in my research, but also because I want to learn more about how Kalman filter that I learned in class is used in a real application.

For the final project assignment of CS281, Advanced Machine Learning, 2015

2. Key Assumptions and Technical Background

Throughout this report, I assume that the size of fluorescence particle is smaller than the diffraction limit, and therefore it appears to be symmetric 2D Gaussian in fluorescence image. That is, the fluorescence intensity $z(x, y)$ can be written as:

$$z(x,y) = I_b + (I_{max} - I_b)\exp(-\frac{(x-x_c)^2 + (y-y_c)^2}{2\sigma_{xy}^2}) \tag{1}$$

where I_b is the background intensity, I_{max} is the peak intensity, x_c and y_c indicate the position of the center of 2D gaussian, and σ_{xy} is the standard deviation of the gaussian. I also assume that the motion of particles is close to random walk, though this method can easily be extended to other cases.

Throughout the paper, $\mathbf{x}$ and $\mathbf{y}$ denote the true and measured state of a particle, respectively. That is, $\mathbf{x}$ and $\mathbf{y}$ are defined to be:

$$\mathbf{x} = [x, \dot{x}, y, \dot{y}, I_{max}, \sigma_{xy}]$$
$$\mathbf{y} = [x', y', I'_{max}, \sigma'_{xy}]$$

The true state $\mathbf{x}$ of a particle is infered through Kalman filtering scheme, using the corresponding fluorescence image

and multiple *generated* measured state $\mathbf{y}$'s. Multiple measured states $\mathbf{y}$'s are generated per filter through a procedure decribed in the next section, and the contribution of each measured state is determined by calculating the likelihood of the state given the corresponding image.

3. Model

I modeled particle dynamics as Hidden Markov Chain as shown in Fig 1. $\mathbf{F}$ and $\mathbf{H}$ are the transition and measurement matrices, respectively. That is,

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \mathbf{F}\mathbf{x}_{t-1}, \mathbf{Q})$$
$$p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t; \mathbf{H}\mathbf{x}_t, \mathbf{R})$$

where covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ determine the noise in transition and measurement, respectively. For random-walking particles, these matrices are set to $\mathbf{F} = \mathrm{diag}(1, 1, 1, 1, 1, 1)$ and:

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

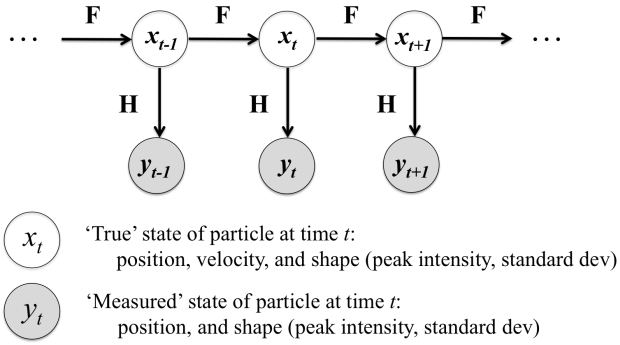

Figure 1. Graphical model of Hidden Markov Model for fluorescent particle tracking.

4. Inference

4.1. Kalman Filtering

Kalman filtering is used to infer posterior $p(\mathbf{x}_t|\mathbf{y}_{1:t})$, where the posterior is assumed to be normal:

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \mathcal{N}(\mathbf{x}_t; \mathbf{m}_t, \mathbf{P}_t)$$

Given the posterior in the previous time step $p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \mathcal{N}(\mathbf{x}_t; \mathbf{m}_t, \mathbf{P}_t)$, the first guess of the mean vector $\mathbf{m}$ an d covariance $\mathbf{P}$ of the posteior of the current time step t is calculated as below:

$$\hat{\mathbf{m}} = \mathbf{F}\mathbf{m}_{t-1} \hat{\mathbf{P}} = \mathbf{F}\mathbf{P}_{t-1}\mathbf{F}^T + \mathbf{Q}$$

The first guess of the mean vector $\hat{\mathbf{m}}$ and covariance matrix $\hat{\mathbf{P}}$ are used to calculate the predicted measurement and Kalman gain in the current time step:

$$\hat{\mathbf{y}} = \mathbf{H}\hat{\mathbf{m}} \tag{2}$$
$$\mathbf{K} = \hat{\mathbf{P}}\mathbf{H}^T(\mathbf{H}\hat{\mathbf{P}}\mathbf{H}^T + \mathbf{R})^{-1} \tag{3}$$

The first guesses $\hat{\mathbf{m}}$ and $\hat{\mathbf{P}}$ are then updated by *innovation* ν and the Kalman gain obtained above:

$$\mathbf{m}_t = \hat{\mathbf{m}} + \mathbf{K}\nu \tag{4}$$
$$\mathbf{P}_t = (\mathbb{I} - \mathbf{K}\mathbf{H})\hat{\mathbf{P}} \tag{5}$$

The way to obtain the *innovation* ν will be explained in the following sections.

4.2. Measurement Generation

In a typical Kalman filtering scheme, innovation is simply the difference between *single* measurement and the predicted measurement, $y - \hat{y}$. In this case, however, it is tricky to tranlate the fluorescence image of particle into a sinlge measurement of position and shape of the particle. The ways many other particle tracking algorithms use to obtain the measurements of particles from images are 1) applying spatial filter to images and picking up the locations above certain threshold, and/or 2) fitting a mixture of appearance models – in this case a mixture 2D gaussian function – to image to determine the locations and appearances of particles. These approaches are called *bottom-up* approaches. The bottom-up approaches are known to work well only when signal-to-noise ratio is large. When data is noisy, the failure in particle recognition leads to the truncation of trajectory.

Instead of using a single measurement, here I simulate *multiple* measurements near the position determined by *bottom-up* approach $\mathbf{y}_{SEF}$, AND near the predicted measurement $\hat{\mathbf{y}}$ obtained from Eq 2. The matching between $\mathbf{y}_{SEF}$ and $\hat{\mathbf{y}}$ is done by global nearest-neighbor searching, described in [11]. If there is no position determined by *bottom-up* approach, only predicted measurement is used to generate measurements. In this way, we may keep updating a trajectory even if the bottom-up approach fails to find the particle. The simulated measurments are depicted in Fig 2. The spread of the measurements is determined by the covariance matrix, $\mathbf{P}$. For more details about measurements, refer [8].

These multiple generated measurements $\mathbf{y}_i$'s are combined into a single innovation by:

$$\nu = \sum_i \beta_i(\mathbf{y}_i - \hat{\mathbf{y}}) \tag{6}$$

where β_i is the weight in each measurement $\mathbf{y}_i$. In the next section, I will explain how the weights β_i's are defined and calculated.

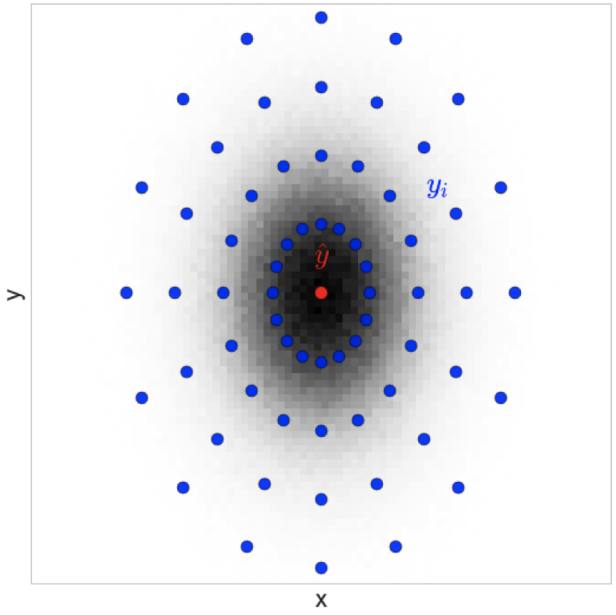

Figure 2. Elliptical measurements (blue) around the predicted measurement, $\hat{\mathbf{y}}$ (red), generated based on the distribution $\mathcal{N}(\Phi\hat{\mathbf{y}}, \mathbf{H}\hat{\mathbf{P}}\mathbf{H}^T + \mathbf{R})$ (grey shaded) where Φ projects measurements onto 2D positions space.

4.3. Getting Combined Innovation

Simply speaking, the weight β_i's represent how likely the measurement $\mathbf{y}_i$ is to be real based on the corresponding images. Thus, we may write $\mathbf{y}_i$ as:

$$\beta_i = p(\mathbf{z}|\mathbf{y}_i) = \frac{p_0(\mathbf{z}|\mathbf{y}_i)}{p_b(\mathbf{z}|\mathbf{b})}$$

where

$$p_0(\mathbf{z}|\mathbf{y}_i) \propto \exp\left(-\frac{D(\mathbf{z}, \mathbf{g}(\mathbf{y}_i)}{2\sigma_n^2}\right)$$
$$p_b(\mathbf{z}|\mathbf{b}) \propto \exp\left(-\frac{D(\mathbf{z}, \mathbf{b})}{2\sigma_n^2}\right)$$

and $D(\bullet, \bullet)$ is the euclidean distance and $\mathbf{z}$ is the intensities around the position $\mathbf{y}_i$, which can be calculated from the appearance model Eq 1.
β_i needs to be corrected if there are other measurements of other filter very close to y_i. This procedure is explained in [8] and will not be covered here.

5. Experiments and Results

5.1. Test Data

The test data were generated by a plugin "ISBI Challenge tracking benchmark generator" running in a Java-platform software Icy (http://icy.bioimageanalysis.org/). The detailed procedure of generating data is described in the supplemental material of [10]. Multiple sets of time-lapse images of 100 time points were generated, each of which has the signal-to-noise ratio 1,2,4 or 7, and low, medium, or high particle densities. (Fig 3) Only random-walk motion is considered in this paper.
Ground truth of particle trajectories in the movies were saved for the evaluation of algorithm.

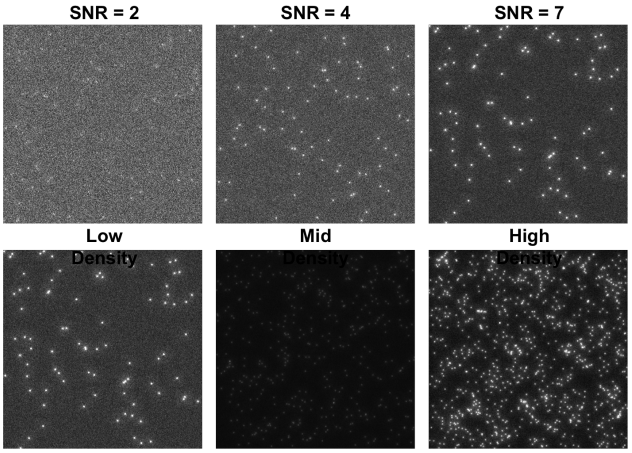

Figure 3. Example of simulated data.

5.2. Convergence of Kalman Filter

I first examined whether the Kalman filter of a particle converges as it updates trajectory of the particle. Letting the covariance matrix of posterior $\mathbf{P}$ represent the Kalman filter, I found that the Kalman filter rapidly stabilizes within only 3 to 5 frames. (Fig 4) Moreover, I found that there is a very small variation in the destined values of $\mathbf{P}$ of different trajectories.

5.3. Evaluation of Algorithm

I evaluated the performance of algorithm using 'Tracking Performance Computation' plugin in Icy, like in [10]. The plugin first pairs the trajectories in the output trajectory set Y of the method of interest with the trajectories in the ground-truth trajectory set X of the simulated data, through the minimization of the total pairwise distances. Then the output trajectory set Y is scored with respect to the ground-truth trajectory set X with many different measures. The

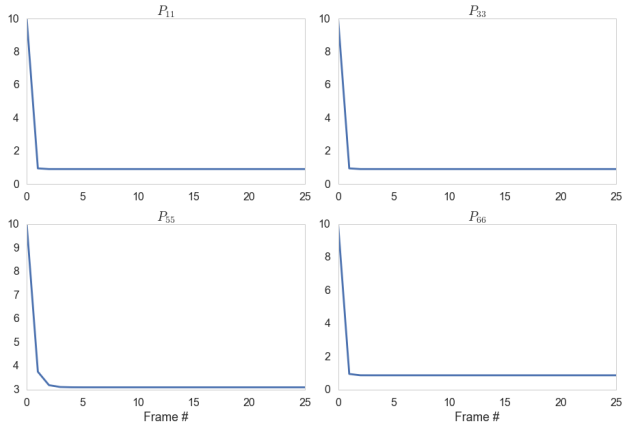

Figure 4. Convergence of the Kalman filter of a selected trajectory, reflected by the change of the covariance matrix initialized to $\mathbf{P} = \text{diag}(10, 10, 10, 10, 10, 10)$ at the beginning of the trajectory.

most informative measures are:

1. α: It scores the best possible pairing of trajectories between X and Y, ignoring the trajectories in Y that fail to be paired.

2. β: It scores the best possible pairing of trajectories between X and Y, but takes the unpaired trajectories in Y into account. It converges to zero as the number of spurious tracks increases.

α and β account both for association errors and localization errors. In addition to these, the following measures account specifically for association errors:

3. JSC: the Jaccard similarity coefficient for positions.

4. JSC_θ: the Jaccard similarity coefficient for trajectories.

and the following measure accounts specifically for localization error:

5. RMSE: The root-mean-square error in the true positive position pairs.

Mathematical definitions and more detailed descriptions of the measures are found in [10].
I also wrote the deterministic particle tracking algorithm in which the feature finding is done by Mexican-hat spatial filtering and thresholding, and linking between particles are achieved by finding the global nearest neighbors. [11] Then

Table 1. The performance measure α and β of the probabilistic (P) and deterministic (D) particle tracking methods for different data sets.

DATA SET	METHOD	α	β
SNR 7, LOW DENSITY	P	0.53	0.43
SNR 7, LOW DENSITY	D	0.63	0.54
SNR 4, LOW DENSITY	P	0.51	0.39
SNR 4, LOW DENSITY	D	0.62	0.51
SNR 2, LOW DENSITY	P	0.24	0.15
SNR 2, LOW DENSITY	D	0.16	0.12
SNR 7, MID DENSITY	P	0.33	0.22
SNR 7, MID DENSITY	D	0.41	0.38

Table 2. The performance measure JSC, JSC_θ, and RMSE of the probabilistic (P) and deterministic (D) particle tracking methods for different data sets.

DATA SET	METHOD	JSC	JSC_θ	RMSE
SNR 7, LOW DENSITY	P	0.57	0.68	1.39
SNR 7, LOW DENSITY	D	0.63	0.69	0.86
SNR 4, LOW DENSITY	P	0.51	0.60	1.21
SNR 4, LOW DENSITY	D	0.61	0.62	0.91
SNR 2, LOW DENSITY	P	0.19	0.27	1.19
SNR 2, LOW DENSITY	D	0.14	0.21	1.03
SNR 7, MID DENSITY	P	0.36	0.52	1.82
SNR 7, MID DENSITY	D	0.38	0.56	1.03

the performance of the probabilistic particle tracking algorithm was compared to the performance of the deterministic algorithm. The results are shown in Table 1 and Table 2. As expected, the prediction gets worse as SNR becomes lower and particle density larger. Sadly, I observed that the deterministic algorithms performs better than the probabilistic algorithm for all the data I examined except only the low SNR data.

6. Discussion

6.1. Performance of Probabilistic Particle Tracking Algorithm

Though [8] and[10] reported that probabilistic fluorescent particle tracking based on Kalman filtering outperforms many other particle tracking algorithms in a variety of data sets, I observed the opposite: the simplest deterministic algorithm that I often use for my biophysics research worked better for all data except for very noisy one. This is likely to be because I used the algorithm without the rigorous optimization of hyper-parameters due to the time constraint.
Even if the probabilistic particle tracking algorithm gives

more accurate prediction than deterministic ones, its application will be limited as it costs significant computation time, and it gets more expensive rapidly with the number of particles; even for the data with low density, the probabilistic algorithm takes ∼30 min while the simplest deterministic algorithm takes only few seconds. As already being applied in robotics, it may be useful for tracking of a single or very few objects.

6.2. Initialization of Kalman Filtering

Though I predicted that the initialization of Kalman filter at the begining of trajectory affects the accuracy significantly, I found out that the accuracy becomes only slightly better if the Kalman filter is initialized to the value it converges to. This is because as shown in Figure 4, the convergence of Kalman filter is very fast compared to the mean length of trajectory. However, the initialization might become more important for the data with short particle trajectories.

7. References

[1] Charogiannis, A., An, J., & Markides, CN. A simultaneous planar laser-induced fluorescence, particle image velocimetry and particle tracking velocimetry technique for the investigation of thin liquid-film flows. *Experimental Thermal and Fluid Science* 68, 516-536 (2015)

[2] Northrup MA., Kulp TJ., & Angel SM. Fluorescent particle image velocimetry: application to flow measurement in refractive index-matched porous media. *Applied Optics* 30, 3034-40 (1991)

[3] Hwang W, Ko HY, Lee JH, Kang H, Ryu SH, Song IC, Lee DS, & Kim S. A nucleolin-targeted multimodal nanoparticle imaging probe for tracking cancer cells using an aptamer. *Journal of Nuclear Medicine* 51 no. 1 98-105 (2009)

[4] Markus, MA, Napp, J, Behnke, T & Mitkovski, M. Tracking of Inhaled Near-Infrared Fluorescent Nanoparticles in Lungs of SKH-1 Mice with Allergic Airway Inflammation. *ACS Nano* (2015).

[5] Saxton, M.J. & Jacobson, K. Single-particle tracking: applications to membrane dynamics. *Annu. Rev. Biophys. Biomol. Struct.* 26, 373399 (1997).

[6] Dahan1, M., Lvi, S., Luccardini, C., Rostaing, P., & Riveau, B. Antoine Triller2,Diffusion Dynamics of Glycine Receptors Revealed by Single-Quantum Dot Tracking. *Science* 302, 442-445 (2003)

[7] Hern, JA., Baig, AH., Mashanov, GI., Birdsall, B., Corrie, JET., Lazareno, S., Molloy, JE., & Birdsall, NJM. Formation and dissociation of M1 muscarinic receptor dimers seen by total internal reflection fluorescence imaging of single molecules. *PNAS* 107, 2693-2698 (2009)

[8] Godinez, W. & Rohr, K. Tracking Multiple Particles in Fluorescence Time-Lapse Microscopy Images via Probabilistic Data Association. *IEEE Transactions on Medical Imaging* 34, 415432 (2015).

[9] Godinez, W. J., Lampe, M., Eils, R., Mller, B. & Rohr, K. Tracking multiple particles in fluorescence microscopy images via probabilistic data association. *IEEE* 19251928 (2011).

[10] Chenouard, N. et al. Objective comparison of particle tracking methods. *Nat. Methods* 11, 2819 (2014).

[11] Sbalzarini, IF & Koumoutsakos, P. Feature point tracking and trajectory analysis for video imaging in cell biology. *Journal of structural biology* (2005).

8. Code and Test Data

The Python code, generated test data sets, and test results can be found at `https://github.com/taebong/ParticleTrackingKalmanFilter`.

Acknowledgments

Thanks to Finale and all the TFs who put a lot of efforts into this amazing class.