FreeTrace

Junwoo Park
junwoo.park@sorbonne-universite.fr
alpha version: DOI:10.5281/zenodo.13336251

1 Introduction

FreeTrace infers individual trajectory from time-series images. To detect the particles and their positions at sub-pixel level, FreeTrace first extends the image sequences by sampling of noises at the edges of images. These extensions of images allow detecting the particles at the edges of images since FreeTrace utilizes sliding windows to calculate the particle's position at sub-pixel level. Next, FreeTrace estimates the existence of particle at a pixel with a given PSF function for each sliding window and makes a hypothesis map to determine whether a particle exists at a given sliding window or not. FreeTrace then finds local maxima from the constructed hypothesis maps. To find the precise center-position of particles at sub-pixel level, FreeTrace performs 2D Gaussian regression by transforming it into a linear system. Finally, FreeTrace reconnects the detected particles by constructing a network and infer the most probable trajectories by calculating the reconnection-likelihoods on paths..

2 Methods

2.1 Extension of images

An extension of images is done by sampling of background noises following the 3-sigma rule of thumb iteratively,

$$-3\sigma + \overline{\boldsymbol{X}}_t < x_{t+1} < 3\sigma + \overline{\boldsymbol{X}}_t$$

where

$\overline{\boldsymbol{X}}_t$: mean of the window created at the edge of image at the iteration t

x_{t+1} : collected noise sample inside the window $\boldsymbol{X}_t$ and used to create a next sampling window $\boldsymbol{X}_{t+1}$

For example, if an image has 16x16 pixels and if we want to extend the image to 20x20, FreeTrace samples the noises by creating a window with size 16x2 at the left edge of the image, then it samples iteratively inside the window until it finds any noises breaking of 3-sigma rule. If the iteration stops, FreeTrace puts the 16x2 window at the left edge of the image. Above steps are done for every 4 edges of an image. Since FreeTrace uses a sliding window for the detection of particles, it cannot make a sliding window at the edges. However We can expect detection of particles at the edge of the image with extensions while avoiding a sampling of the particle's noise with the 3-sigma rule.

2.2 Detection of particle

FreeTrace creates a hypothesis map[1] for the detection of particles from an image by comparing two hypotheses with a sliding window. To construct the map, we find the ML of each parameter at first. The definitions of parameters are $\boldsymbol{Y}$: intensity vector of a sliding window, $\boldsymbol{N}$: Gaussian noise vector and I: Intensity factor of Gaussian PSF.

2.2.1 No particle inside a sliding window (H_0)

$$H_0 : \boldsymbol{Y} = m_0 + \boldsymbol{N}$$
$$L(H_0|m_0, \sigma_0^2) = -(w_s^2/2)\ln(2\pi\sigma_0^2) - (1/2\sigma_0^2)\Sigma(\boldsymbol{Y} - m_0)^2$$
$$\frac{\partial L_{H0}}{\partial \sigma_0^2} = 0 \Longrightarrow \hat{\sigma}_0^2(m_0) = (1/w_s^2)\ \sum(\boldsymbol{Y} - m_0)^2$$
$$\hat{m_0} = (\boldsymbol{Y}/w_s^2)$$
$$\hat{\sigma}_0^2(\hat{m}_0) = (1/w_s^2)\ \sum(\boldsymbol{Y} - \hat{m}_0)^2$$

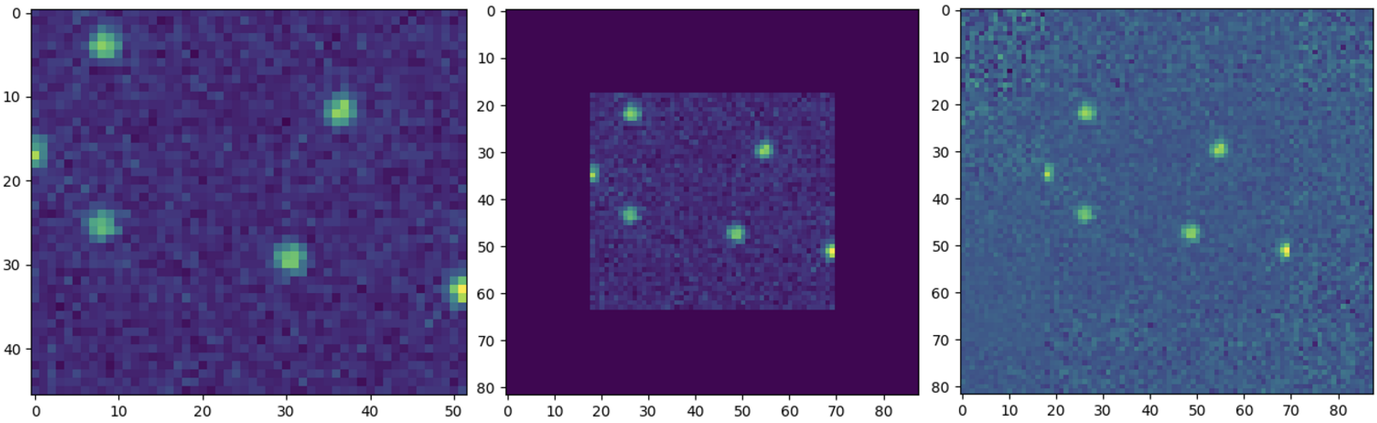

Figure 1: Left: raw image(46x52) before extension. Mid: extended without noise(82x88). Right: extended with noise(82x88). Extended images can use sliding windows to detect the particles at the edges of the image while the left image can't.

Then we calculate the H_1 with a given 2D standard gaussian PSF.

2.2.2 Presence of particle inside a sliding window(H_1)

$$H_1 : \boldsymbol{Y} = I\,\boldsymbol{G} + m_1 + \boldsymbol{N}$$
$$G_{i,j}(i_0, j_0, r_0) = 1/(\pi^{1/2} r_0)\exp(-((i-i_0)^2 + (j-j_0)^2)/2r_0^2)$$
$$L(H_1|m_1, \sigma_1^2, I) = -(w_s^2/2)\ln(2\pi\sigma_1^2) - (1/2\sigma_1^2)\Sigma(\boldsymbol{Y} - m_1 - I\,\boldsymbol{G})^2$$
$$\frac{\partial L_{H1}}{\partial \hat{\sigma}_1^2} = 0 \Longrightarrow \hat{\sigma}_1^2 = (1/w_s^2)\Sigma(\boldsymbol{Y} - m_1 - I\,\boldsymbol{G})^2$$
$$\frac{\partial \hat{\sigma}_1^2}{\partial m_1} = 0 \Longrightarrow \hat{m}_1 = (1/w_s^2)\Sigma(\boldsymbol{Y} - I\,\boldsymbol{G})$$
$$\hat{\sigma}_1^2 = (1/w_s^2)\Sigma(\boldsymbol{Y} - (1/w_s^2)\,\boldsymbol{Y} - I(\boldsymbol{G} - (1/w_s^2)\,\boldsymbol{G}))^2$$
$$\hat{\sigma}_1^2 = (1/w_s^2)\Sigma(\overline{\boldsymbol{Y}} - I\overline{\boldsymbol{G}})^2$$
$$\frac{\partial \hat{\sigma}_1^2}{\partial I} = 0 = \Sigma(\overline{\boldsymbol{Y}} - I\overline{\boldsymbol{G}})\overline{\boldsymbol{G}}$$
$$\hat{I} = \max(0, \Sigma\overline{\boldsymbol{Y}}\ \overline{\boldsymbol{G}}/\Sigma\overline{\boldsymbol{G}}^2)$$
$$\hat{\sigma}_1^2 = (1/w_s^2)\Sigma(\overline{\boldsymbol{Y}} - \max(0, \Sigma\overline{\boldsymbol{Y}}\ \overline{\boldsymbol{G}}/\Sigma\overline{\boldsymbol{G}}^2))\overline{\boldsymbol{G}})^2$$

We finally create a hypothesis map to determine the existence of particles with a threshold.

2.2.3 Hypothesis map test(H_0/H_1)

$$L(H_0|\hat{m}_0, \hat{\sigma}_0^2) = -(w_s^2/2)\ln(2\pi\hat{\sigma}_0^2) - w_s^2/2$$
$$L(H_1|\hat{m}_1, \hat{\sigma}_1^2, \hat{I}) = -(w_s^2/2)\ln(2\pi\hat{\sigma}_1^2) - w_s^2/2$$
$$H_{map} = \ln(L(H_0)) - \ln(L(H_1))$$
$$H_{map} = -(w_s^2/2)\ \ln(\hat{\sigma}_0^2/\hat{\sigma}_1^2)$$

2.3 Gaussian regression to estimate the center of particle at sub-pixel level

From calculated H_{map}, FreeTrace does a Gaussian regression to estimate the center of a particle at sub-pixel level. Since each pixel is 160x160nm^2 in super-resolution images in general, this step provides a more precise position inside the pixel. Guo[2] suggested an simple algorithm to solve the gaussian fitting in 1D. FreeTrace applied his algorithm for Bivariate normal distribution to estimate also the correlation and the standard deviation of x, y which can be used for an approximation of a particle's center in 3 dimensions for astigmatism.

2.3.1 Bivariate gaussian regression of detected particle

$$(\hat{I}, \hat{\rho}, \hat{\sigma}_x, \hat{\sigma}_y, \hat{x}_0, \hat{y}_0) = argmin_{(I,\rho,\sigma_x,\sigma_y,x_0,y_0)}\Sigma(\hat{\boldsymbol{Y}} - (\boldsymbol{Y} - \hat{m}_0))^2$$

Above system can be transformed into a weighted linear least square.
Then, the residual δ becomes

$$\delta = \Sigma(\hat{\boldsymbol{Y}}\ (\ln(\hat{\boldsymbol{Y}}) - \ln(G(\hat{I}, \hat{\rho}, \hat{\sigma}_x, \hat{\sigma}_y, \hat{x}_0, \hat{y}_0))))$$
$$\delta = \Sigma(\hat{\boldsymbol{Y}}\ (\ln(\hat{\boldsymbol{Y}}) - (ax^2 + bx + cy^2 + dy + exy + f)))$$

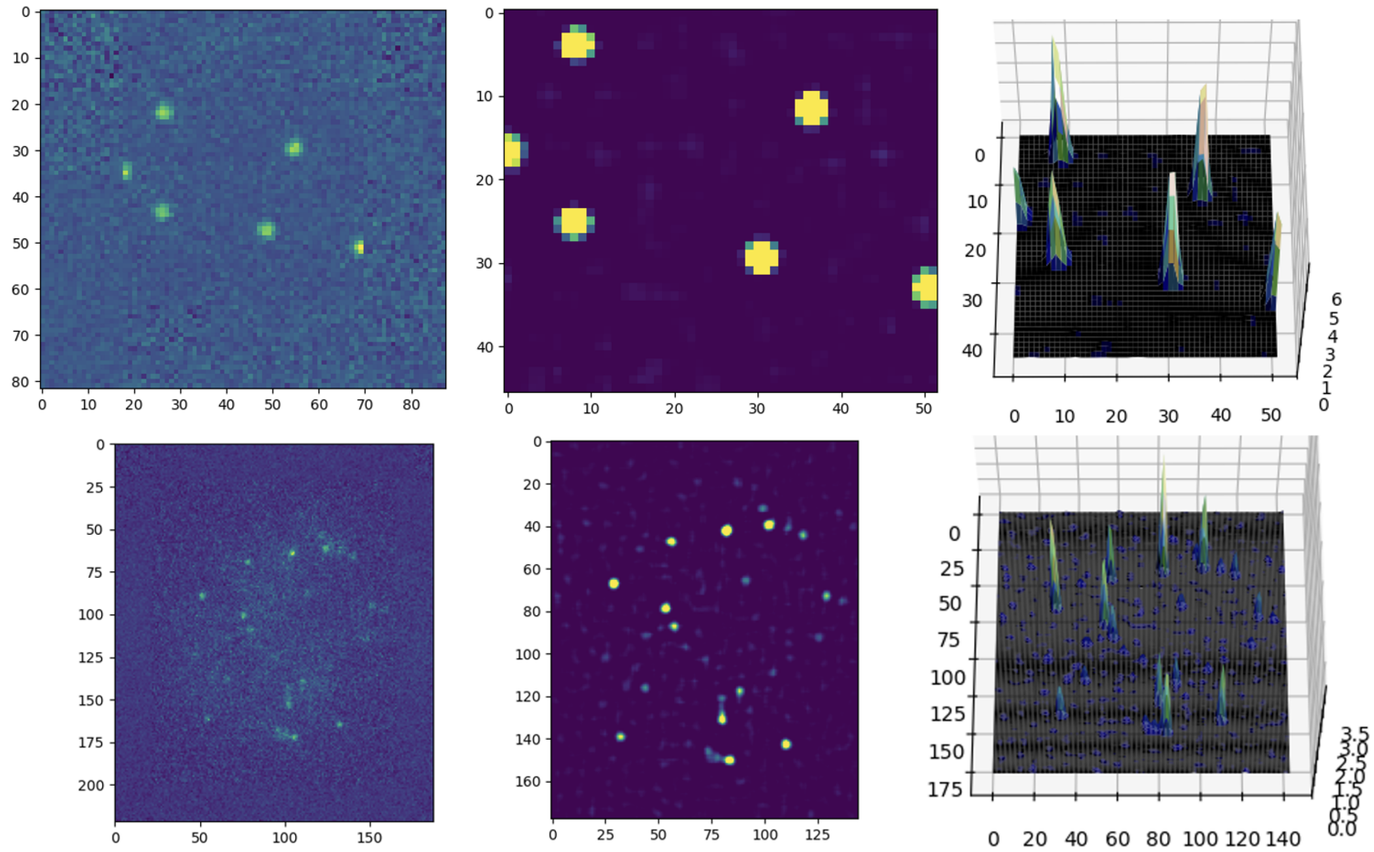

Figure 2: Visualised H_{map}. Left: extended images, Mid: 2D H_{map}, Right: 3D H_{map}. The 1st row images are from simulated data and the 2nd row images are from real data.

$$
\begin{gathered}
\text{where } \hat{\rho} = e\sqrt{(4ac)^{-1}} \\
\hat{\sigma}_x^2 = (-2a(1-\hat{\rho}^2))^{-1} \\
\hat{\sigma}_y^2 = (-2c(1-\hat{\rho}^2))^{-1} \\
\begin{bmatrix} -\hat{\rho}\frac{\hat{\sigma}_y}{\hat{\sigma}_x} & 1 \\ 1 & -\hat{\rho}\frac{\hat{\sigma}_x}{\hat{\sigma}_y} \end{bmatrix} \begin{bmatrix} \hat{\mu}_x \\ \hat{\mu}_y \end{bmatrix} = \begin{bmatrix} d(1-\hat{\rho}^2)\hat{\sigma}_y^2 \\ b(1-\hat{\rho}^2)\hat{\sigma}_x^2 \end{bmatrix} \\
\hat{I} = \exp\left(2\pi\hat{\sigma}_x\hat{\sigma}_y\sqrt{1-\hat{\rho}^2}\right)^{-1}
\end{gathered}
$$

Differentiate δ^2 with respect to a,b,c,d,e and f.

$$
\frac{\partial\delta^2}{\partial a} = 0 \Longrightarrow \Sigma\hat{\boldsymbol{Y}}^2(ax^4 + bx^3 + cx^2y^2 + dx^2y + ex^3y + fx^2) = \Sigma\hat{\boldsymbol{Y}}^2x^2\ln(\hat{\boldsymbol{Y}})
$$

.

.

.

2.3.2 Approximate unknowns of determined linear matrix equation

$$
\begin{bmatrix}
\Sigma\hat{\boldsymbol{Y}}^2x^4 & \Sigma\hat{\boldsymbol{Y}}^2x^3 & \Sigma\hat{\boldsymbol{Y}}^2x^2y^2 & \Sigma\hat{\boldsymbol{Y}}^2x^2y & \Sigma\hat{\boldsymbol{Y}}^2x^3y & \Sigma\hat{\boldsymbol{Y}}^2x^2 \\
\Sigma\hat{\boldsymbol{Y}}^2x^3 & \Sigma\hat{\boldsymbol{Y}}^2x^2 & \Sigma\hat{\boldsymbol{Y}}^2xy^2 & \Sigma\hat{\boldsymbol{Y}}^2xy & \Sigma\hat{\boldsymbol{Y}}^2x^2y & \Sigma\hat{\boldsymbol{Y}}^2x \\
\Sigma\hat{\boldsymbol{Y}}^2x^2y^2 & \Sigma\hat{\boldsymbol{Y}}^2xy^2 & \Sigma\hat{\boldsymbol{Y}}^2y^4 & \Sigma\hat{\boldsymbol{Y}}^2y^3 & \Sigma\hat{\boldsymbol{Y}}^2xy^3 & \Sigma\hat{\boldsymbol{Y}}^2y^2 \\
\Sigma\hat{\boldsymbol{Y}}^2x^2y & \Sigma\hat{\boldsymbol{Y}}^2xy & \Sigma\hat{\boldsymbol{Y}}^2y^3 & \Sigma\hat{\boldsymbol{Y}}^2y^2 & \Sigma\hat{\boldsymbol{Y}}^2xy^2 & \Sigma\hat{\boldsymbol{Y}}^2y \\
\Sigma\hat{\boldsymbol{Y}}^2x^3y & \Sigma\hat{\boldsymbol{Y}}^2x^2y & \Sigma\hat{\boldsymbol{Y}}^2xy^3 & \Sigma\hat{\boldsymbol{Y}}^2xy^2 & \Sigma\hat{\boldsymbol{Y}}^2x^2y^2 & \Sigma\hat{\boldsymbol{Y}}^2xy \\
\Sigma\hat{\boldsymbol{Y}}^2x^2 & \Sigma\hat{\boldsymbol{Y}}^2x & \Sigma\hat{\boldsymbol{Y}}^2y^2 & \Sigma\hat{\boldsymbol{Y}}^2y & \Sigma\hat{\boldsymbol{Y}}^2xy & \Sigma\hat{\boldsymbol{Y}}^2
\end{bmatrix}
\begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix}
=
\begin{bmatrix} x^2\Sigma\hat{\boldsymbol{Y}}^2\ln(\hat{\boldsymbol{Y}}) \\ x\Sigma\hat{\boldsymbol{Y}}^2\ln(\hat{\boldsymbol{Y}}) \\ y^2\Sigma\hat{\boldsymbol{Y}}^2\ln(\hat{\boldsymbol{Y}}) \\ y\Sigma\hat{\boldsymbol{Y}}^2\ln(\hat{\boldsymbol{Y}}) \\ xy\Sigma\hat{\boldsymbol{Y}}^2\ln(\hat{\boldsymbol{Y}}) \\ \Sigma\hat{\boldsymbol{Y}}^2\ln(\hat{\boldsymbol{Y}}) \end{bmatrix} \tag{1}
$$

From the above linear matrix equation, we can approximate the unknowns.
And iterate the system until the convergence.

$$\begin{bmatrix} \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^4 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^3 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2y^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2y & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^3y & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2 \\ \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^3 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2y & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x \\ \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2y^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}y^4 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}y^3 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy^3 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}y^2 \\ \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2y & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}y^3 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}y^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}y \\ \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^3y & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2y & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy^3 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2y^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy \\ \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}x & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}y^2 & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}y & \Sigma\hat{\boldsymbol{Y}}^2_{(k)}xy & \Sigma\hat{\boldsymbol{Y}}^2_{(k)} \end{bmatrix} \begin{bmatrix} a_{(k+1)} \\ b_{(k+1)} \\ c_{(k+1)} \\ d_{(k+1)} \\ e_{(k+1)} \\ f_{(k+1)} \end{bmatrix} = \begin{bmatrix} x^2\Sigma\hat{\boldsymbol{Y}}^2{}_{(k)}\ln(\hat{\boldsymbol{Y}})_{(0)} \\ x\Sigma\hat{\boldsymbol{Y}}^2{}_{(k)}\ln(\hat{\boldsymbol{Y}})_{(0)} \\ y^2\Sigma\hat{\boldsymbol{Y}}^2{}_{(k)}\ln(\hat{\boldsymbol{Y}})_{(0)} \\ y\Sigma\hat{\boldsymbol{Y}}^2{}_{(k)}\ln(\hat{\boldsymbol{Y}})_{(0)} \\ xy\Sigma\hat{\boldsymbol{Y}}^2{}_{(k)}\ln(\hat{\boldsymbol{Y}})_{(0)} \\ \Sigma\hat{\boldsymbol{Y}}^2{}_{(k)}\ln(\hat{\boldsymbol{Y}})_{(0)} \end{bmatrix} \tag{2}$$

$$\hat{\boldsymbol{Y}} = \hat{a}x^2 + \hat{b}x + \hat{c}y^2 + \hat{d}y + \hat{e}xy + \hat{f}$$
$$\hat{\boldsymbol{Y}}_{final} = G(\hat{I}, \hat{\rho}, \hat{\sigma}_x, \hat{\sigma}_y, \hat{x}_0, \hat{y}_0) + \hat{m}_0$$

Above linear system guarantees fast convergence of the system. The estimated $\hat{\rho}, \hat{\sigma}_x, \hat{\sigma}_y$ provide additional information such as motion blur and direction of fast-diffusive particle from image.

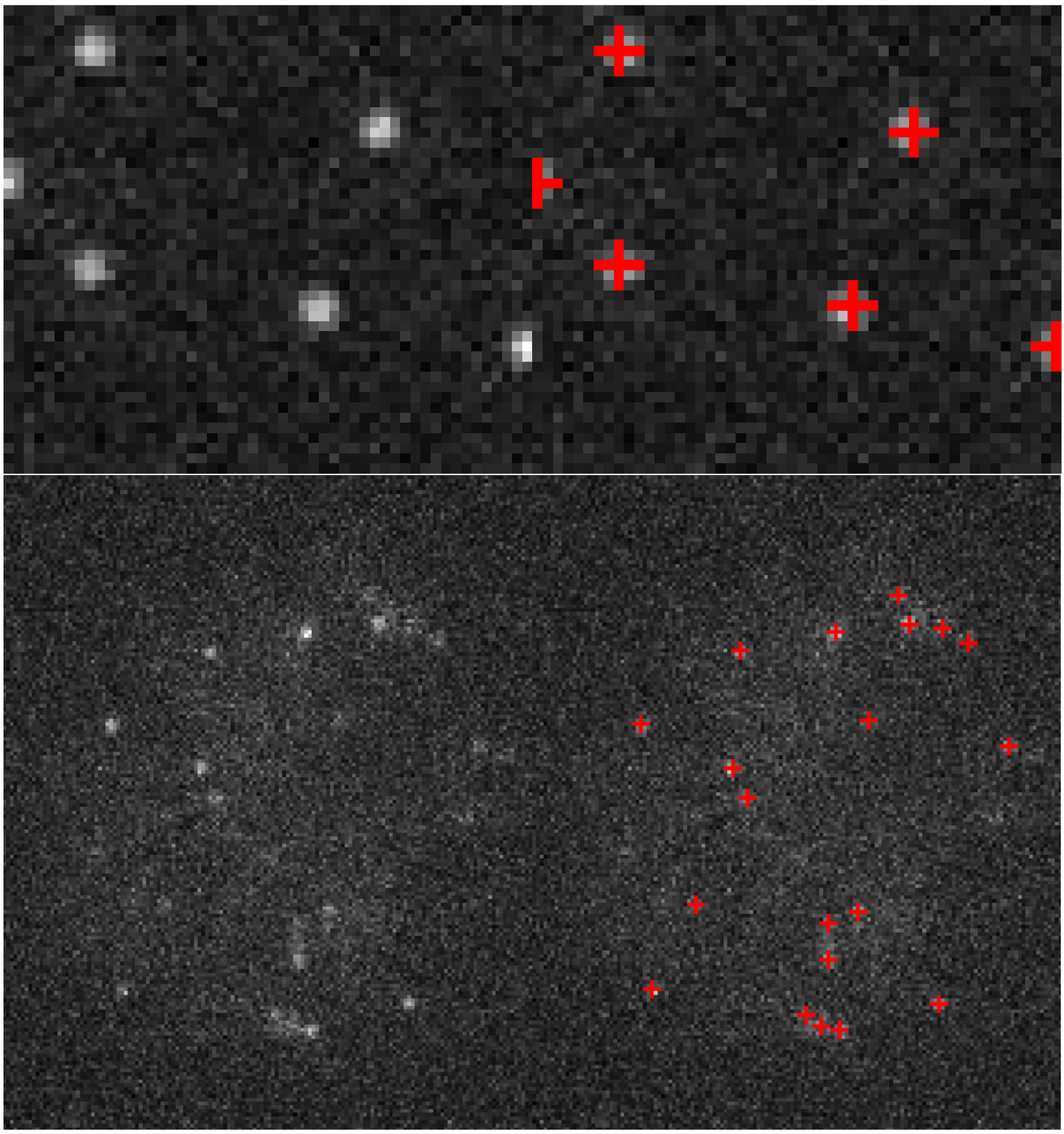

Figure 3: Localized particles. The 1st row images are from simulated data and the 2nd row images are from real data. Red cross indicates the center of a localized particle at sub-pixel accuracy.

2.4 Reconnection of detected particles

From localized particles, FreeTrace does reconnections of detected particles in time series. The reconnection is done by calculating the likelihood of the particle's jump distance and the likelihood of the particle's diffusive state with a construction of the DAG network. Below is the definitions of parameters.
$loc_t(A)$: localized position of particle A at frame t
$\boldsymbol{Euc}(loc_t(A),\ loc_{t+1}(B))$: euclidean distance between particle A at frame t and particle B at frame t+1

2.4.1 $L_{dist(l)}$ approximation

$$L_{dist(l)}(loc_{t+1}(B)|loc_t(A)) = \boldsymbol{PDF}_l(\frac{\boldsymbol{Euc}(loc_t(A),loc_{t+1}(B))}{\alpha})\ where\ \alpha = \left\lceil \frac{sign(\frac{\boldsymbol{CDF}_l^{-1}(0.995)}{\boldsymbol{Euc}(loc_t(A),loc_{t+1}(B))}-1)+1}{2} \right\rceil + 10^{-6}$$

$\boldsymbol{PDF}_l$ and $\boldsymbol{CDF}_l$ are approximated beforehand from an observed localization sample by pairing two closest particles between $l\Delta t$ frames. FreeTrace does a smoothing of $\boldsymbol{PDF}_l$ with Kernel density function of the scikit-learn package.

2.4.2 $L_{elongated}$ approximation with regressed covariance matrix

$$L_{elongated}(loc_{t+1}(B)|loc_t(A),\alpha) = -\frac{\max(\lambda_{(A)})-\min(\lambda_{(A)})}{\Sigma\lambda_{(A)}} \times (\min(\pi - \max(\theta_1,\theta_2), \max(\theta_1,\theta_2)) \times \frac{2}{\pi}) + \frac{\max(\lambda_{(A)})}{\Sigma\lambda_{(A)}}$$

where

$$\theta_1 = \cos^{-1}(\left\|\boldsymbol{v_A}(argmax(\lambda_{(A)}))\right\| \cdot \left\| loc_{t+1}(B) + \alpha\boldsymbol{v_B}(argmax(\lambda_{(B)}))\ \sqrt{|(\frac{\lambda_{1_{(B)}}}{\min(\lambda_{(B)})})^2 - (\frac{\lambda_{2_{(B)}}}{\min(\lambda_{(B)})})^2|} - loc_t(A) \right\|)$$

$$\theta_2 = \cos^{-1}(\left\|\boldsymbol{v_A}(argmax(\lambda_{(A)}))\right\| \cdot \left\| loc_{t+1}(B) - \alpha\boldsymbol{v_B}(argmax(\lambda_{(B)}))\ \sqrt{|(\frac{\lambda_{1_{(B)}}}{\min(\lambda_{(B)})})^2 - (\frac{\lambda_{2_{(B)}}}{\min(\lambda_{(B)})})^2|} - loc_t(A) \right\|)$$

$$\alpha = 1.0$$

2.4.3 $L_{directed}$ approximation

$$L_{directed}(loc_{t+1}(B)|loc_t(A),\sigma_{directed},\Delta t) = \exp(-\frac{1}{2}\frac{(loc_{t+1}(B)-loc_t(A)-(\Sigma_{k=t-\Delta t}^{t}\boldsymbol{v}_k)/\Delta t)^2}{\sigma^2_{directed}}) \text{ if } t > \Delta t$$

$$L_{directed}(loc_{t+1}(B)|loc_t(A)) = \exp(-\frac{1}{2}\frac{(loc_{t+1}(B)-loc_t(A))^2}{\sigma^2_{directed}}) \text{ else}$$

$$\text{where } \sigma_{directed} = 5.0$$

$$\Delta t = 3 \text{ if } t > \Delta t$$

By summing up the three log-likelihoods, FreeTrace reconnects the particle A at the frame t to the particle B at the frame $t+1$ which has the highest log-likelihood.

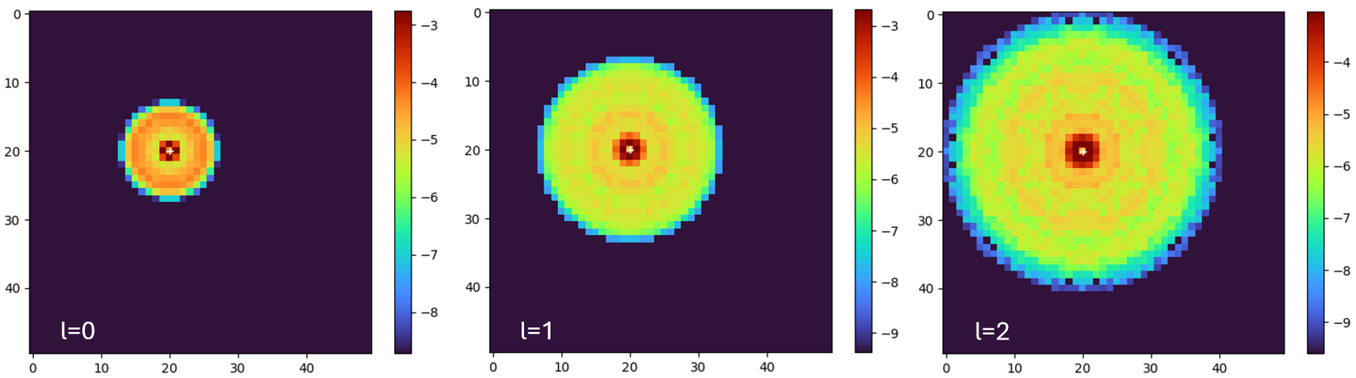

Figure 4: Visualization of log-$L_{dist(l)}$ space for each l. l is a waiting time of the current particle of frame t to reconnect the particle at $l+t+1$. White crosses indicate the current positions of particles.

3 FreeTrace result

With FreeTrace, we can get the traces of particles from video which can be further used to analyze the biophysical properties of particles. The visualized examples of results are in the Fig.7

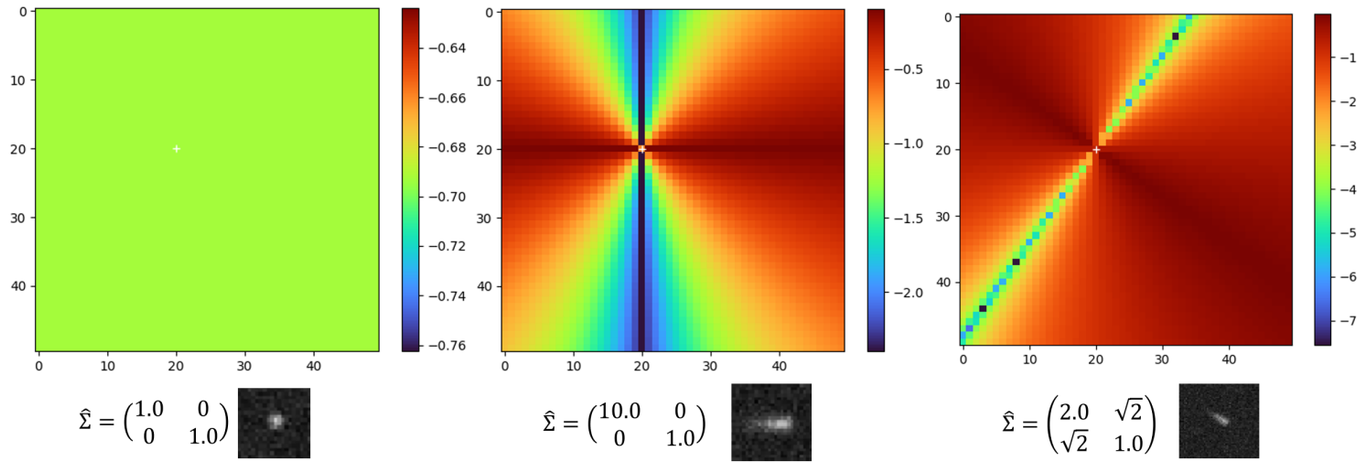

Figure 5: Visualization of log-$L_{elongated}$ space for different regressed covariance matrix which is done during the localization. Small images show the particle's elongated shape with regressed covariance matrices. White crosses indicate the current positions of particles.

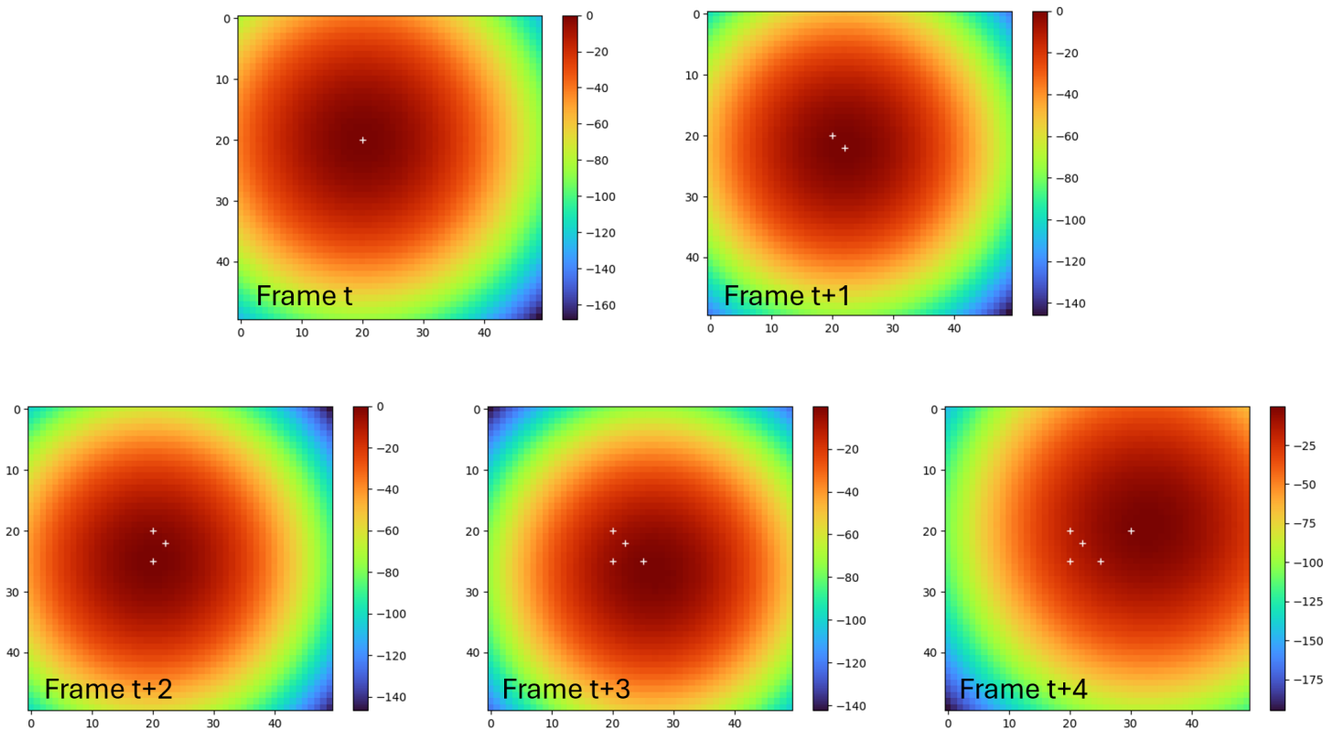

Figure 6: Visualization of log-$L_{directed}$ space for 5 successive frames. White crosses indicate the positions of the same particle. White crosses are accumulated at each frame to trace the movement of the particle. The position of maximum log-$L_{directed}$ changes when the molecule moves.

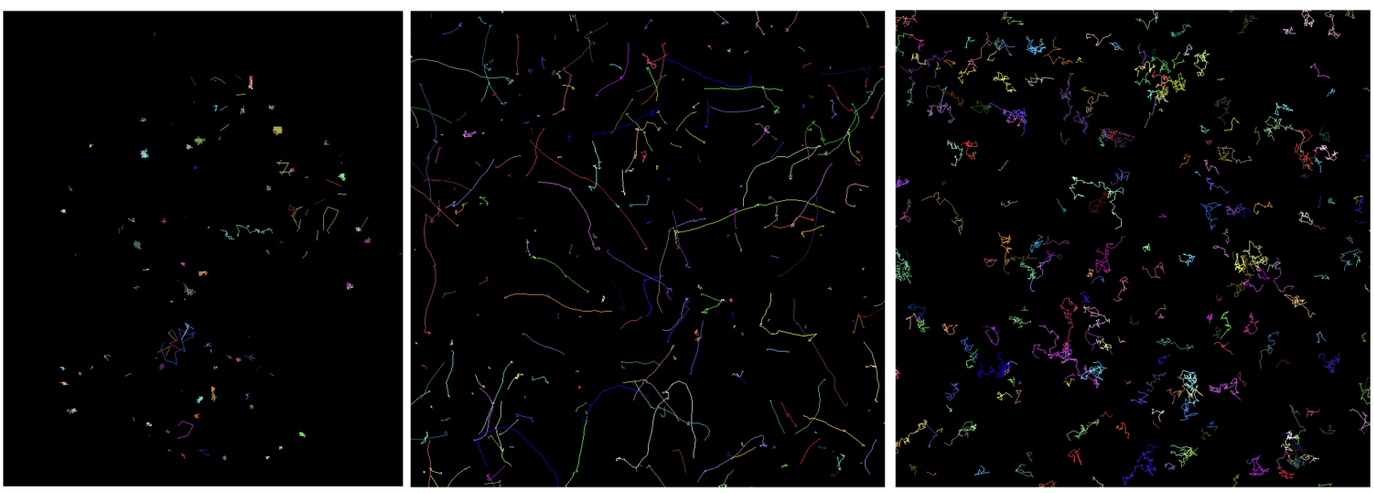

Figure 7: Visualized results of three image-sequences, localized and tracked by FreeTrace. Each color corresponds to the individual traces.

References

[1] Arnauld Sergé et al. “Dynamic multiple-target tracing probes spatiotemporal cartography of cell membranes.” In: (2008). DOI: 10.1038/nmeth.1233.

[2] Hongwei Guo. “A Simple Algorithm for Fitting a Gaussian Function”. In: *IEEE Signal Processing Magazine* 28 (Sept. 2011), pp. 134–137. DOI: 10.1002/9781118316948.ch31.