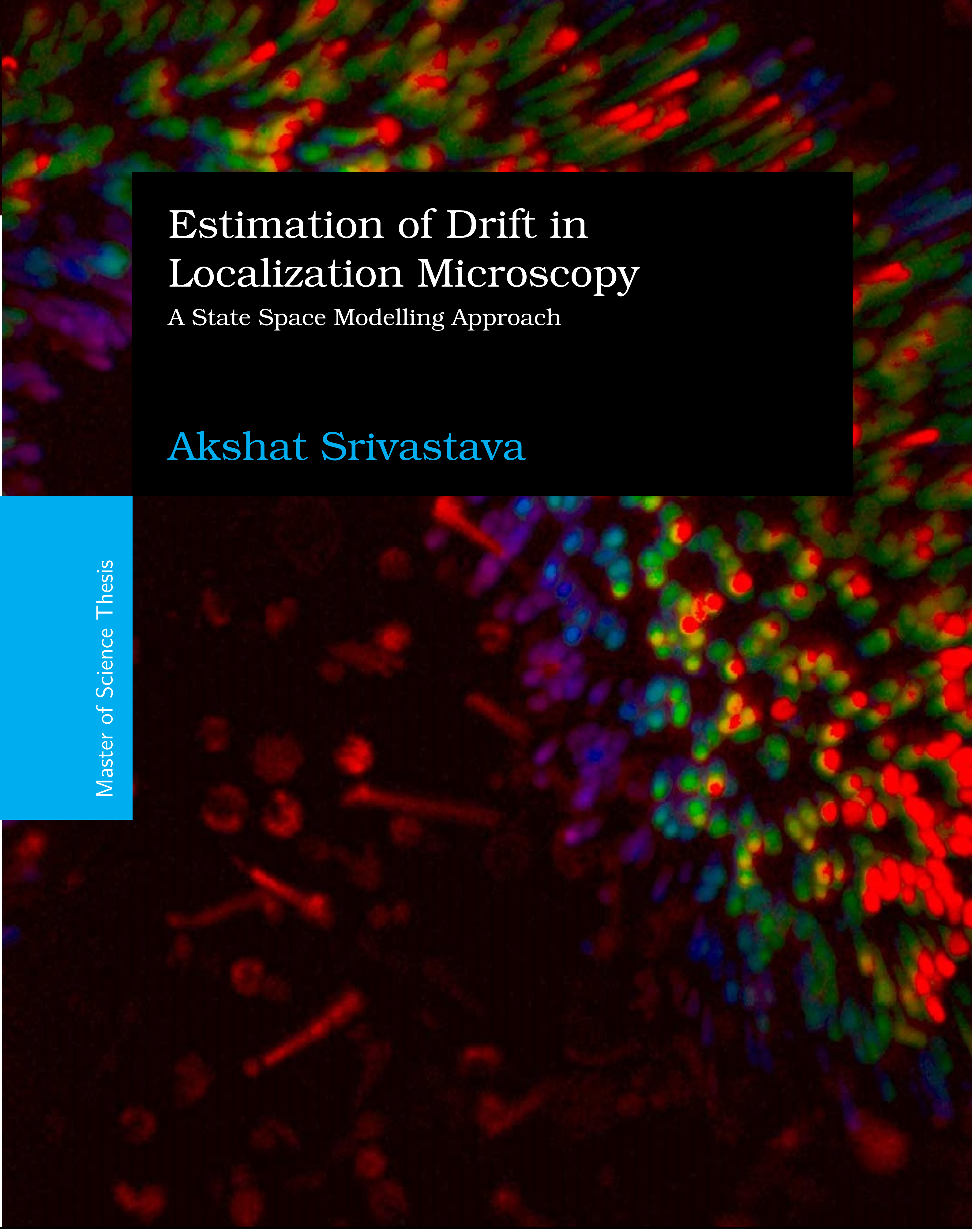

Estimation of Drift in Localization Microscopy

A State Space Modelling Approach

Akshat Srivastava

Master of Science Thesis

Delft University of Technology

Delft Center for Systems and Control

Estimation of Drift in Localization Microscopy
A State Space Modelling Approach

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Akshat Srivastava

February 14, 2022

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

Single Molecule Localization Microscopy (SMLM) has enabled researchers to breakthrough the diffraction limit and obtain nanometer resolution images of macromolecular structures. But due to the time involved in obtaining ample data for proper image, the technique is venerable to many problem including fluctuations due to thermal gradients from surrounding which cause the frames to drift. SMLM relies on the stochastic blinking of fluorophore probes. Thus drift in SMLM could be explicitly modelled as a stochastic state space process. These models could be used to perform drift correction.

Two state space models are proposed relying on different properties of SMLM. The first model utilizes shifting of underlying image structure. The state space model for this property is constructed using shift matrices. A system identification method along with image reconstruction method is also derived to form the drift compensation algorithm for this model. This algorithm is further developed to provide adequate performance within low computational time. The second model relies on the position of emitter molecules and utilizes *linking* or pairing of fluorophore probes in succeeding frame to obtain the output data. Drift compensation algorithm for this model is constructed using Prediction Error Methods (PEM) and Kalman (RTS) smoother.

The drift correction algorithm for these two models are also bench-marked with existing algorithms to obtain insight into performance. Furthermore, other properties of these algorithms are explored using simulation dataset and recommendation are provided for improvement and further research.

Table of Contents

Acknowledgements

Dr. C. Smith, I am more than grateful for your constant support, guidance and understanding throughout the length of this project. It has been steep learning in not just studies, but also project development and work-life balance. Without your motivation on pursuing a problem further and further, this thesis work would not have taken the shape that it has. I am truly thankful.

Prof. Dr.ir. M.H.G. Verhaegen, it has been a pleasure to be guided by you and nothing less than a privilege. I always dreamt of working with or under a man who holds the torch of scientific & mathematical endeavours, and I feel like this is as close as I will ever get. It has been wonderful experience to exchange ideas with you and other members of N4CI.

I would like to thank the rest of my thesis committee, Dr. M. Kok for taking the time to read through this whole report and reviewing it, for assessing my presentation and the work as a whole. It is very much appreciated.

Also, thank you Jelmer Cnossen for your unending support whenever I have been stuck, and also helping me get back up whenever I got hit hard by hurdles. I would also like to thank all members of qnano group for feedback and insights.

Finally and also importantly, I want to thank my parents and brother for their support and advises throughout this phase. I would also like to thank my friends, specially Jyotsna, Ayushi and Guilherme, for all their support in completing this project as well.

Delft, University of Technology
February 14, 2022

Akshat Srivastava

“...If you can meet with Triumph and Disaster,
And treat those two impostors just the same...”

— *If by Rudyard Kipling*

Chapter 1

Introduction

1-1 Diffraction Limit

Ever since the invention of microscope in 1590 by two Dutch spectacle-makers, Hans and Zacharias Janssen, it has greatly empowered the advancements in biotechnology and biomedical research. With the development in technology, the resolution of the image captured by the microscope increased as well. But there is a inherent limit to this resolution. *Huygens wavelet theory* shows that light is a wave, and thus interferes with other light waves to create constructive and destructive patterns. As the light wavefronts pass through the circular aperture, diffraction occurs which limits the optical resolution in application [15]. This so called *diffraction limit* restricts the ability of optical instruments like microscopy to distinguish between two light emitting particles lying at close distance to each other. Figure 1-1 shows the image of Point Source Function (PSF) of two particles at various distance.

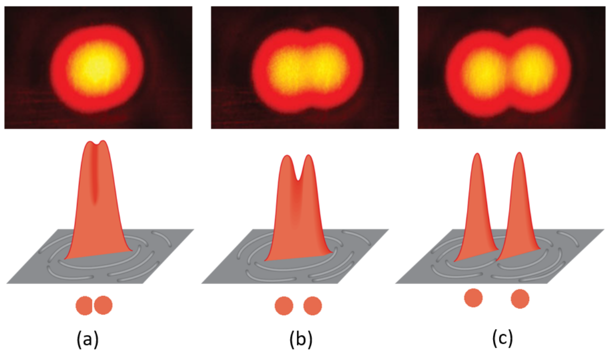

Figure 1-1: (a) Lateral separation between two light source is too small for proper resolution, (b) PSF are marginally distinguishable, and (c) Clearly distinguishable PSF.

Abbe's diffraction limit shown in (1-1) governs the resolution in optical microscopy.

$$d = \frac{\lambda}{2NA} \tag{1-1}$$

where d is the resolution, λ is the wavelength of light used to capture the image and NA is the numerical aperture of lens. Multiple methods have been developed to surpass the diffraction limit such as electron microscopy, atomic force microscopy and Super Resolution Microscopy (SRM). Under the umbrealla of SRM, there are multiple techniques as well for e.g. Scanning Microscopy like ISM, Structures Illumination Microscopy (STED) and Single Molecule Localization Microscopy (SMLM). In this thesis, our focus lies in SMLM.

1-2 Single Molecule Localization Microscopy

SMLM relies on fluorescent markers and fluorescent protein expression. These markers are localized from diffraction limited image sequence and then the sample structure is identified by combining all the localization from the blinking markers. The imaging techniques falling under SMLM capture resolution of upto 10nm with the utilization of two objective lens [27, 24]. One of the early methods of performing SMLM, Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM) focuses on repetitive blinking fluorophore markers attached to certain proteins [20]. Other methodologies also exist, such as Photoactivated Localization Microscopy (PALM) or fluorescence photoactivation localization microscopy (FPALM) [14] and Fluorescence Imaging with One-nanometer Accuracy (FIONA) [28] which relies on stochastic activation of sparse sets of individual fluorophores and subsequent determination of their positions. Furthermore, it is also possible to extract information about the sample in 3-dimension as well. 3D localization methods utilize additional optical instrument for e.g. cylindrical lens insert in the detection path or multifocus configuration to localize the 3D position [17].

1-3 Drift

SMLM requires a large sequence of diffraction limited frames (typically $>10^4$). This in turn demands a large acquisition time through which a number of practical problems arise e.g. elongated photostability, photobleaching and complex preparation. One of the major problem is drift. Equipment such as excitation laser could produce enough thermal gradients to cause the sample to fluctuate. It could produce enough movement to move the sample by few nanometers. This makes it difficult to determine the absolute position of the markers and therefore the molecular density of the sample [19].

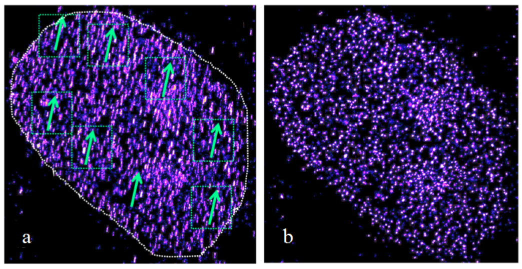

Figure 1-2: (a) The effect of drift in a single time step on localization of emitters (b) Image showing localization after drift correction

Many methods have been proposed to solve this problem. This includes modifying the experimental setup and using advance equipment, known as *active drift compensation* techniques. On the other hand, we can correct the drift after the low resolution frames are recorded, known as *passive drift compensation.* Each method carries a trade-off between costs involved, feasibility, required precision, time and ease of use. Thus which method to utilize depends on the choice of latter factors.

1-4 Thesis Layout

In this thesis study, we explore passive drift correction using methodologies of system and control. In Chapter 2 we discuss in brief existing methodologies. Different techniques, their drawbacks, comparative advantages and performance results are discussed therein. Chapter 3 discusses the state space model based on the shift matrices and molecular density. The model is developed for simple 1D case and then 2D drift in SMLM. Using this model, a drift compensation algorithm is also developed using the tools from system identification and filtering, and further enhanced using tools from image processing and EM algorithm. Chapter 4 follows a similar approach where we develop state space model using the position of emitter molecules. This model is also used to create a drift compensation algorithm. The two drift compensation algorithms developed are benchmarked against the existing algorithms using computational time and RMSE for performance in Chapter 5. We also investigate and discuss insights about the two algorithms, the pros and cons of each algorithm as well as when and why algorithms break down. Chapter 6 ends with conclusion to this thesis study and further developments that could be performed for each algorithm.

In the following chapters, certain terms are used interchangeably. This includes markers which is same as probes and emitters. Often the word *molecule* is also included. State space models is also interchanged with hidden markov models, although the former is a subset of the latter. Similarly, the emitter density is synonymous with molecular density. Furthermore, it is important to understand the meaning between two different terms for e.g. relative drift and absolute drift, molecular density, frame density and molecular frequency, etc. Smoothing is also used in the context of stochastic signal processing as well as image processing. Depending on the context, the word carries different meaning. Such terms are repeatedly mentioned and might cause confusion.

Chapter 2

Drift Correction methods in SMLM

Drift correction or compensation in the context of SMLM refers to removing drift from low resolution frames captured in the imaging process. There are variety of ways to accomplish this. For ease of understanding, we classify the ways in categories shown in Figure 2-1. These categories primarily consist of Active compensation technique and Passive compensation techniques. The former relies on correcting drift in real time (actively) with feedback control system consisting of actuators and sensor whereas the latter focuses on correcting drift after the frame are captured (passively).

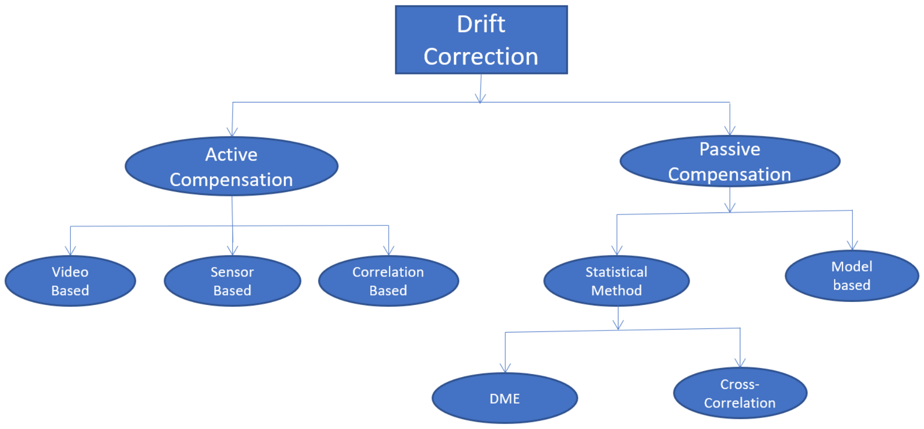

Figure 2-1: Different categories of drift compensation techniques

These techniques are discussed in the following sections. Mathematical details and in-depth results could be found in literature. It is often noticed that due to the fluctuation in the axial direction, the emitters concentrate or blur out [17]. Without correction, the peak signal intensity drops. This would degrade the localization of emitters and the drift correction in the lateral direction. Thus, for axial drift correction, it is required to have separate focus hardware [18] to perform a primary correction in axial axis. Other methods also include correcting axial drift manually by adjusting the stage.

2-1 Active Compensation

This compensation methods removes drift from the frames simultaneously as they are being captured. It uses a closed loop system of lead zirconate titanate (PZT) stage actuator for precise displacement control (but at low frequency) and sensors to detect the drift. The sample is kept on the stage and once the drift is detected, the actuators move in the opposite direction of drift to compensate it. An instrumental component in the configuration is *fiducial* markers. These markers, unlike the fluorophore probe emitter do not blink, but are visible through most of the experiment. Different fiducial markers are available along with their respective pros and cons. Some examples include Gold nanoparticle [30], Nano-patterning of coverslip [17], Fluorescent micro-particles [1] and Nanodiamonds [5]. Ideal properties of fiducial markers include stationarity, monodispersion to avoid aggregation, photostability and non-fluctuating signal. Figure 2-2 shows few types of markers.

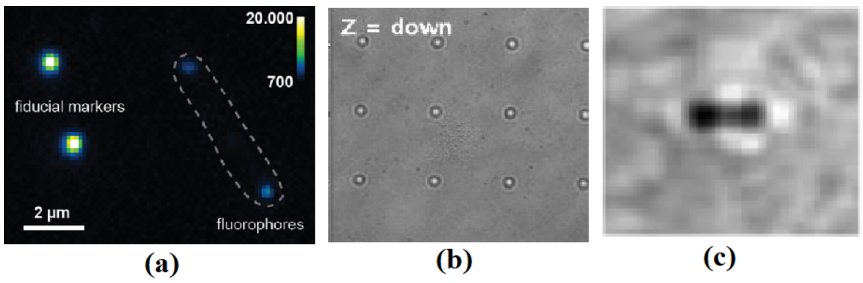

Figure 2-2: Examples of few types of Fiducial markers including (a) Fluorescent micro-particles, (b) Nano-patterning of coverslip and (c) Gold nanoparticle in bright field imaging

Figure 2-3 shows schematic of setup used for active drift correction. Although the actuator remains almost the same in all the setup, what differs is the type of sensor used. These include video camera, which is also used to perform bright-field imaging. A sample image is shown in Figure 2-2(c). IR led and detectors are also used alongside camera to detect fiducial markers as in 2-2(b). The correction is performed by measuring the diameter of the diffraction ring around fiduciary marker, along with known incremental displacement. It is observed that the diameter is proportional to the axial displacement and focus of ∼20nm is achieved. For the lateral calibration and correction, the centers of fiduciary marker are fitted with a gaussian function to identify the centers. The movement of these centers in each frame correspond to the drift which is corrected using the actuators.

Some setups also use laser along with Quadrant Photodiode (QPD) sensor to keep the stage stable [23]. Furthermore, there is an active correlation compensation technique which removes drift using frame cross-correlations. The principle is to track the peak position of the correlation function, which is used to determine the lateral x-y drift in the image plane. The drift in axial plane is evaluated by the change in intensity. The drawback of this method is low frequency of response. Although the CCD camera has a frame rate of 30 Hz, but due to intermediate processes involved such as image correlation, Gaussian fitting and taking the drift adjustment through motor movement into account, the actual response rate is around 7.5 Hz. Improvements could include faster responding xyz piezo stage and faster IR camera. Furthermore, while carrying out image correlation, the contrast of the image plays an important role. The contrast utilized to select specific part of the bright-field image is sometimes low. Without correlation feedback, the average coordinate drift of fiducial marker is more than 100nm in a period of 500s. Through feedback, the stage is locked within a Root Mean Squared Error (RMSE) position error of about 10 nm in the xy plane and about 20 nm in the z direction over a time period of 300 seconds.

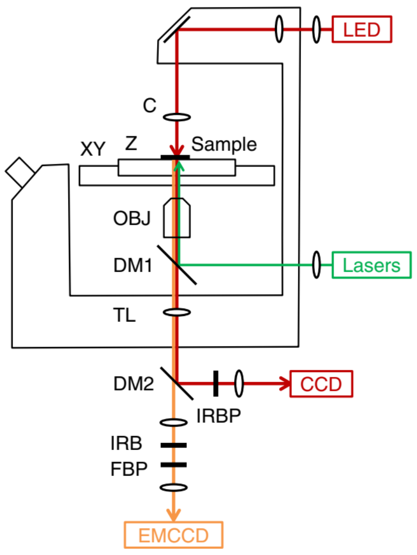

Figure 2-3: Drift Rate correction Setup, C: Condenser, Z: Piezo z stage, XY: motorized xy stage, OBJ: objective, DM1, DM2: dichroic mirrors, TL: tube lens, IRBP: IR band-pass filter, IRB: IR blocking filter, FBP: fluorescence band-pass filter [19]

2-2 Passive Compensation

The whole process of utilizing fiduciary markers, specially for lateral drift correction almost always leads to added complexity. None of them are easily synthesized and require additional equipment as well [21]. Additional camera and marker synthesis raises the costs, escalated higher by a requirement of substantial optics expertise. Passive (or Posterior) drift correction methods operate on frames after the process of capturing them is completed. These methods utilizes the emitters themselves to determine the drift either through statistical method or using model based approach such as drift at minimum entropy DME or Bayesian Sample Drift Inference (BaSDI). For bench-marking purposes, these methods could be tested on simulated and synthesized data as well which allows application of various methodologies.

Drift could be estimated from one frame to another, which is called relative drift or from initial frame to any other frame, which is called absolute drift. Some algorithms estimate the former, while other estimate the latter.

2-2-1 Cross-Correlation

Cross-correlation has been used in energy filtered transmission electron microscopy [13], atomic force microscopy, and scanning tunneling microscopy [22] to correct for sample drift, where subsequent images show the same structure, but differ primarily by their spatial position. The method therefore finds abundant application in SMLM as well. These algorithms assume that drift is a continuous process but not linear over the course of measurement. Furthermore drift occurring in a single camera frame (the time elapsed while capturing an image) is assumed to be negligible and uncorrelated with other spatial coordinates.

There are three different methods which makes use of correlation, either in different manner, or by using some empirical methodology (such as elimination of redundant dataset). Underlying these algorithm lies some fundamental operation required to properly execute the algorithms briefly discussed as follows:

1. Aggregation: Frames within a time interval T are aggregated. This interval is such that it is long enough to assume drift within each time step is linear but also small enough to capture enough particles (~ 1000). The time step, lets say 5s would capture some particles. This is necessary since in a single frame, one could only expect a small number of emitters. The size of the aggregates, m (a.k.a segments [29]) needs to be selected for proper Signal to Noise Ratio (SNR). When m is small, there are few localization points. This leads to noisy correlation function which could contain false maxima Figure 2-4(d). On the other hand, taking m large leads to an inevitable linear interpolation of drift calculation which in turn reduces the accuracy of the correction (even though high SNR is obtained Figure 2-4(c)).

   - Segment S_j of image frames is defined as the collective set of coordinates of emitters from m frames. $S_j = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathbb{R}^3$ is the position of ith fluorophore and n is the total number of emitters in segment j. Notice that n differs for each segment.
   - Drift in segment j is denoted by $\mathbf{d}_j$. The resulting observed segments S_j consist of, $S_j = \{x_1 + d_j, x_2 + d_j, \ldots, x_n + d_j\}$.

2. Projection: For each aggregated data frame, the xyz projection are made by forming three 2D planes (xy,yz, and xz). The particles are also binned into pixels (histograms) which represent the instantaneous molecular density. The bin size is in pixel dimension, which is small enough to capture the details (for e.g. importance for separation of one fluorephore point from another) and large enough to have apt number of localization points. The choice of the bin size, dictated by the Nyquist Shannon sampling theorem as twice finer than the standard deviation of minimum effective structure of interest. The effective minimum structure is in turn a function of marker distribution's standard deviation and the localization precision [10].

3. Evaluation: Cross-correlation (CC) [29] is evaluated, $C_{ij}(\mathbf{r})$ between two segments, S_i and S_j. The argument $\mathbf{r}_{ij}$ which maximizes the value of cross-correlation reflects how much drift has occurred.

$$\boldsymbol{D}_{ij} = \boldsymbol{r}_{ij}, \text{ where } \boldsymbol{r}_{ij} \text{ maximizes } C_{ij}(\boldsymbol{r}) \tag{2-1}$$

There are three methods through which correlation (or evaluation step) could be performed. Furthermore, a cubic spline is optionally fit to the resulting curves for noise reduction. Drift within each time interval is then determined by linear interpolation [21]. The first most basic

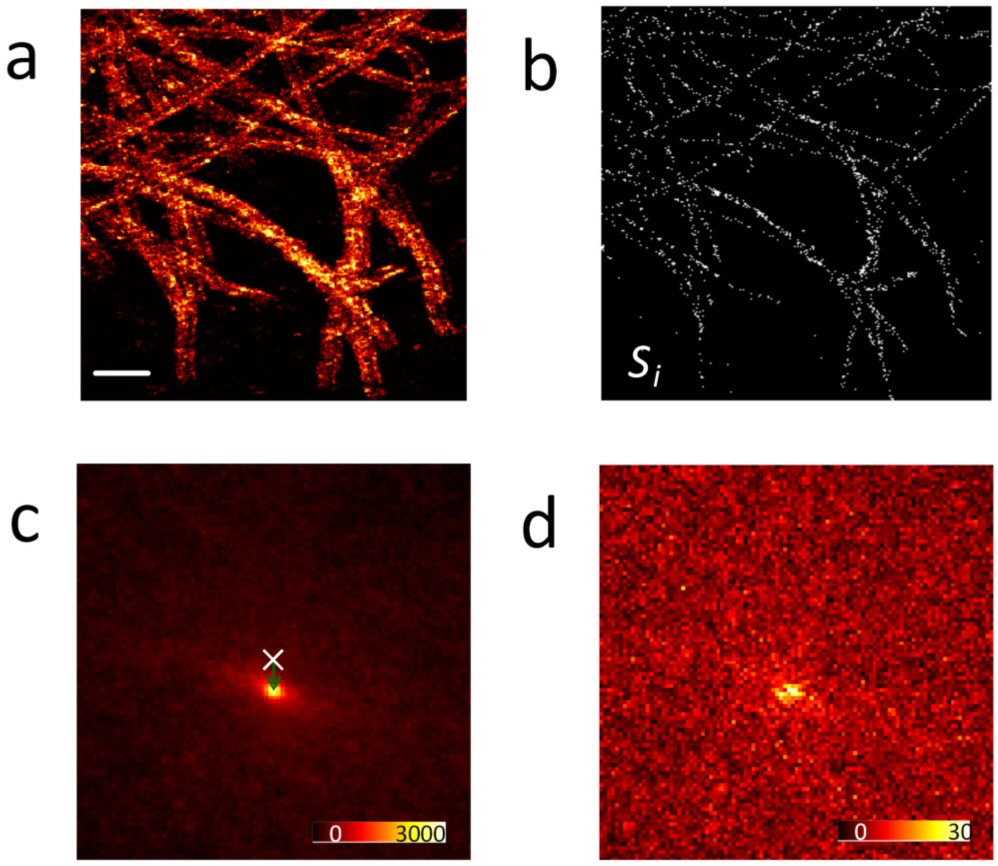

Figure 2-4: The effect of aggregate on the correlation (a) A superresolution image of microtubules that contains severe sample drift. The image is spatially binned into a 2D histogram with a bin size of 30 nm. Scale bar: 1 μm. (b) One time segment with f = 1000 frames. (c) The cross-correlation function between the first and the last segments with f = 1000 frames. The white cross shows the auto-correlation peak of the first image. The arrow points out the direction and the amount of the drift between these two intervals. (d) The cross-correlation function between the first and the last segments with f = 100 frames. Note that the SNR in the map is greatly decreased [29]

method is called Direct Cross Correlation (DCC). The initial segment is correlated with the current segment and the drift corresponding to the maximum value is evaluated. Mean Cross Correlation (MCC) evaluates the drift as a solution to LLSQ problem. It considers the correlation of any segment i with each and every other N frames. Redundant cross-correlation (RCC) is the most widely used correlation method for removing drift. This drift correction algorithm exploits the redundancy among the correlation pairs. RCC is quite similar to MCC. The difference between MCC and RCC lies within forming an over-determined set of linear equations solved for the value of drift.

These methods are benchmarked using both simulated and synthesized data-sets are utilized. For example, Figure 2-5(a) shows simulated structure of mitochondria generated using a custom written program in LabVIEW, consisting of about 80,000 particles. The structure is contaminated with a Zero Mean White Noise (ZMWN) and also injected with a known sinusoidal drift in time to obtain Figure 2-5(f). It should be noticed that this is just a single plane projection as indicated in the Figure. Other projections, different structures along with the most simplest correlation technique, DCC, are discussed in [21]. Furthermore, open training data is also available at [6].

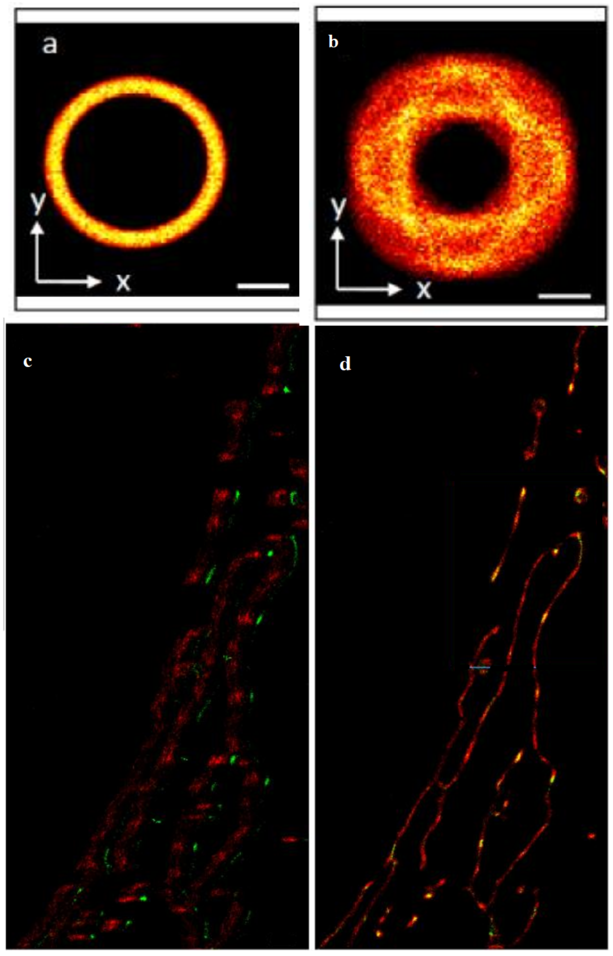

Figure 2-5: (a)Simulated structure of "mitochondria" in x-y plane (b)Same structure contaminated with drift (c) Aggregated localized image of mitochondria with drift from two color channels (see [23] for two color localization measurement) (d) Drift corrected image of mitochondria using posterior correction method [21]

One of the determining factor for posterior drift correction is also the sample structure. For example, consider a structure with it's longest length oriented towards either of the lateral axis. Determining the drift in in the longest length will be difficult since no matter how much sample drift occurs in that axis, it will look nearly identical. As a rule of thumb, only molecules that are within a distance a from an object "edge" (a structural feature indicating a strong change in density) contribute to detection of drif. This phenomenon has to be considered when choosing the Region of Interest (ROI) for the projection.

The drift correction algorithm was implemented (in literature) on the experimental data which contains drift of about 150nm in x-direction and 100nm in the y-direction. The spatial bin size of the drift extracted was 15nm. To quantify the results obtained on the experimental data, RMS error was used which yielded 1.56nm. Analysis of real data yielded an image FRC resolution in the range of 47.8nm. The precision of this method increases with increase in number of frames. The best image resolution attained through this method is 45.7nm.

Other important parameters include the number of molecules in each time interval and the localization precision. Many articles discuss and state that drift correction using cross-correlation is able to remove drift of several nanometers below 5nm with only 2,000 localized molecules in each interval. Hitherto, correlation based methods are the most frequently used in SRM due to their speed and accessibility.

2-2-2 Drift at Minimum Entropy

An alternative is to parameterize the drift as a specialized cubic spline and utilize the probability density formed as a measurement of entropy. This so called "entropy" arises from the information theory and machine learning but in a slightly different context. The drift is estimated by minimizing an upper bound on this entropy. DME run-time is on the same order of magnitude as correlation methods. Furthermore, the method could also be tuned to suit the computational time in case a large but redundant (i.e. low number of bright spots) dataset is at disposal. DME outperforms both, model based and correlation based methods. It is able to produce a more detailed drift trace without consuming as much time as BaSDI. Furthermore, when localization within one bin becomes a limiting factor for correlation based methods (sparser dataset), DME still outperforms other methods. For a 250 frame size, correlation method produce mean absolute error of 8nm whereas DME produces only 3nm [4].

In typical SMLM reconstruction, a binning size of around 50 frames per bin is optimal for DME, versus a minimum of 500 required for cross-correlation methods. Depending on the drift in the system, this can result in an improvement in image quality. A major advantage of the DME is that in addition to the drift estimate, it also gives an estimate of it's own precision. The implementation reports this using the RMSD between the drift estimates of the split dataset, which can then be used to inspect the quality of the DME output and tune the bin size.

2-2-3 BaSDI algorithm

BaSDI algorithm is short form for Bayesian sample drift inference. This algorithm assumes a Hidden Markov Model (HMM), consisting of drift $\boldsymbol{d_k}$ as the hidden state and the observed frames $\boldsymbol{f}_k$ as the observed state. The absolute lateral drift in kth frame is denoted as $d_k \in \mathbb{Z}^{N\times 2}$ where N is the total number of frames. The observed frames, $\boldsymbol{f}_k \in \mathbb{N}^{h\times w}$ with h and w the height and width of image frame respectively. The frame consist of the number of emitters present in the bin at lateral dimension i, j denoted by f_{ij}. Collectively, all the frames are represented as $\boldsymbol{o} = \{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_N\}$. The number of emitters appearing in each bin and each frame are dictated by the molecular density of each bin, $\boldsymbol{\theta} \in \mathbb{R}^{h\times w}$ consisting of $\theta_{i,j}$ $\forall i, j$ which acts as a parameter for this model. Our aim is to find the drift in each frame $\boldsymbol{d_k}$ and thereby the molecular density, $\boldsymbol{\theta}$ with the only information available about the observed frames, $\boldsymbol{o}$.

The dynamical relationship between the current hidden state and the next hidden state gives rise to transition probability. In our case, we assume the drift as a random walk model The

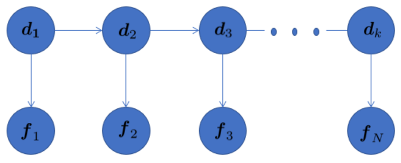

Figure 2-6: Hidden Markov Model (HMM)

relationship between the observed variable and the hidden variable gives us emission probability or observation probability. Using these probabilities, our aim for finding $\boldsymbol{d}_k$ and $\boldsymbol{\theta}_k$ could be analogously be stated as a twofold step. Firstly, performing system identification to find parameter $\boldsymbol{\theta}_k$ and secondly, obtaining the hidden states, $\boldsymbol{d}_k$ using smoothing. Unfortunately, performing either step requires the information about other. Whichever probability is easiest to deduce and carries most information, dictate what step is conducted first.

The drift is assumed to be dictated by a random-walk model which does not provide much information. On the other hand, the availability of frame data could provide an approximate information about molecular density. Therefore we assume the initial molecular density as the sum of all the frames and perform smoothing using forward-backward algorithm to obtain the posterior distribution of drift given the frame data, $\boldsymbol{f}_k$ and assumed molecular density, $\widehat{\boldsymbol{\theta}}^{\boldsymbol{i}}$ denoted by $P\left(\boldsymbol{d}|\boldsymbol{o}, \widehat{\boldsymbol{\theta}}^{\boldsymbol{i}}\right)$. Using this distribution we find the new parameter, $\widehat{\boldsymbol{\theta}}^{\boldsymbol{i+1}}$ which maximizes the (log) likelihood of observing the data, $\mathcal{L}\left(\boldsymbol{o}|\boldsymbol{d}, \boldsymbol{\theta}\right)$.

The problem statement is to obtain the MAP estimation of $\boldsymbol{d}$. This is in practice achieved using the viterbi algorithm. (2-2)

$$\widehat{\boldsymbol{d}}_{\text{MAP}} = \arg\max_{\mathbf{d}} P(\boldsymbol{o} \mid \boldsymbol{d}) P(\boldsymbol{d}) \tag{2-2}$$

Within the EM algorithm, there are multiple steps for E (Expectation evaluation) and M (Maximization) part. They are discussed thoroughly in the following sub-sections. For a short explanation and simple example of the EM algorithm, see Appendix A-2.

E-step

As the name states, this step aims at evaluating the expectation (over drift) of observing the frame data, $\boldsymbol{o}$ given the approximate molecular density, $\widehat{\boldsymbol{\theta}}$ and the marginalized distribution of drift for all it's support (i.e. $[-d_{max}, d_{max}]$) for all frames, $P(d_{k,1}, d_{k,2}|\boldsymbol{o}, \widehat{\boldsymbol{\theta}})\ \forall k$ ($d_{k,1}$ refers to drift in x-axis while $d_{k,2}$ indicates drift in y-axis). Since it is more convenient to evaluate the logarithm of probability distribution, we find log in (2-3). The reasoning behind taking the expectation over drift is explained in the M-step.

$$\mathbb{E}_{\boldsymbol{d} \in \boldsymbol{D}} \left[\log P(\boldsymbol{o} \mid \boldsymbol{d}, \widehat{\boldsymbol{\theta}}) \mid \boldsymbol{o}, \widehat{\boldsymbol{\theta}}\right] \tag{2-3}$$

To evaluate this, we need to obtain the marginal distribution $P(d_{k,1}, d_{k,2}|\boldsymbol{o}, \widehat{\boldsymbol{\theta}})\ \forall k$ which, in turn, requires the posterior distribution of drift. Invoking bayes theorem, we could state the dependence of this posterior distribution on the likelihood and the prior as follows

$$P\left(\boldsymbol{d} \mid \boldsymbol{o},\widehat{\boldsymbol{\theta}}\right) \propto P\left(\boldsymbol{o} \mid \boldsymbol{d},\widehat{\boldsymbol{\theta}}\right) P(\boldsymbol{d}) \tag{2-4}$$

Once the posterior distribution is obtained, one could perform marginalization, but since the resulting distribution is not analytical, marginalization has to be performed using a numerical technique. This is computationally expensive. Instead the authors utilize the forward-backward algorithm to directly compute the marginal probability in a computationally efficient manner. We discuss both of these component as follows and then focus on the forward-backward algorithm.

<ins>Prior</ins>
To be able to obtain the prior distribution of drift, an underlying displacement model must be assumed. As previously stated, the model is a Markov process called random walk. The algorithm has also assumed a maximum drift, d_{max}, thus the resulting distribution of the random walk model must be truncated. The resulting gaussian distribution is actually a truncated gaussian distribution.

$$\begin{aligned} P(\boldsymbol{d}) &= P\left(d_{11}\right) P\left(d_{12}\right) \prod_{k=2}^{N} P\left(d_{k,1}, d_{k,2} \mid d_{k-1,1}, d_{k-1,2}\right) \\ &= P\left(d_{1,x}\right) P\left(d_{1,y}\right) \prod_{k=2}^{N} P\left(d_{k,x}, d_{k,y} \mid d_{k-1,x}, d_{k-1,y}\right) \\ &= P\left(d_{1,x}\right) P\left(d_{1,y}\right) \prod_{k=2}^{N} t\left(d_{k,x}-d_{k-1,x}, d_{k,y}-d_{k-1,y}\right) \end{aligned} \tag{2-5}$$

Since the assumed random walk model is expressed as $x(k+1) = x(k) + \epsilon(k)$, it would mean that $\epsilon(k) = x(k+1) - x(k) \in \mathcal{N}(0, \sigma^2)$. Therefore,

$$t(\delta d_x, \delta d_y) = \mathcal{N}\left(\delta d_x; 0, \sigma^2\right) \mathcal{N}\left(\delta d_y; 0, \sigma^2\right) + \epsilon; \delta d_x < s, \delta d_y < s \tag{2-6}$$

The first argument of the normal distribution indicates which variable is the distribution being evaluated for (like δx). The rest indicates the mean and the variance of the distribution. For example, if we have $x(k+1) = x(k) + \epsilon(k)$, and it is required to evaluate $P(x_1|x_0) = P(x_0)P(x_1|x_0)$. It could be easily shown that $P(x_1|x_0) = P(x_0) * \mathcal{N}(\mathbb{E}[x(0)], \sigma^2)$. One would obtain the product of normal distributions for $P(x_n|x_{n-1})$ in general. With this, we obtain the prior of equation (2-4).

<ins>Probability of Observation</ins>
Normalized Molecular density at each spatial coordinate represents the probability of finding a molecule at that given coordinate. This is an example of multinomial distribution, whereby, the possibility of finding an emitter at position (i, j) is given $\theta_{i,j}$. The joint probability of finding or observing all these emitters (or localization events) in a frame $\boldsymbol{f}$ is given as follows.

$$P(\boldsymbol{f} \mid \boldsymbol{\theta}) = (\|\boldsymbol{f}\|_1)! \times \prod_{i,j} (\theta_{i,j})^{f_{ij}} \tag{2-7}$$

The first term in the above equation represents the number of sequences in which we would observe the emitters present in the frame $\boldsymbol{f}$. It is always an integer. Since there is drift, $(d_x$,$d_y)$ acting on this frame, the resulting probability needs to be adjusted by shifting the frame by this amount (indicated by the shift in indices). In some cases, this shifting could be large enough to push the emitter position out of the reference frame. This is why we pad the frame image with a "0" border of size d_{max}. Then, as long as the maximum drift does not exceed d_{max}, we can still perform the computation to obtain reasonable approximate of the a posteriori distribution. Furthermore, as is conventional, we employ logarithm of the probability instead.

$$\log P(\boldsymbol{f} \mid \boldsymbol{\theta}, d_x, d_y) \approx \log(\|\boldsymbol{f}\|_1)! + \sum_{i,j} f_{i-d_x,j-d_y} \log \theta_{i,j} \tag{2-8}$$

Performing this computation over all the frames, $\boldsymbol{o}$ and segregating the term that are independent of $\boldsymbol{\theta}$, we obtain the final form of probability of observation.

$$\log P(\boldsymbol{o} \mid \boldsymbol{d}, \boldsymbol{\theta}) \approx \sum_k \sum_{i,j} o_{i',j',k} \log \theta_{i,j} + \text{ constant} \tag{2-9}$$

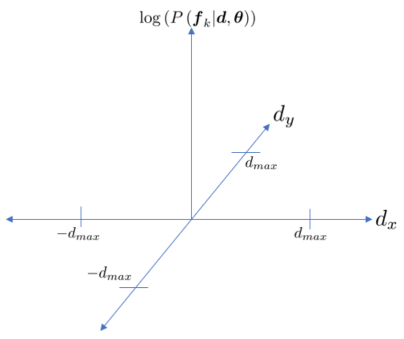
$\log(P(\boldsymbol{f}_k|\boldsymbol{d},\boldsymbol{\theta}))$
d_y
d_{max}
$-d_{max}$
d_{max}
d_x
$-d_{max}$

Figure 2-7: The probability of observing the localization given the molecular density and the value of absolute drift. The evaluation is across different values of drift in lateral axis calculated using convolution

with i' and j' representing the shifted indices of each frame. The above operation involve shifting and subsequent multiplication which is basically to correlation. Image correlation suddenly finds a meaningful explanation here. For efficient computation, 2D convolution is utilized to evaluate the above, with one of the term (either $\log \theta_{i,j}$ or $\boldsymbol{o}_{i',j',k}$) reversed. The result is a matrix $\in \mathbb{R}^{N \times 2d_{max} \times 2d_{max}}$ consisting of the logarithmic probabilities with different values of drift. For a single frame, the support of evaluated probability takes the form as shown in Figure 2-7.

Forward-Backward Algorithm

To evaluate the marginal distribution, *forward-backward* algorithm is utilized. This algorithm, instead of evaluating the marginal distribution through performing numerical integration over the joint drift distribution, directly computes the marginal distribution by operating over observation probability with the prior. There are two parts of computation, forward and backward. In the forward computation, the algorithm takes the starting frame and work it's way forward through all the combinations of drift which would lead to the current frame $\boldsymbol{f}_k$. In the backward computation, the final image (or equivalently, the last frame) is worked backwards through all the drift sequences to reach the current frame $\boldsymbol{f}_k$.

1. **Forward:** This computation begins from the first frame and goes up-to the current frame through all the possible drift sequences. The probability of this transition is obtained through (2-10).

$$\begin{aligned} &\alpha(d_x, d_y, 1) \sim P\left(\boldsymbol{f}_1 \mid \widehat{\boldsymbol{\theta}}^{(i)}, d_x, d_y\right) \\ &\alpha(d_x, d_y, k) \sim P\left(\boldsymbol{f}_k \mid \widehat{\boldsymbol{\theta}}^{(i)}, d_x, d_y\right) \sum_{\delta d_x, \delta d_y} \alpha(d_x, d_y, k-1) t(d_x - \delta d_x, d_y - \delta d_y), \end{aligned} \tag{2-10}$$

   Upon careful observation, it could be concluded that the summation in the second term is a convolution operation with a gaussian kernel. This fact is utilized for computational efficiency whilst implementation.

   (2-10).

2. **Backward:** This computation is performed using (2-11). The execution is quite similar to the Forward part except that we iterate backwards.

$$\begin{aligned} &\beta(x, y, N) = 1 \\ &\beta(x, y, k) \sim \textstyle\sum_{\delta x, \delta y} P\left(\boldsymbol{f}_k \mid \widehat{\boldsymbol{\theta}}^{(i)}, \delta x, \delta y\right) \beta(x, y, k+1) t(\delta x - x, \delta y - y) \end{aligned} \tag{2-11}$$

After obtaining both the forward and backward part, we multiply them both to obtain the final marginal distribution (2-12).

$$P\left(d_{k,1}, d_{k.2} \mid \boldsymbol{o}, \widehat{\boldsymbol{\theta}}^{[i]}\right) \sim \alpha(x, y, k)\beta(x, y, k) \tag{2-12}$$

With this, all the necessary terms to evaluate the expectation (2-3) over drift are at hand.

M-Step

The M-step is aimed at obtaining the maximum likelihood of the expectation evaluated in the E-step. The reason why we evaluate expectation over drift is to remove the drift argument, $\boldsymbol{d}$ from the expectation. This is only possible if we have the marginal distribution of drift, which is already evaluated in the E-step.

$$\widehat{\boldsymbol{\theta}}^{[i+1]} = \arg\max_{\theta} \mathbb{E}_{\boldsymbol{d} \in \boldsymbol{D}} \left[\log P(\boldsymbol{o} \mid \boldsymbol{d}, \boldsymbol{\theta}) \mid \boldsymbol{o}, \boldsymbol{\theta}^{[i]}\right] \tag{2-13}$$

It is possible to obtain an analytical solution to the above optimization problem [8]. The calculation are thoroughly stated in the reference and the final result is stated below.

$$\widehat{\theta}_{ij}^{[i+1]} \propto \sum_k \sum_{d_{k,x}, d_{k,y}} P\left(d_{k,x}, d_{k,y} \mid \boldsymbol{o}, \widehat{\boldsymbol{\theta}}^{[i]}\right) o_{i-d_{k,x}, j-d_{k,y}, k} \tag{2-14}$$

In principle, the above equation is a weighted sum over all the frame shifted by a drift value with the weight as the probability of that value of drift thus the most probable value of drift carries the most weight. At the end of the iteration, an improved approximate of the molecular density is obtained.

Iterations

Once an improved approximate is obtained, the EM cycle could be repeated. It is ensured that with each iteration, an improved estimate is obtained and that the estimate converges to a specific value as well. The convergence of drift's distribution and the maximum number of iterations is used to dictate the flow of loop. For e.g. if the difference between the standard deviation of drift in one iteration and the succeeding iteration is smaller than a certain pre-defined a value, then the loop stop. Furthermore, at the beginning of loop, the molecular density is smoothed using a gaussian kernel. The scale of this kernel is decreased with each iteration such as to be able to extract more information.

Results

BaSDI algorithm achieves a better precision as compared to the existing correlation methods and almost as good as fiducial marker based correction method. The key difference between the BaSDI method and the correlation method lies in the step of aggregation, which produces a discredited trace of drift between each aggregate in correlation based method. BaSDI on the other hand, created a more detailed trace of the drift between each frames resulting in better resolved image features. In comparison to correlation based method that produced an error of 7 pixels, BaSDI is able to deliver an error of 1 pixel on the same dataset [8]. The main drawback with BaSDI is for large data-set (for e.g. DNA-PAINT), it consumes a large processing time to be able to produce drift results which are similar in terms of precision to fiducial marker correction.

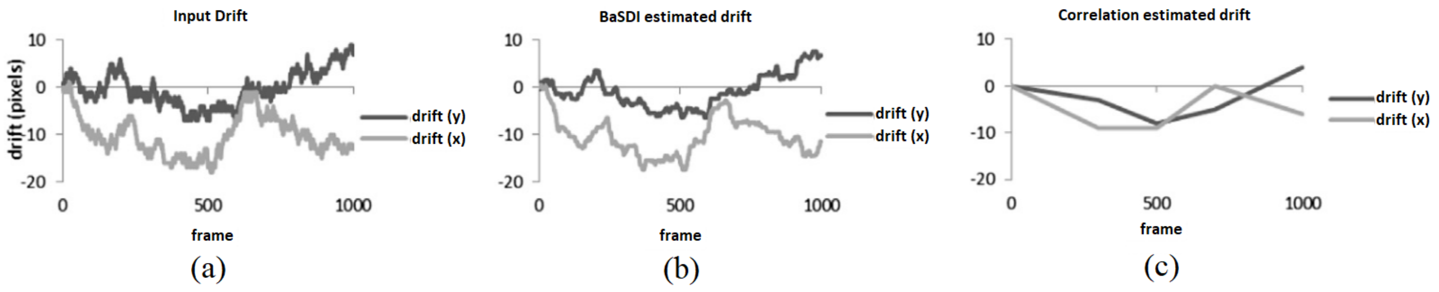

Figure 2-8: Results showing low rate performance of BaSDI and image correlation method for drift trace estimation on simulated data [8]

2-3 State Space Modelling of Drift in SMLM

State space models refers to a class of mathematical modelling techniques which utilizes first order differential equations to model the dynamics of physical and abstract processes. They are alternative to traditional transfer function/differential equations which are mostly used to model Single Input Single Output (SISO) systems with zero initial condition. The state space representation of a dynamical system consists of the evolution model for the state variables (2-15), also called process dynamics and the observation model (2-16) that links the observations or output to the state variables. The state, $x(t) \in \mathbb{R}^n$ and the output $y(t) \in \mathbb{R}^m$ are represented as follows.

$$\dot{x}(t) = f\left(x(t), y(t)\right) \tag{2-15}$$

$$y(t) = h\left(x(t), y(t)\right) \tag{2-16}$$

Linear state space models are ones for which $f(\cdot)$ and $h(\cdot)$ are linear functions. Furthermore, state space models are also used to represent stochastic processes such as linear gaussian process or linear poisson processes, depending on the type of process and output noise influencing the system. Stochastic processes represented in state space format are also called Hidden Markov Model (HMM). The probability distribution of state for Markov processes depends only on the present state, not the past. The reason why it is called *Hidden* is because of the fact that one can only observe the output, $y(t)$, not the state, $x(t)$ which remains hidden. For computational purposes, we utilize discrete time systems as dealing with real-time systems requires collecting data at specific time-steps or sampling frequency. Equation (2-17) and (2-18) represents an autonomous (i.e. without the external influences of input) example of linear time invariant gaussian process. It is time-invariant since the process dynamics matrix A and the output matrix C do not change with time. This representation is used throughout the discussion.

State Dynamics:

$$\begin{aligned} x(k+1) &= Ax(k) + w(k) \\ \text{with } & w(k) \sim \mathcal{N}(\mathbf{0}, \Sigma_w) \end{aligned} \tag{2-17}$$

Output Dynamics:

$$\begin{aligned} y(k) &= Cx(k) + v(k) \\ \text{with } & v(k) \sim \mathcal{N}(\mathbf{0}, \Sigma_v) \end{aligned} \tag{2-18}$$

Hitherto, utilization of state space for modelling drift in SMLM is not documented, although using it might yield edge in terms of performance, computational time and ease of understanding. Furthermore, previously mentioned BaSDI algorithm utilizes forward-backward algorithm which is similar to the Kalman RTS smoother algorithm. Furthermore, BaSDI also uses a implicit HMM model for drift. We aim to model the process of drift using explicit state space models to discover properties of system. The following chapters develop two state space models to describe drift in SMLM. These models are further used to perform drift compensation through the methods of system identification and filtering/smoothing for respective models.

Chapter 3

State Space Modelling using Shift Matrices

The spatial molecular density dictates the number of emitter molecules present at each spatial position, as shown in Figure 3-1. This density roughly represents the underlying sample image which drifts throughout the course of experiment. The process of SMLM and drift could be represented by the combination of this spatial molecular density and drift, which is the foundation for this state space model. BaSDI algorithm also uses shifting of observed frames in it's likelihood function (2-8), while keeping spatial molecular density fixed. Our observation is that one could shift the spatial molecular density instead of shifting frame and obtain an explicit linear hidden markov model or a state space model as shown in this chapter.

3-1 1D Shift Matrix Model

We assume in this model that the drift takes integer values, i.e. the molecules could shift by integer multiple of a defined unit. This is because the spatial molecular density is defined for discretized spatial grid. For simplicity, we start with 1D representation of the model i.e. we only consider drift in one axis. Furthermore, for the purpose of simulation and generating data, we also assume the following.

Assumption 1: The maximum absolute drift is d_{max}.

Assumption 2: The relative drift, $d(k) \in \mathbb{Z}$ takes a value within a specified interval $[-\bar{d}, \bar{d}]$ where $\bar{d}$ is smaller than d_{max}.

With the first assumption, we would be able to select the width of padding required to avoid loss of information of spatial molecular density. The second assumption lays the foundation for system identification algorithm for this model. We represent the relative drift at time step k as $d(k) \in \mathbb{Z}$ and spatial molecular density as $\theta(k) \in \mathbb{R}^{n+2d_{max}}$ where n is the number of

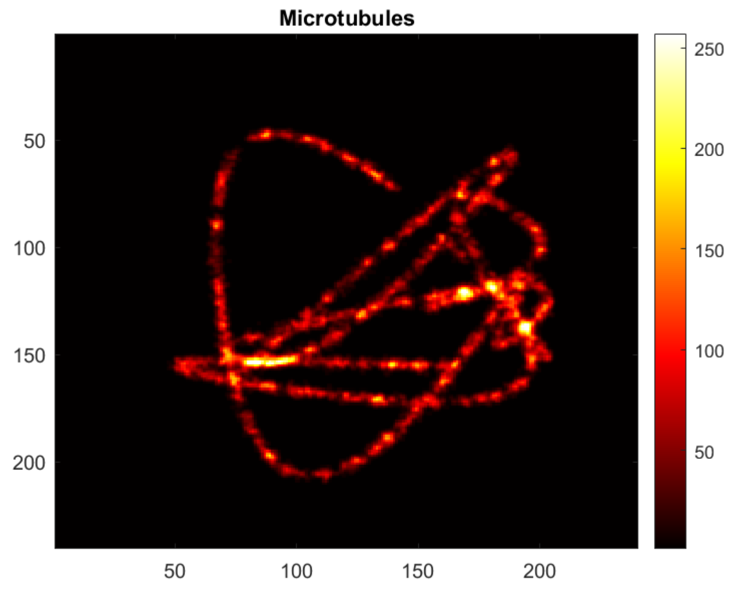

Figure 3-1: Simulated SMLM image of Microtubules showing the regions with high and low number of emitter molecules

bins or size of grid and d_{max} is the maximum drift. The extended size of spatial molecular density $(2d_{max})$ is analogous to padding an image with zeros following the first assumption. Padding is important such as to avoid loss of information of spatial molecular density through successive shifting of image. The output frame, $y(k) \in \mathbb{Z}_+^{n+2d_{max}}$ consist of subset of emitters blinking at time instant k in each spatial coordinate.

State Dynamics:

Spatial Molecular Density Dynamics

$$\theta(k+1) = A_{d(k)}\theta(k) \quad \text{where,} \tag{3-1}$$

The state transition matrix $A_{d(k)}$ are shift matrices with shift equal to the value of $d(k)$. For e.g. if $d(k) = 1, -1$ or 0, $A_{d(k)}$ takes the following instances of upper shift, lower shift and identity matrices. This could be extended to any value of $d(k)$

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \dots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} A_{-1} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \dots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix} A_0 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \dots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix} \tag{3-2}$$

Output Dynamics:

$$y(k) = \text{Poisson}\left(\theta(k)\right), \tag{3-3}$$

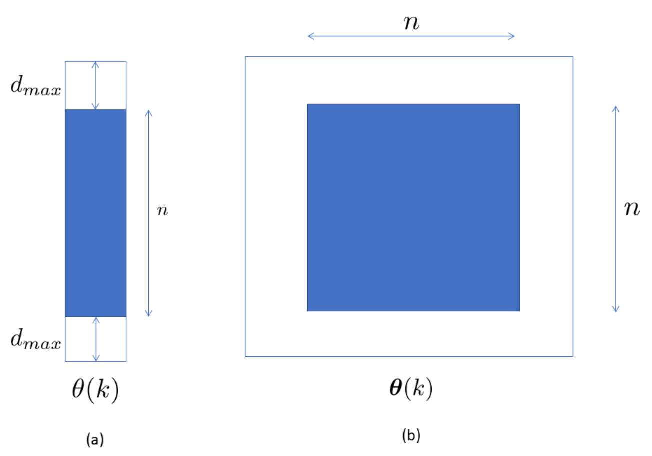

Figure 3-2: (a)Padding spatial molecular density vector $\theta(k)$ with zeros of length equal to the maximum expected drift, d_{max} (b) Padding an image

$$\text{where,}\quad \text{Poisson}(\lambda) = \frac{\lambda^z}{z!}e^{-\lambda}$$

Depending on the exponent, each support of poisson distribution carries a different probability. This is shown for various values of exponent in Figure 3-3. As the exponent of the distribution decreases, the values near zero are more likely to occur. In case of above SMLM drift model, the spatial emitter density influences the exponent, and therefore the shape of the poisson distribution curve.

$$\begin{aligned} \theta(k+1) &= A_{d(k)}\theta(k) \\ y(k) &= \text{Poisson}\left(\theta(k)\right) \end{aligned} \tag{3-4}$$

The value of spatial emitter density depends on a lot of factors. To name a few, it depends on the sample structure, the type fluorophore probe (cy5,cy3 etc), the amount of dispersion, inherent properties like photo-bleaching and excitation rate. For the sake of simplicity, we simply assume that the spatial molecular density depends only on the shape of the sample image. Since $\theta(k)$ is our state, the initial value of state, $\theta(0)$ represents the underlying sample image that we would like to obtain. All the subsequent states, $\theta(k)$ simple represent a shifted version of $\theta(0)$. The step for evaluating the initial state is therefore called *Image Reconstruction* whereas estimating drift from data is referred to as *System Identification.*

Figure 3-4 shows data from simulating the model in (3-1) and (3-3). In Figure 3-4(a), one could notice the initial state $\theta(0)$ which is shown in the left most side appended graph depicting two rectangles. These rectangles get displaced by drift, $d(k)$ across frames and using these values through a poisson distribution, we obtain the output data, $y(k)$.

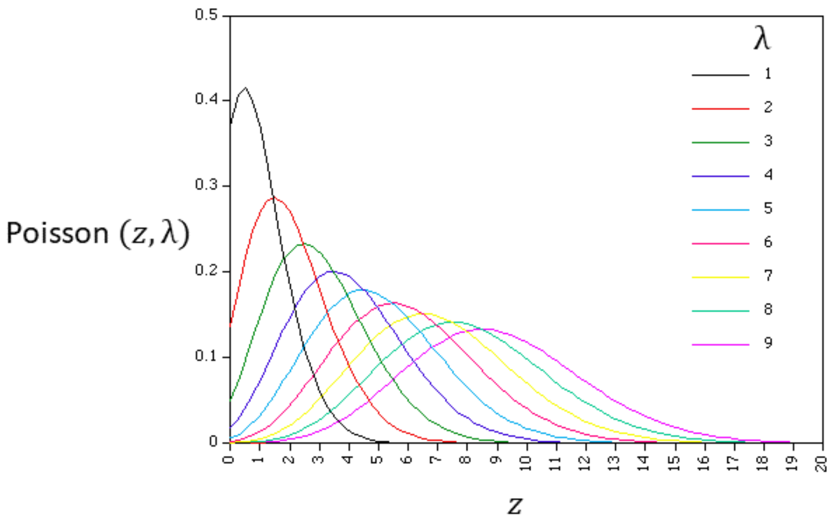

Figure 3-3: Poisson Distribution for different exponent values, λ. For large values of λ, the distribution converges to a gaussian distribution and for small values of λ, the values near zero are more likely to occur.

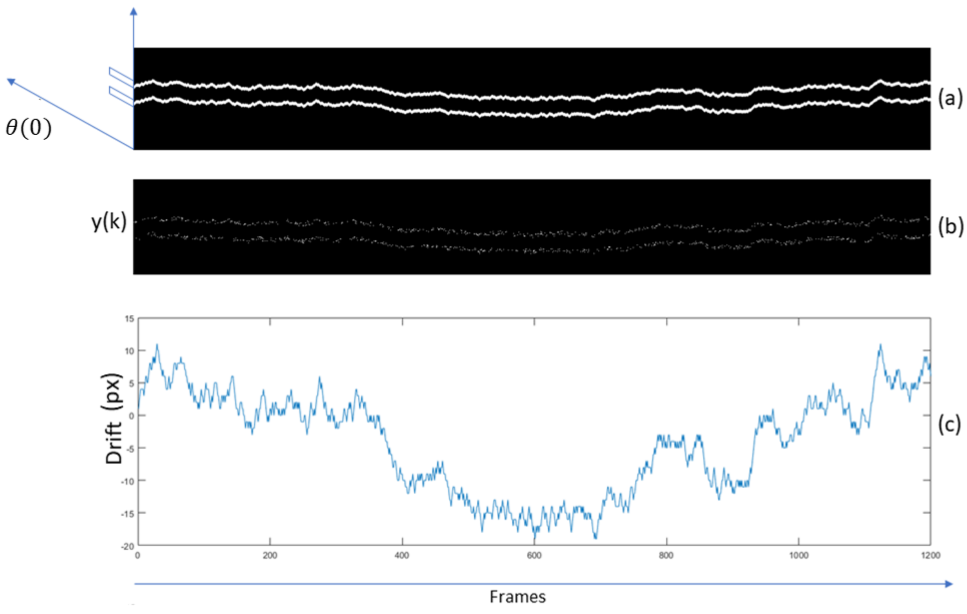

Figure 3-4: Illustration of the data obtained from simulating shift state space model. The data is generated using $\theta(0)$ as two stripes, as shown in the left most border of (a) having a value of 0.01. The state of the system, namely spatial molecular density, $\theta(k)$, is shown for all time steps as well. (b) The output of the system, $y(k)$ and (c) The drift, $d(k)$

3-2 2D Shift Matrix Model

In 2D, the spatial molecular density is represented by an image which could drift right and left along with upwards and downwards. We use **bold** symbols to separate the 1D model with 2D model. Now, lateral drift is denoted by $\boldsymbol{d}(k) \in \mathbb{Z}^2$ such that $\boldsymbol{d}(k) = [d_x(k), d_y(k)]^\top$

with $d_x(k)$ as the drift in horizontal direction and $d_y(k)$ as the drift in vertical direction. The state, which is spatial molecular density denoted by $\boldsymbol{\theta}(k) \in \mathbb{R}^{(n+2d_{max})\times(n+2d_{max})}$ with $d_{max} \in \mathbb{Z}$ as the maximum drift. $\boldsymbol{\theta}(k)$ is a matrix representation of sample image. Similarly, the output frames, $\boldsymbol{y}(k) \in \mathbb{Z}^{(n+2d_{max})\times(n+2d_{max})}$ are matrix representation of localizations. These localizations are extracted from frames captured through microscope. It should be noted that the padding here is around the four borders of the image as shown in Figure 3-2(b). Furthermore, it is not necessary that image should have same breadth and width, but for the sake of simplicity we take them to be same.

<u>State Dynamics</u>:

Spatial Molecular Density Dynamics

$$\boldsymbol{\theta}(k+1) = A_{d_x(k)}\boldsymbol{\theta}(k)B_{d_y(k)} \tag{3-5}$$

<u>Output Dynamics</u>:

$$\boldsymbol{y}(k) = \text{Poisson}(\boldsymbol{\theta}(k)) \tag{3-6}$$

Both $A_{d_y(k)}$ and $B_{d_x(k)}$ are shift matrices as elaborated in (3-2). In the above state dynamics, we could observe both, a pre and post multiplication with shift matrices. Pre-multiplication shifts the spatial molecular density upwards and downwards, similar to the 1D model. Post-multiplication shifts it right and left. Figure 3-5 shows some highlights obtained by simulating the above equations. Although complete, the model is bi-linear due to the post and pre multiplication. For analytical purpose, it is easier to deal with a linear model. Thus, to obtain linear form of this equation, we utilize the following theorem.

<u>Theorem</u>: *The consecutive multiplication of three matrices could be stated as follows in a vectorized format*

$$\text{vec}(ABC) = \left(C^\top \otimes A\right)\text{vec}(B)$$

where $\otimes$ is *Kronecker Product*

Using this theorem, we could extend our 2D spatial molecular density into a 1D vector. We re-write (3-5) as follows.

$$\begin{aligned}\text{vec}(\boldsymbol{\theta}(k+1)) &= \text{vec}(A_{d_y(k)}\boldsymbol{\theta}(k)B_{d_x(k)}) \\ &= \left(B^\top_{d_x(k)} \otimes A_{d_y(k)}\right)\text{vec}(\boldsymbol{\theta}(k))\end{aligned} \tag{3-7}$$

Thus, we re-define our new spatial molecular density state, the state transition matrix and the new output as,

$$\begin{aligned}\boldsymbol{\Theta}(k) &= \text{vec}(\boldsymbol{\theta}(k)) \\ \boldsymbol{A}_{d_x(k),d_y(k)} &= \left(B^\top_{d_x(k)} \otimes A_{d_y(k)}\right) \\ \boldsymbol{\mathcal{Y}}(k) &= \text{vec}(\boldsymbol{y}(k))\end{aligned} \tag{3-8}$$

which gives us our final model. This is a linear time varying stochastic system with integer output. The model is similar to the 1D state space model derived in previous section.

$$\begin{aligned} \mathbf{\Theta}(k+1) &= \boldsymbol{A}_{d_x(k),d_y(k)}\mathbf{\Theta}(k) \\ \boldsymbol{\mathcal{Y}}(k) &= \text{Poisson}\left(\mathbf{\Theta}(k)\right) \end{aligned} \tag{3-9}$$

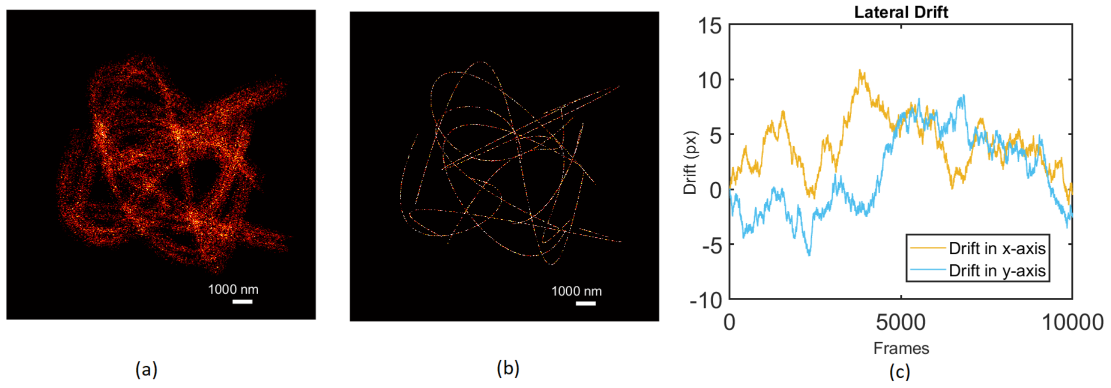

Figure 3-5: Result of simulating 2D state space model using the spatial molecular density data from microtubule simulation. The data consist of a 200px×200px (100nm/px) image with 1000 binding sites and frame molecular density of about 1.6×10^{-4}. (a) Shows the sum of all output frames $\boldsymbol{y}(k)$. In comparison to this, (b) Shows the actual underlying microtubule image, $\boldsymbol{\theta}(k)$. Upon comparing both, it is clear how drift produces a smearing effect. (d) Shows the generated drift used to form the data

3-3 Drift Compensation Algorithm using Shift Matrix State Space

After we have developed the Shift Matrix State Space (SMSS) models, our aim is to determine the parameters of this model using observed data. These parameters are the drift $d_x(k)$ and $d_y(k)$, i.e. the subscript of shift matrices, $\boldsymbol{A}_{d_x(k),d_y(k)}$ and the underlying sample image, $\mathbf{\Theta}(k)$ which together lead to the process of eliminating drift, or drift compensation. The method of finding the parameters is called *System identification* and the process of finding the underlying sample images, $\mathbf{\Theta}(k)$ is called *Smoothing*. But since $\mathbf{\Theta}(k)$ is just a shifted version of $\mathbf{\Theta}(0)$, our aim is to only find $\mathbf{\Theta}(0)$ which is referred to as *Image Reconstruction* algorithm.

3-3-1 System Identification

The system under consideration has integer observation, $\boldsymbol{\mathcal{Y}}(k)$. Subspace identification technique usually deal with Linear Gaussian systems. Thus a more viable option is to use the conventional likelihood maximization for estimating the state transition matrix $\boldsymbol{A}_{d_x(k),d_y(k)}$ and thereby the drift $\boldsymbol{d}(k)$. Since the assumed model utilizes relative drift, each value of $\boldsymbol{d}(k)$ is independent of other and thus each state transition matrix is independently determines the shift in the subsequent image. An important thing to recall is the drift set for which the likelihood function is evaluated. This set of drift values is given by $[-\bar{d}, \bar{d}]$ for both d_x and d_y in line with **Assumption 2**.

Thus we can optimize over each value of observed output frame independently. Before the System identification algorithm is initiated, we require information about the initial state,

$\boldsymbol{\Theta}(0)$, which is the underlying sample image. Since we do not know it beforehand, we roughly approximate it by summing up all the output frames and assuming the result as our initial states. Let this approximation of initial state be represented by $\widehat{\boldsymbol{\Theta}}(0)'$,

$$\widehat{\boldsymbol{\Theta}}(0)' = \sum_{k}^{N} \boldsymbol{\mathcal{Y}}(k) \tag{3-10}$$

There are three steps involved for system identification algorithm of the above model are as follows.

- **Evaluation** : The log likelihood is evaluated for the current frame with state transition matrix $\boldsymbol{A}_{d_x(k),d_y(k)}$ corresponding to all values of drift from the set $[-\bar{d},\bar{d}]$ for both d_x and d_y. This is following **Assumption 2**. The likelihood for $\boldsymbol{\mathcal{Y}}(k)$, given the previous state, $\widehat{\boldsymbol{\Theta}}(k-1)'$ depends on $\boldsymbol{A}_{d_x(k),d_y(k)}$ since

  $$\boldsymbol{\mathcal{Y}}(k) = \text{Poisson}\left(\widehat{\boldsymbol{\Theta}}(k)'\right) = \text{Poisson}\left(\boldsymbol{A}_{d_x(k-1),d_y(k-1)}\widehat{\boldsymbol{\Theta}}(k-1)'\right)$$

  since,

  $$\widehat{\boldsymbol{\Theta}}(k)' = \left(\boldsymbol{A}_{d_x(k-1),d_y(k-1)}\widehat{\boldsymbol{\Theta}}(k-1)'\right)$$

  Thus we denote the likelihood as $\mathcal{L}\left(\boldsymbol{\mathcal{Y}}(k)|\widehat{\boldsymbol{\Theta}}(k-1)', d_x(k-1), d_y(k-1)\right)$, i.e. the likelihood of observing $\boldsymbol{\mathcal{Y}}(k)$ given the information about the last state and assumed drift value. For notation purpose, the subscript i denotes the $i'th$ element of the vector. For e.g. $\boldsymbol{\mathcal{Y}}(k)_i$ is the i'th element of output vector $\boldsymbol{\mathcal{Y}}(k)$.

  $$\mathcal{L}\left(\boldsymbol{\mathcal{Y}}(k)|\widehat{\boldsymbol{\Theta}}(k-1)', d_x(k-1), d_y(k-1)\right) = \\ \prod_i \frac{\left(\left(\boldsymbol{A}_{d_x(k-1),d_y(k-1)}\widehat{\boldsymbol{\Theta}}(k-1)'\right)_i\right)^{\boldsymbol{\mathcal{Y}}(k)_i} \exp\left(-\left(\left(\boldsymbol{A}_{d_x(k-1),d_y(k-1)}\widehat{\boldsymbol{\Theta}}(k-1)'\right)_i\right)\right)}{(\boldsymbol{\mathcal{Y}}(k)_i)!} \tag{3-11}$$

  A more convenient way of evaluating the likelihood is by evaluating the logarithm of the likelihood as follows:

  $$\log\left(\mathcal{L}\left(\boldsymbol{\mathcal{Y}}(k)|\widehat{\boldsymbol{\Theta}}(k-1)', d_x(k-1), d_y(k-1)\right)\right) = \sum_i \boldsymbol{\mathcal{Y}}(k)_i \log\left(\left(\boldsymbol{A}_{d_x(k-1),d_y(k-1)}\widehat{\boldsymbol{\Theta}}(k-1)'\right)_i\right) \\ - \sum_i \left(\boldsymbol{A}_{d_x(k-1),d_y(k-1)}\widehat{\boldsymbol{\Theta}}(k-1)'\right)_i \tag{3-12}$$

  ignoring the term which remains constant, specifically, the numerator from (3-11).

- **Optimization** : Whichever value of $d_x(k)$ and $d_y(k)$ yields the highest likelihood is the most likely value of drift for frame k. Since the number of values is drift values over which likelihood is evaluated is small, one can evaluate the minimizing argument in significantly small computational time without requiring complex technique such as gradient descent.

$$\widehat{\boldsymbol{d}}(k-1) = \underset{d_x(k-1),d_y(k-1)}{\arg\max} \left(\log\left(\mathcal{L}\left(\boldsymbol{\mathcal{Y}}(k)|\widehat{\boldsymbol{\Theta}}(k-1)', d_x(k-1), d_y(k-1)\right)\right)\right)$$

- **Update** : The next state is updated using the recursion in (3-9).

$$\widehat{\boldsymbol{\Theta}}(k)' = \boldsymbol{A}_{d_x(k-1),d_y(k-1)}\widehat{\boldsymbol{\Theta}}(k-1)'$$

  and the whole step is performed again for observed data $\boldsymbol{\mathcal{Y}}(k+1)$. All the three steps are then repeated for all observed frames.

It should be noticed that the same algorithm could be used for 1D case as well since both the systems are exactly same except the state transition matrix $\boldsymbol{A}_{d_x(k-1),d_y(k-1)}$ which would be replaced by $A_{d_x(k-1)}$ and we would maximize for the single argument, $d_x(k)$.

3-3-2 Image Reconstruction

Once we have obtained the most likely drift trace from the output data using System Identification, the question lies in finding out the initial spatial molecular density, $\boldsymbol{\theta}(0)$ which is equivalent to reconstructing the underlying sample image.

In the identification cycle, we assumed the initial state as the normalized sum of the output (3-10) to find the best estimate of drift. Once the best estimate of drift $\widehat{\boldsymbol{d}}(k)$ is available, we can counter adjust each of the output frames to obtain the drift compensated output, $\widehat{\boldsymbol{\mathcal{Y}}}(k)$. This output could then be used to find a better approximation of spatial molecular density by summing up all the drift corrected output frames. The following step show the process of obtaining the proper estimate of the spatial molecular density.

1. **Evaluation**: Evaluate the absolute drift, $\tilde{\boldsymbol{d}}(k)$ using the estimated relative drift, $\widehat{\boldsymbol{d}}(k)$, which is basically the cumulative sum of the latter.

2. **Compensation**: Obtain the counter drift, i.e. $-\tilde{\boldsymbol{d}}(k)$

3. **Reconstruction**: Apply this absolute counter drift on each of the output to obtain negate the effect of drift.

$$\widehat{\boldsymbol{\mathcal{Y}}}(k) = \boldsymbol{A}_{-\tilde{d}_x(k-1),-\tilde{d}_y(k-1)}\boldsymbol{\mathcal{Y}}(k)$$

   The new spatial molecular density,$\boldsymbol{\Theta}(k)^*(0)$ is formed using the drift compensated output frames

$$\boldsymbol{\Theta}(k)^*(0) = \frac{1}{N}\sum_k \widehat{\boldsymbol{\mathcal{Y}}}(k) \tag{3-13}$$

3-4 Algorithm Development

Now, we would like to test the drift estimation (system identification) and image reconstruction algorithm and make adjustment to improve the performance. We start with the simplest case first, that of 1D case using two stripe spatial molecular density data as previously shown in Figure 3-4, but with rich frame output data (i.e. large number of molecules appearing in each output frame). The dataset consist of 200px data in length with discrete relative drift generated from uniform distribution over values $[-1\text{px}, 0\text{px}, 1\text{px}]$. Furthermore, the spatial molecular density of simulated date is assumed to be uniform over all the sample image. Upon successful working, we would increase the sparsity of the data by scaling the spatial molecular density desirably. The output frames are shown in Figure 3-6 (b).

We begin by evaluating the approximate of initial spatial molecular density, $\theta(0)$ which is obtained through normalized sum of all output frames. Then the likelihood is evaluated for the second frame, $y(1)$ over the drift set considered, $[-\bar{d}, \bar{d}]$ for $\bar{d} = 1$, i.e. $\mathcal{L}(y(1)|\theta(0), -1)$, $\mathcal{L}(y(1)|\theta(0), 0)$ and $\mathcal{L}(y(1)|\theta(0), 1)$ with A_{-1}, A_0 and A_1 respectively. Whichever value of $d(0)$ gives the least negative log likelihood is the best estimate of drift in first frame. This estimate is used to update the current state. The updated state, $\theta(1)$ is used in the next iteration with frame $y(2)$ to find $d(1)$. Likewise, we evaluate the drift for the whole dataset. The resulting estimated drift is shown in Figure 3-6 (a) along with the drift compensated output data in (c).

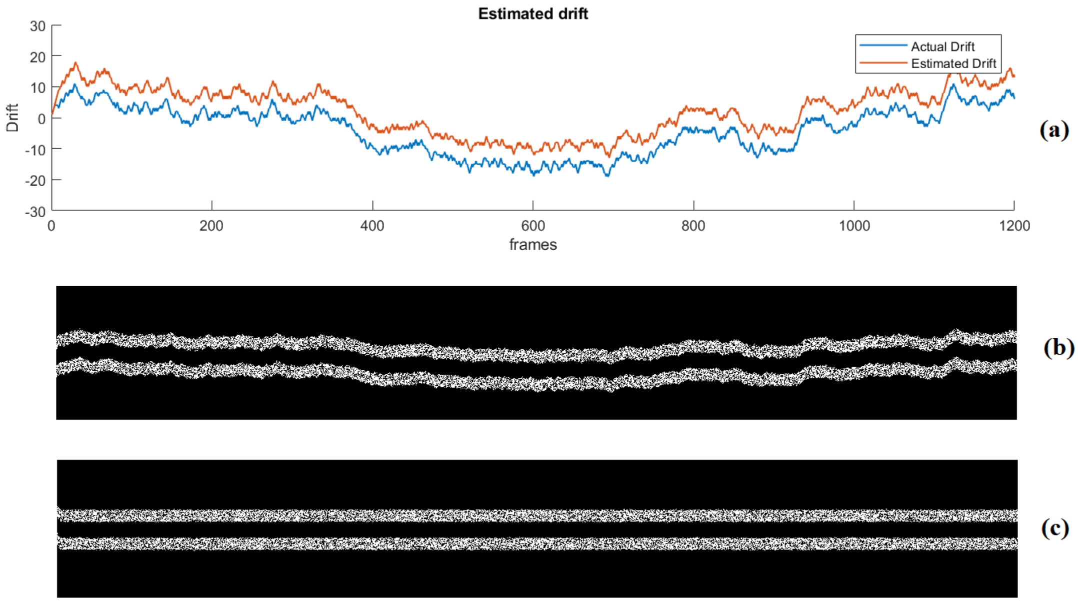

Figure 3-6: Result of System Identification and Image reconstruction algorithm on a dataset generated using 1D state space model. The data is generated using $\theta(0)$ consisting of two stripes as shown in Figure 3-4 having unit value. This generates a rich output data, $y(k)$ as shown in (b) consisting of 30 molecules per frame (m/f) on average. Using the System Identification we obtain the estimated drift, $\widehat{d}(k)$ as shown in (a) along with the ground truth drift, $d(k)$. A bias is added in the resulting estimated drift to show both curves separately. Using the estimated drift along with image correction algorithm, we obtain the drift compensated data,$\hat{y}(k)$ in (c)

To increase the complexity similar, to actual experiments, we observe the result obtained using lowered spatial molecular density as shown in Figure 3-7. The drift estimation fails due to the reduced number of molecules appearing in each frame. Due to this, all drift values have equal likelihood and thus optimization operation becomes trivial. To increase the information in single frame, we combine a specific number of frames together to form a single frame. This is referred to as *frame binning*. Throughout these combined frames, drift is assumed to be constant. The result of binning is a new output dataset, $Y(k)$ with reduced number of frames. The process of binning is illustrated in Figure 3-8

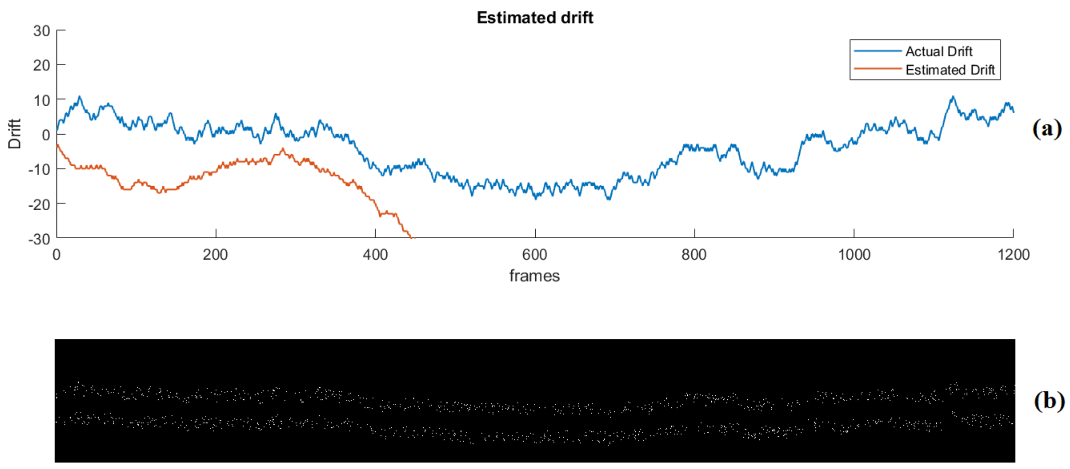

Figure 3-7: Result of System Identification on sparser data generated using $\theta(0)$ consisting of two stripes with value scaled down to 0.03. As evident in (a) the drift estimation fails due to reduced number of emitters appearing in each frame as shown in (b) which consist of 1 molecule per frame (m/f) on average

As a result of binning, it is more likely that larger values of drift could be encountered which is why, we increase the set of drift values for which we perform optimization as well (i.e. increase the value of $\bar{d}$). This modified set is different from the one considered in **Assumption 2**.

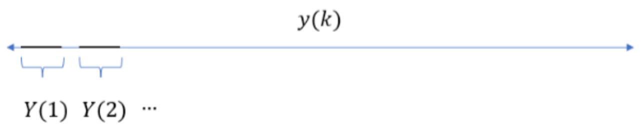

Figure 3-8: Illustration of the process of binning

Furthermore, the resulting reconstructed sample image, $\theta(k)^*(0)$ from the image reconstruction step, (3-13), is utilized to perform the whole cycle of system identification and image reconstruction again by feeding back the reconstructed image as the initial state for the second iteration. This is similar to the EM algorithm which reuses the maximized argument to re-evaluate the expectation, yielding a better estimate at each iteration [7]. More information about EM algorithm is available in Appendix A-2. In generalized sense, the reconstrcuted image obtained at each loop could be represented by $\theta(k)(0)^i$ where superscript i is the iteration number. This whole process of recycling the reconstructed image could be repeated until a convergence is obtained in the drift estimate, $\hat{d}(k)$. We could measure the Root Mean Squared Error (RMSE) between the current drift estimate (obtained through current iteration) and the one obtained in previous iteration. If the RMSE is below a pre-specified threshold, then convergence is said to be achieved. Furthermore, we also limit the number of iterations by a pre-specified maximum value. Figure 3-9 shows this loop schematically.

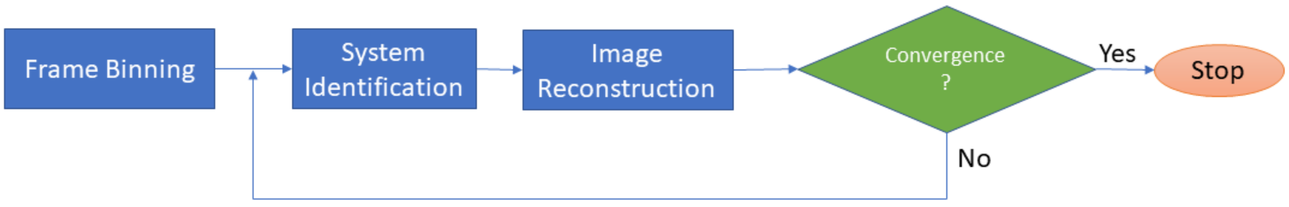

Figure 3-9: Illustration of the extended drift correction algorithm

Using the extended drift correction algorithm, we simulate few more increasingly sparsed dataset using the 1D model and observe the performance as shown in Figure 3-10.

The resulting performance is adequate to test the extended algorithm on 2D state space model. One important thing to keep in mind is that the same spatial molecular density value would generate different number of emitter molecules per frame in 1D case as compared to the 2D case because there is an added dimension. Due of this, we would like to test the 2D drift correction algorithm on even smaller scaled values of $\theta(0)$ as compared to the 1D model to obtain results which resemble the actual experimentation. Furthermore, since $\boldsymbol{\theta}(k)$ and $\boldsymbol{y}(k)$ are more informative than $\boldsymbol{\Theta}(k)$ and $\boldsymbol{\mathcal{Y}}(k)$ due to the former pair being an image matrix, thus we choose to show them in the following figures.

To generate data, we use a sample image of 4 squares as the spatial molecular density, $\boldsymbol{\theta}(0)$. This is a bitmap image where each white pixel has unit value. This is shown in Figure 3-11. To obtained different amount of sparsity in data, we scale pixel by desired value.

We begin testing by scaling each unit value of $\boldsymbol{\theta}(0)$ to 0.002 and 0.001 which produces 18m/f and 9m/f on average respectively. The results are shown in Figure 3-13. The drift correction works successfully with this frame density but for smaller scaling i.e. 0.001 which produces about 9m/f, the algorithm fails. Furthermore, the computational time is still comparable to BaSDI which is one of the slowest drift correction algorithm. To improve both the speed and performance of 2D drift correction, we add two more operations in the algorithm.

To solve the performance issue, we pre-process the spatial molecular density by smoothing (as in context of image processing) it with a gaussian filter. To intuitively understand how this helps in improving the process, we must understand what the likelihood function does. The likelihood function basically compares the observed data, $\boldsymbol{y}(k)$ with the shifted spatial molecular density, $\boldsymbol{\theta}(k)$ to evaluate the most likely drift value. By gaussian smoothing the spatial molecular density, we highlight the part with higher molecular density as compared to the lower density part in the initial estimate of image, $\boldsymbol{\theta}(0)$[1] producing a more convex optimization problem. Smoothing the image simultaneously also accounts for the localization error as well. A smoothed image is shown in Figure 3-12(b) which otherwise, due to coarse estimate in 3-12(a) leads to non-linearities in the optimization problem.

Furthermore, since we are dealing with small values of spatial molecular density, we also notice that logarithm of a small value is much larger than the value itself, (3-14). Using this, we can safely ignore the last term in (3-12) and rewrite the likelihood function as in (3-15) which

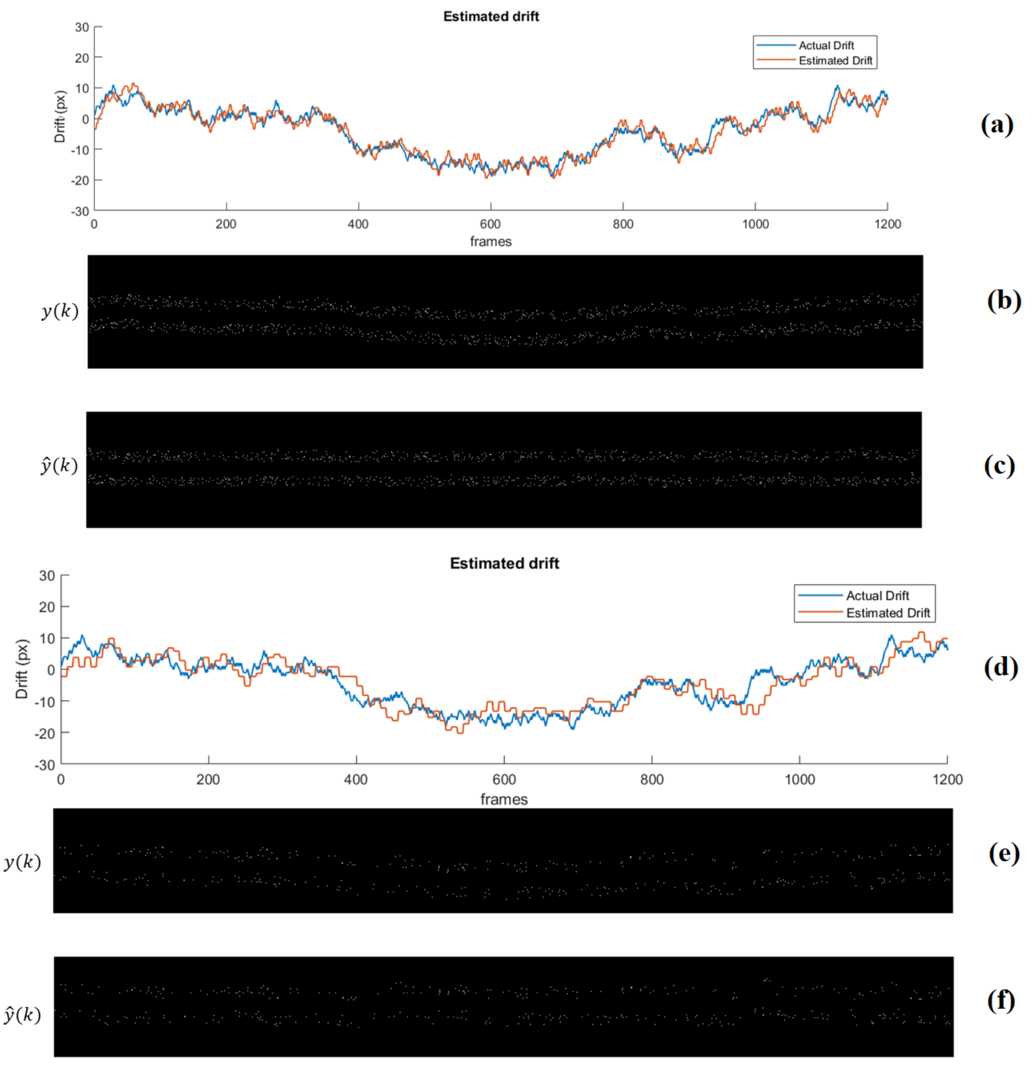

Figure 3-10: Result of extended drift correction algorithm on sparser data generated using $\theta(0)$ consisting of two stripes with value 0.03 in (b) and 0.01 in (e) with an average of 1 molecule per frame in (b) and 1 molecule per 3 frames in (e). For (b), the estimated drift $\hat{d}(k)$ as shown in (a) was obtained using a binning size $s = 3$ and drift set as $[-5, 5]$, i.e. $\bar{d} = 5$. The resulting RMSE error is 2.2134 and the drift corrected data, $\hat{y}(k)$ is shown in (c). Similarly for data in (e), the estimated drift as shown in (d) was obtained using a binning size $s = 8$ and drift set as $[-7, 7]$. The resulting RMSE error is 3.1691 and the drift corrected data, $\hat{y}(k)$ is shown in (c).

decreases the overall computational speed. The new drift compensation algorithm cycle is shown schematically in Figure 3-15. We test this modified algorithm on data generated with scale down value of initial molecular density of 0.001 and 0.0007. The results are shown in Figure 3-14.

$$\log(\boldsymbol{\Theta}(k)) \gg \boldsymbol{\Theta}(k), \quad \text{for small } \boldsymbol{\Theta}(k) \tag{3-14}$$

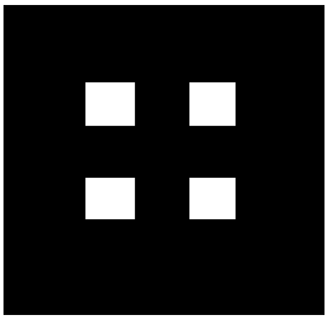

Figure 3-11: Sample image, $\boldsymbol{\theta}(0)$ shape used to generate data for 2D drift correction testing. The image consist of four squares placed equidistant to each other. The resulting spatial molecular density is uniform over the white region. The value of $\boldsymbol{\theta}(0)$ is scaled desirably to control the amount of sparsity in output frames

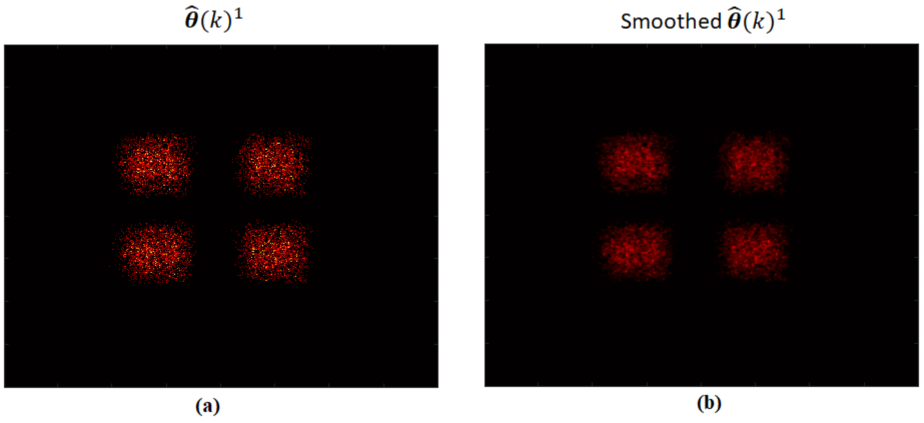

Figure 3-12: (a)Coarse sample image estimate formed using summation of all frames and (b) Gaussian smoothed image highlighting the part with high spatial molecular density

$$\log\left(\mathcal{L}\left(\boldsymbol{\mathcal{Y}}(k)|\widehat{\boldsymbol{\Theta}}(k-1)', d_x(k-1), d_y(k-1)\right)\right) \simeq \sum_i \boldsymbol{\mathcal{Y}}(k)_i \log\left(\left(\boldsymbol{A}_{d_x(k-1),d_y(k-1)}\widehat{\boldsymbol{\Theta}}(k-1)'\right)_i\right) \tag{3-15}$$

3-4-1 Final Algorithm

The algorithm now depends on lot of parameters compared to the ones introduced in the previous section. These include the binning size, s, the drift values over which optimization problem is performed, $\bar{d}$, the window size of the gaussian smoother and the threshold for confirming convergence of drift values. All these parameters effect the resulting estimate of drift in various ways thus it is important what values of the parameters must one select. The choice of parameters depends on different factors.

Binning for example increases the richness of data at the expense of eliminating information about drift from individual frames. This becomes even worse when we have to account for

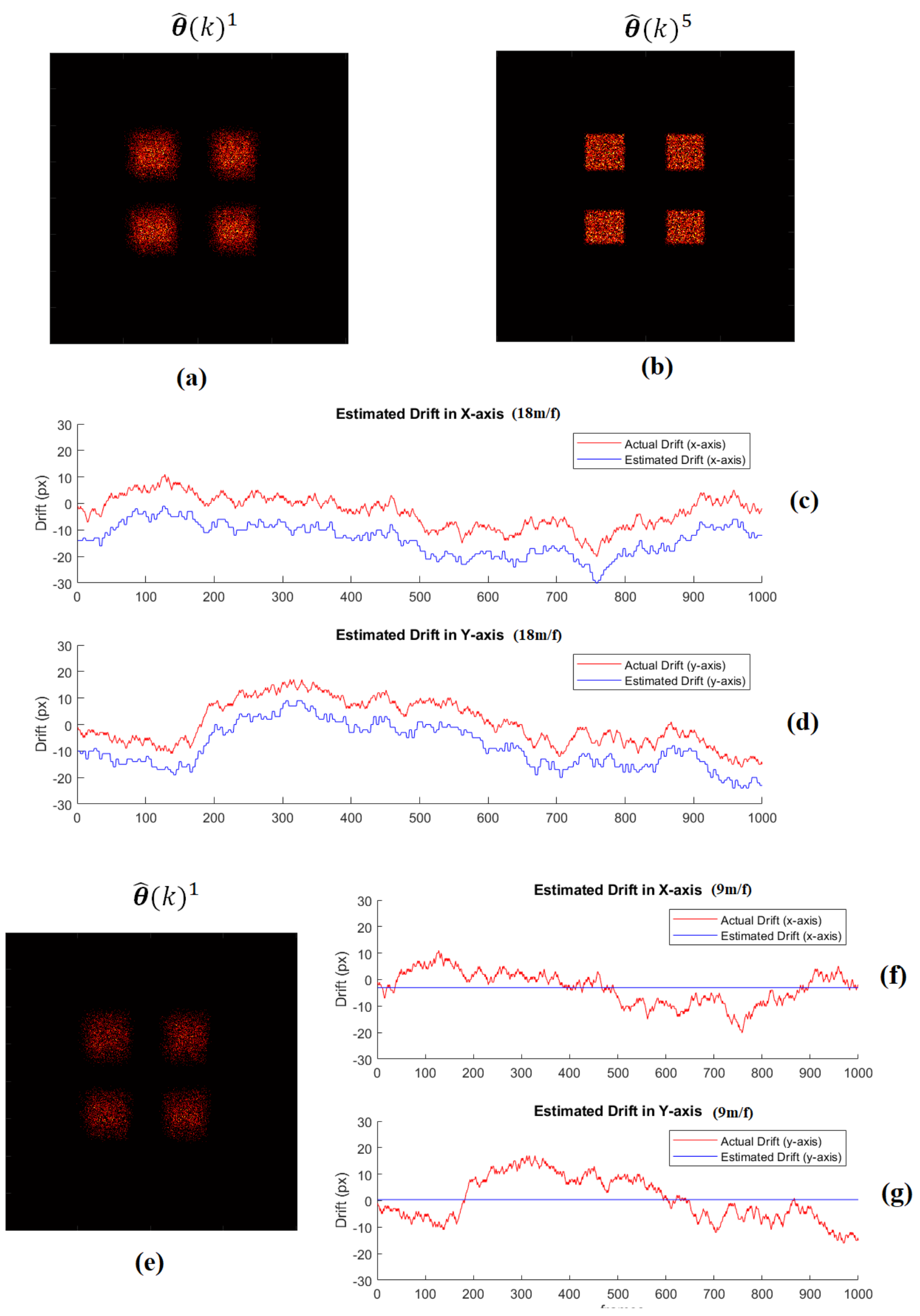

Figure 3-13: Result of 2D drift correction on data generated using 2D state space model as described in (3-7). The output data, $\boldsymbol{y}(k)$ in (a,b,c,d) consist of about 18m/f on average, with each frame having dimension of 400×400 px. The drift is generated using a uniform distribution over $[-1\text{px}, 0\text{px}, 1\text{px}]$. (a) Shows the initial state estimate, $\boldsymbol{\theta}(0)^1$ which is obtained by binning all the output frames, $\boldsymbol{y}(k)$ into single frame. (b) Shows the correct estimate, $\boldsymbol{\theta}(0)^5$ along with the estimated drift for x and y axis in (c) and (d) respectively. A bias is added in the estimate to show the two curves separately. The binning size, $s = 4$ with $\bar{d} = 7$px. Convergence was achieved within 5 iterations using a threshold RMSE of 1.5. For Frame density of 9m/f consisting of (e,f,g), the algorithm fails to converge even after binning and feedback as evident by the drift estimate in (f) and (g). (e) shows the data initial estimate of state, $\boldsymbol{\theta}(0)^1$

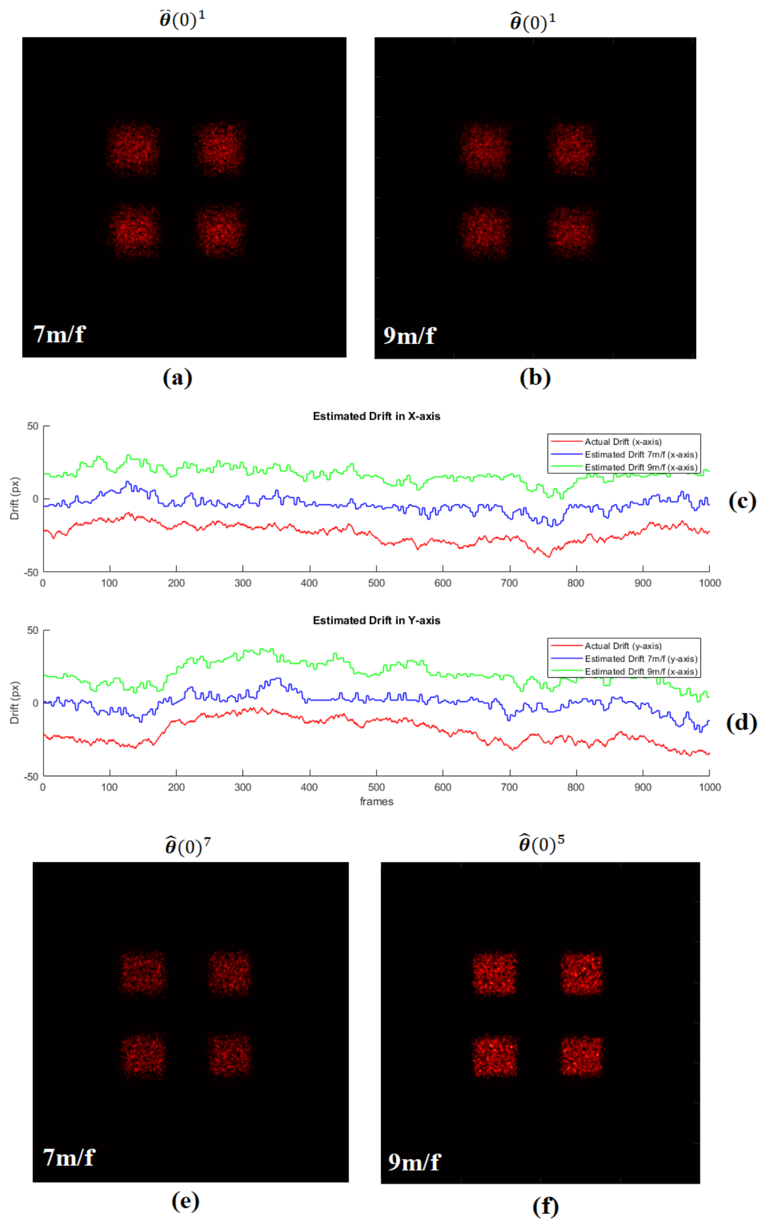

Figure 3-14: Result of 2D drift correction using modified compensation algorithm in Figure 3-15. The output data is generated using the 2D model with 4 squares scaled to 0.001 and 0.0007 which produces on average 9m/f and 7m/f respectively. The dimension of the data is $400 \times 400px$. The ground truth drift is generated using a uniform distribution over $[-1\text{px}, 0\text{px}, 1\text{px}]$. (a) shows the initial state approximation, $\widehat{\boldsymbol{\theta}}(0)^1$ for data of 7m/f. The resulting drift estimate in (c) and (d) is obtained using a binning size, $s = 4$ and $\bar{d} = 11$ having RMSE of 6.3284 and (e) shows the final estimate of initial state, or the sample image. Similarly, (b) shows the initial state approximation, for data of 9m/f. The resulting drift estimate in (c) and (d) is obtained using a binning size, $s = 4$ and $\bar{d} = 7$ having RMSE of 4.5634 and (f) shows the final estimate of initial state, or the sample image.

average on-time of emitters. The reason behind this trade off is quite intuitive. Binning *vir-*

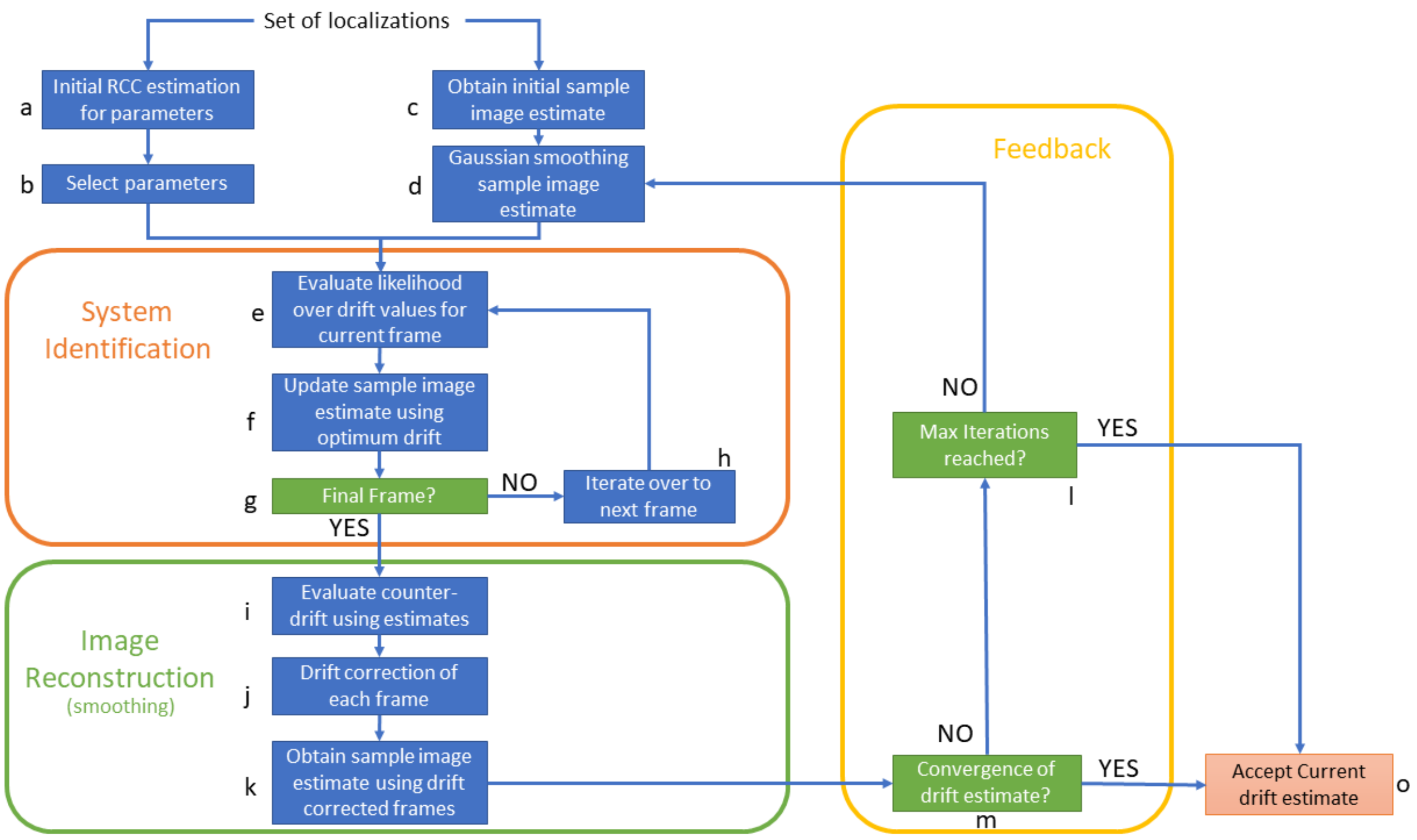
Set of localizations
a
Initial RCC estimation for parameters
b
Select parameters
c
Obtain initial sample image estimate
d
Gaussian smoothing sample image estimate
Feedback
System Identification
e
Evaluate likelihood over drift values for current frame
f
Update sample image estimate using optimum drift
g
Final Frame?
NO
h
Iterate over to next frame
YES
NO
Max Iterations reached?
YES
l
Image Reconstruction (smoothing)
i
Evaluate counter-drift using estimates
j
Drift correction of each frame
k
Obtain sample image estimate using drift corrected frames
NO
Convergence of drift estimate?
m
YES
Accept Current drift estimate
o

Figure 3-15: Schematic representation of drift compensation algorithm based on state space model in 3-9. To select the parameters for drift compensation, we obtain a rough estimate using RCC in (a) and (b). Furthermore, the initial estimate of sample image is smoothed using gaussian blurring (d) and the likelihood is evaluated for each frame one by one to estimate the drift in respective frames through the loop (e-f-g-h-e). The relative drift obtained is used to evaluate counter drift (i), and each frame is treated for drift correction (j) to produce a better estimate of sample image (k). The convergence of estimated drift is tested (m). If the number of iterations reach a maximum value or convergence is achieved, the algorithm stops (o), otherwise, it continues to feedback the reconstructed image to obtain an even better estimate of drift

tually increase the frame molecular density, due to which we have higher number of molecules per frame, but this increase is strictly virtual. By combining the frame together, we ignore the fact that molecule could have moved between these frames, even though not visible. Furthermore, as stated, if the molecule continued to appear in succeeding frames (on-time), we also discard the drift information that the continued blinking of that certain molecules would have given us. This is empirically discussed further in Chapter 5. Generally, deciding the correct binning size is a matter of trial and error since all datasets differ from each other.

The drift values over which optimization is performed, $\bar{d}$ could be decided fairly by assessing the actual drift curve. Since in most cases, little to no information is available about ground truth drift, we must rely on other ways for obtaining a fast initial estimate of drift. For example, RCC with a large binning size gives a fair estimate of the shape of drift curve using which, one could decide the value of $\bar{d}$. The size of the smoothing window depends on the size of the sample structure and sample image size. It should be about some fraction of the size of the image. Furthermore, the threshold for drift convergence is also a matter of trial and error. It could differ from dataset to dataset but empirically, we could state that a small value (0.3-0.5) is best. The final SMSS drift compensation algorithm pipeline is shown in Figure 3-15 with different sub-algorithms. This drift compensation method is tested against other algorithms in the Chapter 5 for computational time and performance.

3-5 2D drift correction using 1D model

In the last section, we developed both the 1D model and the 2D model along with the system identification and image reconstruction algorithm as well. It should be obvious that 1D drift correction is faster as compared to the 2D correction since the computational complexity of the former is on the order of O($\frac{N}{s} \times n$) whereas for the latter it is O($\frac{N}{s} \times n^2$) where n is the dimension of the state space, N is the number of frames and s is the binning size. A question we should ask is, if it is possible to treat the drift correction of 2D image as two separate cases of 1D drift correction.

It is possible to treat the 2D drift correction as two separate case of 1D correction [3]. We proceed first by evaluating the sum of image matrix, $\boldsymbol{\theta}(0)$ along both the axis (row and column) which could be achieved using `sum` command in MATLAB, resulting in $\theta(0)_x$ and $\theta(0)_y$. This is similar to marginalizing a probability distribution. The same operation of summing along axis is also performed on all the output frames, $\boldsymbol{y}(k)$ to obtain $y_y(k)$, $y_y(k)$, representing drifted data in x and y axis respectively. Then we could proceed to apply the 1D drift correction algorithm on both dataset for respective axis. The resulting drift is used in the 1D Image reconstruction algorithm and the loop is repeated until convergence is achieved. The final estimate of drift from both axis is used to reconstruction the 2D image. By treating the 2D drift compensation problem as two separate cases of 1D, we would for sure loose some information, so a subsequent question would be how much information do we loose in treating the 2D drift correction case as 1D and what is the affect of that loss. These questions are answered in Chapter 5.

Chapter 4

State Space Modelling using Emitter Position

To develop an alternative state space description of drift in SMLM, we consider another property of this process, namely, the position of emitter molecules. The position of emitter molecules is obtained from the experiments through the process of localization in which we estimate the position of the emitter molecules from the Point Source Function (PSF) generated by it [11]. This estimation is accomodated with an error called *localization error*. This error differs for each emitter but could be well described with a normal distribution having a particular mean and standard deviation. Before introducing notations, please note that the notations in this chapter have no relation with previous chapter.

Similar to our previous modelling steps, let us begin by considering 1D drift only. Let our state, $x(k) \in \mathbb{R}^N$ represent the position of all N molecules observed throughout the image capturing and localization process. Another fair assumption is that drift is dictated by a random walk model. This implies that the position of each emitter molecules will be influenced by a zero mean normally distributed noise term, $w(k) \in \mathbb{R}^N$. On the other hand, the output $y(k)$, consist of molecules visible in frame k. These molecules are a subset of $x(k)$ which means that the operation of extracting a subset of $x(k)$ could be described through multiplication with $C(k)$ which is a diagonal matrix containing 1's and 0's for emitter which are ON and OFF respectively. To account for the localization error, the ON molecules would be contaminated by ZMWN, $v(k)$. The state space is summarized in (4-1) and (4-2).

<u>State Dynamics</u>:

$$\begin{aligned} x(k+1) &= x(k) + w(k) \text{ where} \\ w(k) &\sim \mathcal{N}\left(\overrightarrow{0}_N, \sigma_w^2 \boldsymbol{I}_N\right) \end{aligned} \tag{4-1}$$

<u>Output Dynamics</u>:

$$y(k) = C(k)x(k) + C(k)v(k) \text{ where} \tag{4-2}$$

$$v(k) \sim \mathcal{N}\left(\overrightarrow{0}_N, \sigma_v^2 \boldsymbol{I}_N\right)$$

with $\overrightarrow{0}_N$ is a vector containing N zeros and $\boldsymbol{I}_N$ is an identity matrix of size N. One should also note that the output equation (4-2) consist of multiplication with matrix $C(k)$ with both terms, $x(k)$ and $v(k)$ since localization error is only applicable to the emitter molecules turning ON in frame k.

Ideally, we would proceed by performing system identification to find out the matrices $C(k)$ and then performing smoothing using RTS smoother to find the states, $x(k)$ which we can use to find the drift. Apparently, this model carries huge drawbacks. To elaborate them, notice that the output vector, $y(k)$ must contain the position of the molecules turning ON and, for those molecules which stay turned OFF, the corresponding index in the vector $y(k)$ should have zero value. For this, we will need to know precisely which molecule appears in what frame. Such information is very difficult to attain since there is no way to know if the emitter molecule blinking in one frame is the same as blinking in other frame or not. Furthermore, the average number of emitter molecules and number of frames in an SMLM experiment is about $10,000$ and $20,000$ respectively. This means for each time step of $20,000$ frames, $C(k)$ would have dimensions $10,000 \times 10,000$. Even if $C(k)$ is sparse matrix, we would have to find $10,000 \times 20,000$ parameters using System identification which is quite impractical.

Since the dimension explosion creates a big hurdle in the stated model, we must modify or come up with an alternative state space description which has reduced dimension and therefore, lesser number of parameters. To do so, we introduce an important step called linking.

4-1 Linking

Although it might not be possible to label the emitter molecules precisely and know when and in which frame they blink and when they do not, we could approximate this information. The process of pairing these emitter molecules is called *Linking*. To elaborate this process, consider two frames with certain number of molecules in them as in Figure 4-1(a). Let frame 1 contain molecules shown in blue and frame 2 contain molecules shown in orange.

To carry out the linking process, we must select a metric to label the molecules. We choose l_2 norm which is basically radial threshold, σ_r around a center point. This radial threshold is of fixed value. Using the position of emitter molecules appearing in frame 1 as the center point, we attempt to identify if the emitter molecules appearing in next frame are within or outside the radial threshold, σ_r. If the emitter molecules are within the bound, *we assume that it is the same molecule*, only appearing in next frame. If not, then it is altogether a different molecule. Furthermore, if multiple molecules are detected within the specified linking radius, then the emitter molecules lying closest is assigned as a pair.

An important question here is how do we select the value of radial threshold (or the linking radius) since through it, we conclude if the emitter molecules in succeeding frame is same or not. A good estimate of this threshold would combine the information from the relative drift distribution and the distribution of the localization error. This is because we expect correct linkings to have drifted (dictated by the distribution of drift) along with mis-located (which is governed by distribution of localization error). In our analysis we assume the linking

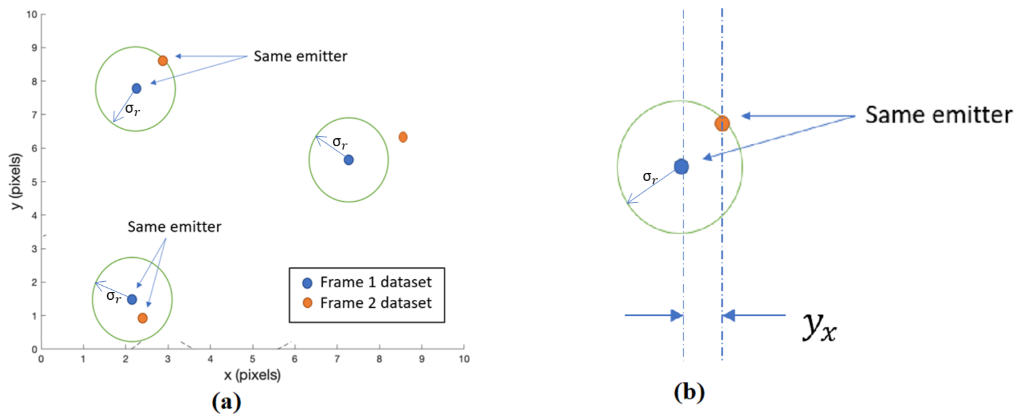

Figure 4-1: Illustration of the process of Linking in which we identify and label if the molecules appearing the succeeding frames are same or not.

radius to be almost equal to the sum of standard deviation of distribution of drift and that of localization error. In most experimental data, the distribution of relative drift in not available. This could be roughly evaluated by obtaining an initial estimate of drift using RCC. Further investigation and discussion on linking radius is reserved for Chapter 5.

Using the labeled emitter molecules, we evaluate the horizontal and vertical displacement that occurred from one frame to another. For example, the horizontal displacement observed in one of the pair is shown in Figure 4-1(b). With each succeeding pair of frame, there would be a different number of molecules linked together. Let m_{max} represent the maximum number of molecules linked in any given succeeding pair of frames. This value will dictate the dimension of the output observed in new state space model.

4-2 Modified Model : Position Linking State Space

The resulting data from the linking process, which consist of relative displacement of the linked emitter molecules is the new observed/output, $\mathbf{y}(k) \in \mathbb{R}^{2m_{max}}$ of the alternative state space model we aim to construct in this subsection. Let us call this model Position Linking State Space (PLSS). In each pair of succeeding frames, the number of linked pairs are represented by $m(k)$. The output, $\mathbf{y}(k)$ represents the displacement of different linked molecules in frame k and frame $k+1$ and so on. The dimension is $2m_{max}$ to represent the data in both the axis i.e. 2 times m_{max}. Each vector $\mathbf{y}(k)$ would consist of $2m(k)$ non-zero entries and $2\left(m_{max} - m(k)\right)$ zeros entries from top to bottom. We can then define our state as the relative drift, $d(k) \in \mathbb{R}^2 = [d_x(k), d_y(k)]^\top$ being frame to frame drift taking place between frame k to frame $k+1$ in both lateral axis.

State Dynamics:

$$d(k+1) = Ad(k) + w(k) \quad \text{where} \tag{4-3}$$
$$w(k) \sim \mathcal{N}\left(\overrightarrow{0}_2, \sigma_w^2 \boldsymbol{I}_2\right)$$

Output Dynamics:

$$\mathbf{y}(k) = C(k)d(k) + N(k)v(k) \text{ where} \quad (4\text{-}4)$$
$$v(k) \sim \mathcal{N}\left(\overrightarrow{0}_{m_{max}}, \sigma_v^2 \boldsymbol{I}_{m_{max}}\right)$$

with $w(k) \in \mathbb{R}^2$ and $v(k) \in \mathbb{R}^{m_{max}}$. The output matrix $C(k)$ is obtained from the number of molecules linked in frame k and $k+1$, denoted by $m(k)$. It consist of 1's and 0's at index corresponding to the molecules which are linked. The Noise matrix, $N(k)$ is also obtained from the linking process and is a diagonal matrix which imparts localization error on the linked pair of emitter molecules only. It contains same number of 1's as equivalent to twice the number of non-zeros entries in the observed output, $\mathbf{y}(k)$. An example output vector having $m(k) = 2$, i.e. number of linked molecules pairs is 2, the output vector, $\mathbf{y}(k)$ with corresponding $C(k)$ matrix and Noise matrix, $N(k)$ is shown in (4-5) for $m_{max} = 3$.

$$\mathbf{y}(k) = \begin{bmatrix} y_{1,x}(k) \\ y_{2,x}(k) \\ y_{1,y}(k) \\ y_{2,y}(k) \\ 0 \\ 0 \end{bmatrix}, \quad C(k) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad N(k) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (4\text{-}5)$$

with $y_{1,x}(k)$ is the horizontal displacement of first pair of emitter molecules and $y_{1,y}(k)$ is the vertical displacement of the same pair. The advantage of this state space model is that it has significantly lesser dimension as compared to the former model. Due to this, the computational resources required to obtain drift, $d(k)$ is significantly low. But one important attribute of this model is that it has non-stationary output noise due to the combination of $N(k)$ an $v(k)$. This is discussed in following section.

4-3 Drift Compensation algorithm using Position Linking State Space

Similar to shift state space model, we would like to find out the parameters of the model derived in this chapter, specifically A matrix and also find out the process (a.k.a hidden) states of the model, $d(k)$. The process state is the drift and yet again, system identification and smoothing algorithm give rise to the drift compensation algorithm using PLSS. We discuss the development as follows.

4-3-1 System Identification using PEM

Prediction Error Methods (PEM) are used to find the system matrices for models having both deterministic and stochastic part. For example, our aim here is to find the matrix A in (4-3) from the data obtained through linking. This is the deterministic part of the system while the process and output noise form the stochastic part of the system. Our aim is to first parameterize the system matrices, of a suitable equivalent model and then form an optimization problem to find the best value of the parameters using the information we have.

The information available to us includes $\mathbf{y}(k)$, $C(k)$ and $N(k)$. Since we do not have information about the noise, $w(k)$ and $v(k)$, so we ought to form an equivalent model which could approximately represent these noise terms. Using kalman filter theory, the most suitable equivalent model for (4-3) and (4-4) is the innovation state space model in (4-6).

$$\begin{aligned} \hat{d}(k+1) &= A\hat{d}(k) + K(k)e(k) \\ \mathbf{y}(k) &= C(k)d(k) + e(k) \end{aligned} \tag{4-6}$$

with $e(k)$ as the innovation signal which is ZMWN. The problem with using this innovation model for system identification is twofold. First, it has a time varying kalman gain, $K(k)$, i.e. for each time step, we would have a different Kalman gain which should be estimated. This is a very hard problem to solve without sufficient dimension of the output signal, $\mathbf{y}(k)$. Secondly, the innovation signal $e(k)$ is used to represent a ZMWN signal, but the noise term we have in our system have non-stationary distribution since the covariance of output noise varies with time due to the term $N(k)$.

To work around both the problems, we slightly modify the output dynamics (4-4), resulting in the following.

State Dynamics:

$$d(k+1) = Ad(k) + w(k) \tag{4-7}$$

Output Dynamics:

$$\begin{aligned} C(k)^{\dagger}\mathbf{y}(k) &= d(k) + C(k)^{\dagger}N(k)v(k) \\ \boldsymbol{y}(k) &= d(k) + \tilde{v}(k), \\ \text{where } \tilde{v}(k) &= C(k)^{\dagger}N(k)v(k), \text{ and} \\ \boldsymbol{y}(k) &= C(k)^{\dagger}\mathbf{y}(k) \end{aligned} \tag{4-8}$$

with $C(k)^{\dagger}$ as the pseudo-inverse of matrix $C(k)$. Also note that $\boldsymbol{y}(k) = C(k)^{\dagger}\mathbf{y}(k)$. We transferred the $C(k)$ matrix from the state term on the right side of (4-8), but the system still remains time varying due to the output noise term. The noise is non-stationary because of it's time varying covariance matrix by multiplication with the term $C(k)^{\dagger}$. It could be shown that for small enough localization error, i.e. small σ_v^2, we may assume that the non-stationary output noise signal behaves like a stationary signal. The reason behind this assumption is discussed in the next subsection.

The resulting innovation model for this modified system is given as follows.

$$\begin{aligned} \widehat{d}(k+1) &= A\widehat{d}(k) + Ke(k) \\ \boldsymbol{y}(k)(k) &= d(k) + e(k) \end{aligned} \tag{4-9}$$

with $\boldsymbol{y}(k)(k) = C(k)^{\dagger}\mathbf{y}(k)$ as the new output data. Note that the new innovation model contains a time invariant kalman gain. Using this model, the one step ahead prediction for the innovation model in (4-9) is given by (4-10).

$$\begin{aligned}\widehat{d}(k+1|k) &= (A-K)\widehat{d}(k|k-1) + K\tilde{\mathbf{y}}(k)\\ &= \tilde{A}\widehat{d}(k|k-1) + K\tilde{\mathbf{y}}(k)\\ \widehat{\boldsymbol{y}}(k)(k|k-1) &= d(k|k-1)\end{aligned} \tag{4-10}$$

with $\tilde{A} = (A-K)$. Furthermore, we also parameterize the A and K matrices with parameters $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \dots, \alpha_8\}$ as follows.

$$A = \begin{bmatrix}\alpha_1 & \alpha_2\\ \alpha_3 & \alpha_4\end{bmatrix}, \quad K = \begin{bmatrix}\alpha_5 & \alpha_6\\ \alpha_7 & \alpha_8\end{bmatrix} \tag{4-11}$$

Next, we must form an optimization problem to find the best estimate of these parameters using the data. The cost function utilized is the $l-2$ norm of the predicted output, $\widehat{\boldsymbol{y}}(k|k-1)$ as shown follows

$$J_N(\boldsymbol{\alpha}) = \frac{1}{N}\sum_{k=0}^{N-1} \|\tilde{\boldsymbol{y}}(k) - \widehat{\boldsymbol{y}}(k|k-1,\boldsymbol{\alpha})\|_2^2 \tag{4-12}$$

where $\widehat{\mathbf{y}}(k|k-1,\boldsymbol{\alpha})$ is obtained by using the one-step ahead prediction equation in (4-10) and the initial state $d(0) = [0,0]^\top$.

$$\begin{aligned}\widehat{d}(k|k-1,\boldsymbol{\alpha}) &= \tilde{A}^k d(0) + \sum_{\tau=0}^{k-1} \tilde{A}^{k-1-\tau} K\boldsymbol{y}(\tau)\\ \widehat{\boldsymbol{y}}(k|k-1,\boldsymbol{\alpha}) &= \widehat{d}(k|0)\end{aligned} \tag{4-13}$$

Using (4-13), (4-12) and (4-11) we aim to find the argument which minimizes the cost function,

$$\widehat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \left(\frac{1}{N}\sum_{k=0}^{N-1} \|\boldsymbol{y}(k) - \widehat{\boldsymbol{y}}(k|k-1,\boldsymbol{\alpha})\|_2^2\right) \tag{4-14}$$

The above optimization problem is a unconstraint non-linear programming which could be solved using `fminunc` MATLAB command. We would have to provide suitable initial values of the parameters, $\boldsymbol{\alpha}$ to the function. It is possible to evaluate a fine initial estimate for $A(\alpha)$ but not for $K(\alpha)$. To understand how we would evaluate the initial estimate of $A(\alpha)$, we will make a poor yet fair assumption that the value of state, $d(k) \simeq \tilde{\mathbf{y}}(k)$. Let us call this approximation, $\tilde{d}(k)$. Using equation (4-9) where the innovation signal, $e(k) = 0$ since $\tilde{d}(k) = \tilde{\mathbf{y}}(k)$, we must find out the approximation of state dynamics matrix $\tilde{A}(\alpha)$ such that, for all k

$$\tilde{d}(k+1) = \tilde{A}(\alpha)\tilde{d}(k), \quad \forall k \tag{4-15}$$

which could be boiled down to a least square/regression problem as follows.

$$\begin{aligned}\begin{bmatrix}\tilde{d}_1(1) & \tilde{d}_1(2) & \tilde{d}_1(3) & \dots & \tilde{d}_1(N)\\ \tilde{d}_2(1) & \tilde{d}_2(2) & \tilde{d}_2(3) & \dots & \tilde{d}_2(N)\end{bmatrix} &= \begin{bmatrix}\alpha_1 & \alpha_2\\ \alpha_3 & \alpha_4\end{bmatrix}\begin{bmatrix}\tilde{d}_1(0) & \tilde{d}_1(1) & \tilde{d}_1(2) & \dots & \tilde{d}_1(N-1)\\ \tilde{d}_2(0) & \tilde{d}_2(1) & \tilde{d}_2(2) & \dots & \tilde{d}_2(N-1)\end{bmatrix}\\ \tilde{\boldsymbol{d}}_{1,N} &= \tilde{A}(\alpha)\tilde{\boldsymbol{d}}_{0,N-1}\end{aligned} \tag{4-16}$$

for which, the solution to the problem is as follows

$$\tilde{A}(\alpha) = \left(\tilde{\boldsymbol{d}}_{0,N-1}\right)^\dagger \tilde{\boldsymbol{d}}_{1,N}$$

where the supercript † is used to represent the pseudo-inverse. This $\tilde{A}(\alpha)$ could be used as an initial value for the optimization problem in (4-14). As stated, the initial value of matrix K cannot be evaluated since the innovation signal, $e(k) = 0$ when $\tilde{d}(k) = \tilde{\mathbf{y}}(k)$, therefore we assign an arbitrary initial value.

4-3-2 Non-stationary noise analysis

The modified state space equations in (4-8) contain non-stationary output noise due to the multiplication with time varying matrix $C(k)^\dagger$ and $N(k)$. The general structure of matrix $C(k)$ is shown below.

$$C(k) = \begin{array}{r} m(k)\left\{\begin{array}{c} \\ \\ \\ \\ \end{array}\right. \\ m(k)\left\{\begin{array}{c} \\ \\ \\ \\ \end{array}\right. \\ m_{max}-2m(k)\left\{\begin{array}{c} \\ \\ \\ \\ \end{array}\right. \end{array} \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix} \tag{4-17}$$

Since the original output noise $v(k)$ has zero mean, the mean of the non-stationary noise stays unaffected. What changes at every time step k is the variance of the noise. We will investigate the bounds of this time varying variance of the non-stationary noise and find out the condition under which the non-stationary noise signal could be assumed to be fairly stationary.

To begin our analysis, we first describe the structure of the matrices $C(k)$, $C(k)^\dagger$ and $N(k)$. A sample of these matrices is shown in (4-5). Due to their structure, these matrices and their product follow special properties as discussed below.

Property 1: $C(k)^\dagger N(k) = C(k)^\dagger$

Explanation:

$$C(k)^\dagger = \left(C(k)^\top C(k)\right)^{-1} C(k)^\top$$

which means that

$$C(k)^\dagger N(k) = \left(C(k)^\top C(k)\right)^{-1} C(k)^\top N(k)$$

The key here is to realize that

$$C(k)^\top N(k) = C(k)^\top$$

This could be understood by peeking into the structure of these matrices.

$$
\underset{C(k)^\top}{\left[\begin{array}{cccccccc|cc}1 & 1 & \dots & 1 & 0 & 0 & \dots & 0 & 0 & 0\\ 0 & 0 & \dots & 0 & 1 & 1 & \dots & 1 & 0 & 0\end{array}\right]}
\quad
\underset{N(k)}{\left[\begin{array}{ccccccc|cc}1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & \dots & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & \dots & 0 & 0 & 0\\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \\ 0 & 0 & 0 & 0 & 0 & \dots & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0\end{array}\right]}
$$

Figure 4-2: Illustration of $C(k)^\top N(k) = C(k)^\top$

The highlighted green part in Figure 4-2 represents the non-zero entries of both the matrices. As is evident, the non-zero entries of the $N(k)$ matrix form an identity matrix with the exact same size as the number of non-zero columns in $C(k)^\top$.

Property 2: The $C(k)^\dagger = \frac{1}{m(k)} C(k)^\top \quad \text{for} \quad m(k) > 0$

$$
C(k)^\dagger = \begin{cases} \frac{1}{m(k)} \begin{bmatrix} 1 & \dots & 1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 & 0 & \dots & 0 \end{bmatrix} & m(k) > 0 \\ \begin{bmatrix} 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 \end{bmatrix} & m(k) = 0 \end{cases}
$$

with $m(k)$ the number of linked molecules corresponding to data $\boldsymbol{y}(k)$. The non-zero columns containing unit in the above matrix is equal to $2m(k)$. The first $m(k)$ columns of the first row contain ones and the next $m(k)$ columns in the second row also contain ones. Rest of the entries have zeros.

Using Property 1 and Property 2, we can state that the non-stationary output noise is given as

$$
C(k)^\dagger N(k) v(k) = C(k)^\dagger v(k) = \begin{cases} \frac{1}{m(k)} \begin{bmatrix} 1 & \dots & 1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} v_1(k) \\ v_2(k) \\ v_3(k) \\ \vdots \\ v_{m_{max}}(k) \end{bmatrix} & m(k) > 0 \\ \begin{bmatrix} 0 \\ 0 \end{bmatrix} & m(k) = 0 \end{cases}
$$

where $v_i(k) \sim \mathcal{N}\left(0, \sigma_v^2\right) \forall i$ which means that all v_i are random variables having same distributions. Focusing on the case where $m(k) > 0$, we have the resulting noise terms,

$$C(k)^{\dagger}v(k) = \frac{1}{m(k)} \begin{bmatrix} \sum_{i=1}^{m(k)} v_i(k) \\ \sum_{i=1+m(k)}^{2m(k)} v_i(k) \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}, \qquad m(k) > 0$$

Now, we will find the distribution of the resulting random variables. Let us evaluate the distribution of the first element, z_1 using fundamental properties of expectation and variance.

$$\begin{aligned} z_1 &= \frac{1}{m(k)} \sum_{i=1}^{m(k)} v_i(k) \\ \mathbb{E}(z_1) &= \frac{1}{m(k)} \sum_{i=1}^{m(k)} \mathbb{E}(v_i(k)) = 0 \\ Var(z_1) &= \frac{1}{m(k)^2} \sum_{i=1}^{m(k)} Var(v_i(k)) \\ &= \frac{Var(v_i(k))}{m(k)}, \quad \text{since } Var(v_i(k)) = \sigma_v^2 \ \forall i \end{aligned} \tag{4-18}$$

For $m(k) = 0$, the variance, $Var(z_1) = 0$. Furthermore, the distribution of the second element, z_2 is the same as z_1.

To evaluate the upper limit and the lower limit of the variance, we use the expression in (4-18). The lower bound on variance denoted by the subscript L, $Var(z_1)_L = 0$ is given when $m(k) = 0$ for all frames, (i.e. $\forall k$). The upper bound is, $Var(z_1)_U = \sigma_v^2$ when $m(k) = 1$ for all frames.

$$\begin{aligned} Var(z_1)_L &= 0, \ \text{ and} \\ Var(z_1)_U &= \sigma_v^2 \end{aligned} \tag{4-19}$$

The above analytical results are confirmed using ensemble variance for different generated datsets with a defined value of σ_v^2. To evaluate the lower bound, we generate 100 realizations of data with $m(k) = 0 \ \forall k$ using (4-8) and then evaluate the ensemble variance. Similarly, for the upper bound, we generate 100 realizations with $m(k) = 1 \ \forall k$. Furthermore, we generate 15 random sequence of $m(k)$. For each sequence, we generate 100 realizations and then we evaluate the variance. This dataset resembles actual experimentation. The results are shown in Figure 4-3 (a). We also evaluate ensemble variance for different values of $m(k)$ by generated 100 realizations for each value of $m(k)$ (and therefore $C(k)$) as shown in Figure 4-3 (b) along with the analytical values obtained from (4-18).

With this observation, we can conclude that for localization error, $v(k)$ having considerably small variance, σ_v^2, the output non-stationary noise, $\tilde{v}(k)$ in (4-8) roughly behaves like a stationary noise signal. Data with such characteristic is more suitable for system identification algorithm discussed previously as compared to one having localization error with higher variance.

For the verification of the system identification algorithm, we generate 20 realizations with 1000 samples using a specific A matrix containing complex eigenvalues. The number of

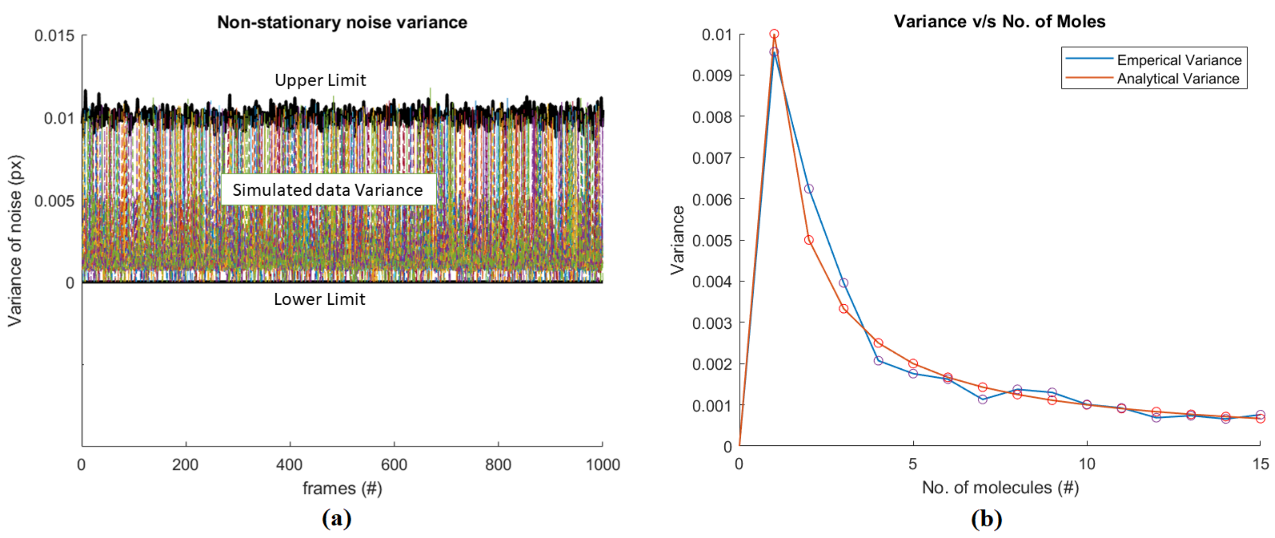

Figure 4-3: The upper and lower limit of the Variance for non-stationary noise is shown here. The datasets are generated using 1000 samples of the non-stationary noise term $\boldsymbol{v}(k) = C(k)^{\dagger}v(k)$ with ZMWN localization error, having $\sigma_v^2 = 0.01$. Using (4-18), the upper bound on the non-stationary noise term is given as $Var(\boldsymbol{v}(k))_U = \sigma_v^2$ and the lower bound is given by $Var(\boldsymbol{v}(k))_L = 0$. This is confirmed in (a) where the upper and lower limit of variance is obtained by evaluating the ensemble variance shown in black curve, of non-stationary noise generated with $m(k) = 1$ and $m(k) = 0$ respectively for all frames,k. To show that the variance of non-stationary noise lies within these limits, 15 different datasets are generated each containing 1000 realizations. These dataset have random number of linked molecules generated using a normal distribution over $[0, m_{max}]$. As one could see, the ensemble variance of these simulated datasets lies within the upper and lower limit. Furthermore, we also generated 100 realization of non-stationary noise for $m(k) = 0, 1, 2, \ldots, 15$. The ensemble variance for each $m(k)$ is shown in (b) along with the analytical variance for each $m(k)$. As stated previously, the maximum variance is obtained for $m(k) = 1$ and lowest for $m(k) = 0$.

emitter molecule linked pairs in each frame is generated using a uniform distribution over $[0, m_{max}]$ with m_{max} as the maximum number of emitter molecule pair. The matrix $C(k)$ is generated using this information. The process noise $w(k) \sim \mathcal{N}\left(\overrightarrow{0}_2, \sigma_w^2 \boldsymbol{I}_2\right)$ and output noise $v(k) \sim \mathcal{N}\left(\overrightarrow{0}_{m_{max}}, \sigma_v^2 \boldsymbol{I}_{m_{max}}\right)$. The resulting eigenvalues of the estimated matrix, $\hat{A}$ are plotted against the actual eigenvalues. As shown in Figure 4-4 (a), varying output noise variance to a higher value produces less precise results which has been concluded from the non-stationary noise analysis above.

4-3-3 Kalman Smoother

To evaluate an optimum estimate of the states of a state space model such as one for drift in (4-3) and (4-4) we use so called *Filters* or *Smoothers*. There are many filters which are used to deal with noise of certain characteristics and extract the underlying information (for e.g. states). A low pass filter-for example-eliminates noise having the Power Spectral Density (PSD) below a certain magnitude. Band-pass filter removes noise lying within a certain frequency band. However these filter are not optimum, in the sense that they do not

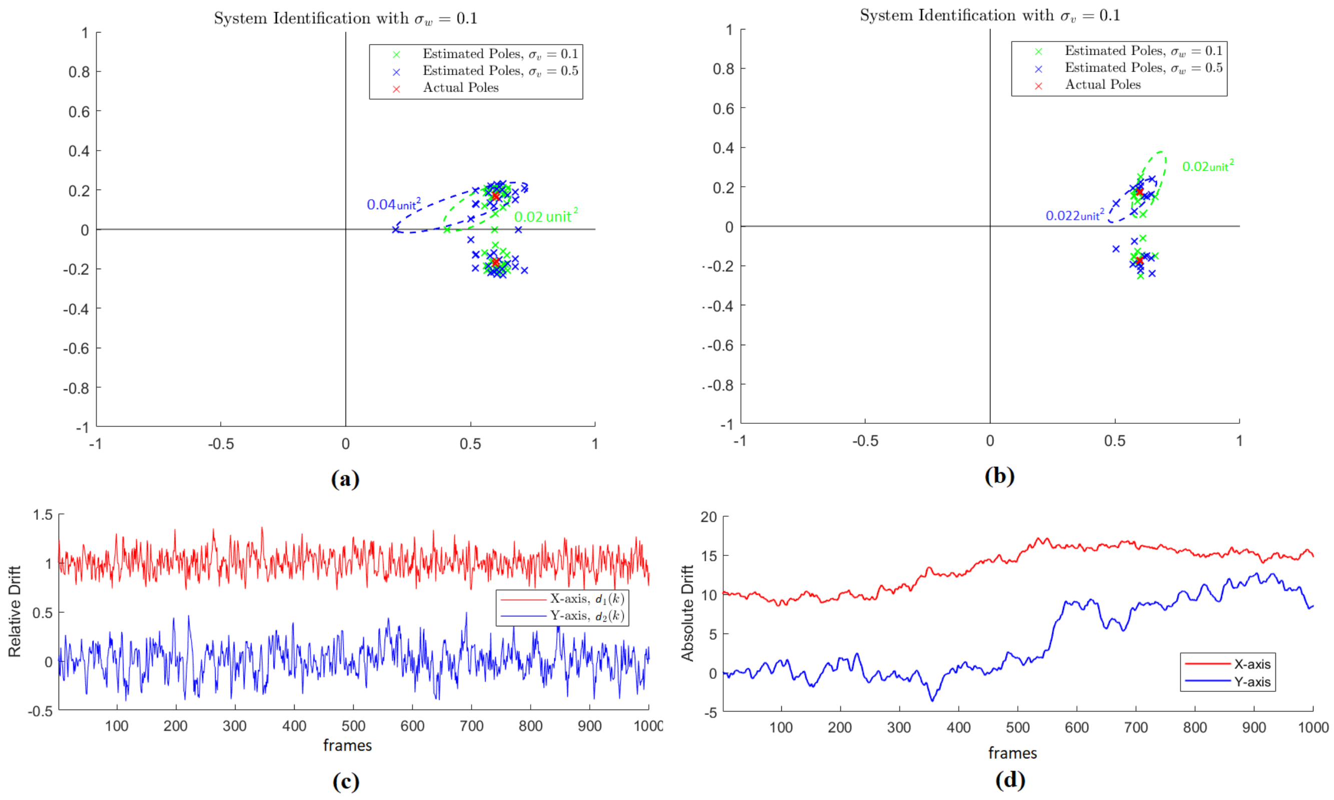

Figure 4-4: Results of System Identification using the PEM model with different process and output noises. 20 realizations of 1000 samples are generated using the model in (4-3) and (4-8) having $A = \begin{bmatrix} 0.5 & 0.1 \\ -0.4 & 0.7 \end{bmatrix}$ with eigenvalues $0.6 \pm 0.1732j$. The system identification algorithm is used to estimate the system dynamics matrix $\hat{A}$ for each realization. (a) and (b) shows the result with varying output noise and process noise respectively. As stated previously, the system Identification algorithm performs better with output noise having low variance ($\sigma_v = 0.1$) as compared to high variance ($\sigma_v = 0.5$), shown in (a). The algorithm is robust with varying process noise shown in (b) with very good precision, to the extend the one complex pole is almost hidden underneath the estimated poles. (c) shows the states, $d_1(k)$ and $d_2(k)$ of single realization generated using the stated A matrix and process noise variance, $\sigma_w = 0.1$. This is relative drift and (d) shows the cumulative sum of the states which is the absolute drift

necessarily produce the best estimate of the desired signal/state. Optimum filter on the other hand, provide the best estimate of the desired signal.

In general, filtering is the process of obtaining a better estimate of state or signal using observation available up-til the current time-step, k. This is the classic problem considered by Wiener [12] in which we have signal/state are corrupted by ZMWN, and the goal is to estimate the signal using a causal filter, i.e., using current and past observations. Smoothing is the same as the filtering problem except that the filter is allowed to be noncausal, i.e. using observation of past, present and future. This is similar to the forward-backward algorithm. In many situations, for e.g. super resolution microscopy and satellite imaging, smoothing is more suitable since we have observation of the whole experiment and we would like to use it all to evaluate the best estimate. Smoothing is suitable as a post processing technique. Furthermore, forward-backward algorithm is a derivative of smoothing algorithm itself.

There are three different ways to perform smoothing *Fixed point*, *Fixed Lag* and *Fixed Interval*. Since we have all the observations in the interval $k = 0$ to $k = N$, we will use the fixed interval smoothing. There are two methods of Fixed interval smoothing. One of which is theoretically straight forward method which relies on evaluating the forward part and backward part separately and then combining them. This is similar to the forward-backward algorithm described briefly in section 2-2-3. The other counterpart called *RTS algorithm*, which is conceptually more difficult but is computationally cheaper than forward-backward smoothing.

The mathematical treatment of both these smoothers is extensive and could be found in [12]. The key idea is to use a forward filter (basic kalman filter) along with a reverse filter which relies on the one time step back dynamic recursion [25]. The Fixed interval smoothing algorithm (or RTS algorithm) is described in (Al. 1).

To apply the Fixed interval smoothing algorithm, we require information about the initial state, $d(0)$, state space matrices, the covariance matrix of the initial state, $P_f(0|0)$, covariance of process noise, Q and output noise, R. All the information is available except the process and output noise covariances. To obtain this covariance matrices, we utilize (4-6). Since the innovation model in (4-6) is an equivalent representation of the actual model in (4-3) and (4-4), we may conclude that

$$\begin{aligned} Var\left(w(k)\right) &\simeq Var\left(K(k)e(k)\right) = Var\left(\tilde{d}(k+1) - A\tilde{d}(k)\right) \\ Var\left(N(k)v(k)\right) &\simeq \left(e(k)\right) = \left(y(k) - C(k)\tilde{d}(k)\right) \end{aligned} \tag{4-20}$$

with $\tilde{d}(k) = C(k)^{\dagger}y(k)$ same as in (4-15). Thus we aim to evaluate the ensemble variance of signal $\tilde{d}(k+1) - A\tilde{d}(k)$ and $y(k) - C(k)\tilde{d}(k)$. Before that, we must select for which time step, k is the ensemble variance being evaluated for. To evaluate the variance for $w(k)$, we evaluate $\tilde{d}(k+1) - A\tilde{d}(k)$ for all k and then evaluate the ensemble variance. For variance of $v(k)$, it should be clear through some reasoning that to obtain a good estimate, we should use all the time-steps k for which $m(k) = m_{max}$. For these particular k, we will have $N(k) = \boldsymbol{I}_{m_{max}}$ and $y(k)$ will be a vector with non-zero elements since $m(k) = m_{max}$.

To test the RTS smoother algorithm, we generate several dataset having different localization error distribution and different output dimension which is based on maximum linked pairs in any succeeding frame, m_{max}. The resulting cumulative sum of the estimated states in y-axis are shown in Figure 4-5. It is observed that the smoother algorithm produces poor estimate when the output dimension, m_{max} is low and localization error is large. Furthermore, we should note that the estimated states consist of relative drift values. Therefore, any error in the estimation is cumulative when evaluating absolute drift using these values. It is for this reason that some deviation should be expected in any estimation process as the time-step increases which is also evident in Figure 4-5.

4-3-4 Final Algorithm

In last few subsections, we developed the tagging, system identification and smoothing algorithm. By appending the image reconstruction algorithm similar to subsection 3-3-2 at the

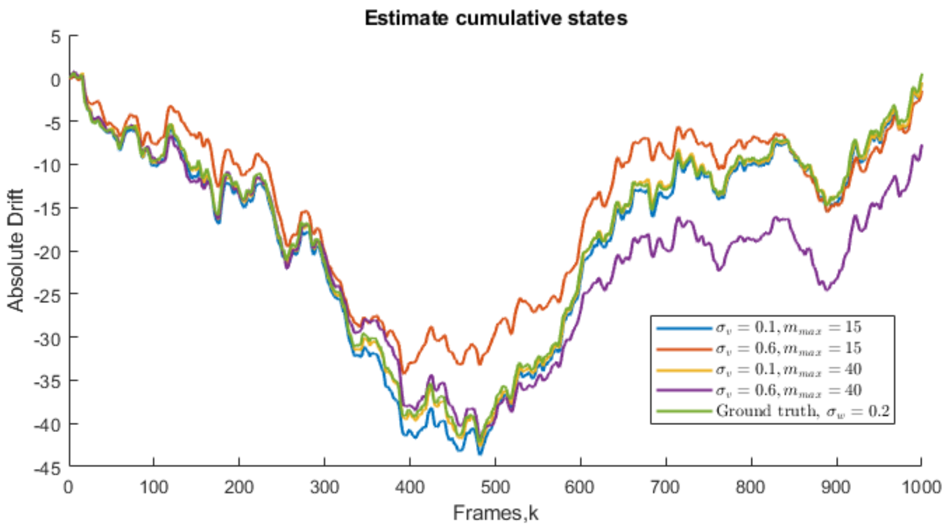

Figure 4-5: Cumulative sum of estimated state obtained using different output noise variance, σ_v and output dimension, m_{max}. As observed, the estimation is best with low localization error (output noise variance) and high output dimension or equivalently, high number of linking within data. The Smoother is more sensitive to the localization error overall.

end, we obtain the drift compensation algorithm using the model in (4-3) and (4-4).

```
Algorithm 1: RTS Smoother
  Result: $\hat{x}(k)$, $P(k)$, $K(k)$
  Initialize the forward filter;
      $\hat{x}_f(0) = E(x_0)$
      $P_f(0|0) = E\left[(x_0 - \hat{x}_f(0))(x_0 - \hat{x}_f(0))^T\right]$
      $k = 1$
  while $k \leq N$ do
      Execute Standard Kalman filter;
          $P_f(k|k-1) = A(k-1)P_f(k-1|k-1)A(k-1)^T + Q(k-1)$
          $K_f(k) = P_f(k-1|k)C(k)^T \left(R(k) + C(k)P(k|k-1)C(k)^\top\right)^{-1}$
          $\hat{x}_f(k|k-1) = A(k-1)\hat{x}_f(k-1|k-1)$
          $\hat{x}_f(k|k) = \hat{x}_f(k|k-1) + K_f(k)(y(k) - C(k)\hat{x}_f(k|k-1))$
          $P_f(k|k) = (I - K_f(k)C(k))P_f(k|k-1)$
          $k = k + 1$
  end
  Initialize RTS smoother;
      $\hat{x}(N) = \hat{x}_f(N|N)$
      $P(N) = P_f(N|N)$
      $j = N - 1$
  while $j \geq 1$ do
      Execute the RTS smoother;
          $\mathcal{I}_f(j+1|j) = (P_f(j+1|j))^{-1}$
          $K(j) = P_f(j|j)A(j)^T\mathcal{I}_f(j+1|j)$
          $P(j) = P_f(j|j) - K(j)(P_f(j+1|j) - P(j+1))K(j)^T$
          $\hat{x}(j) = \hat{x}_f(j|j) + K(j)(\hat{x}(j+1) - \hat{x}_f(j+1|j))$
          $j = j - 1$
  end
```

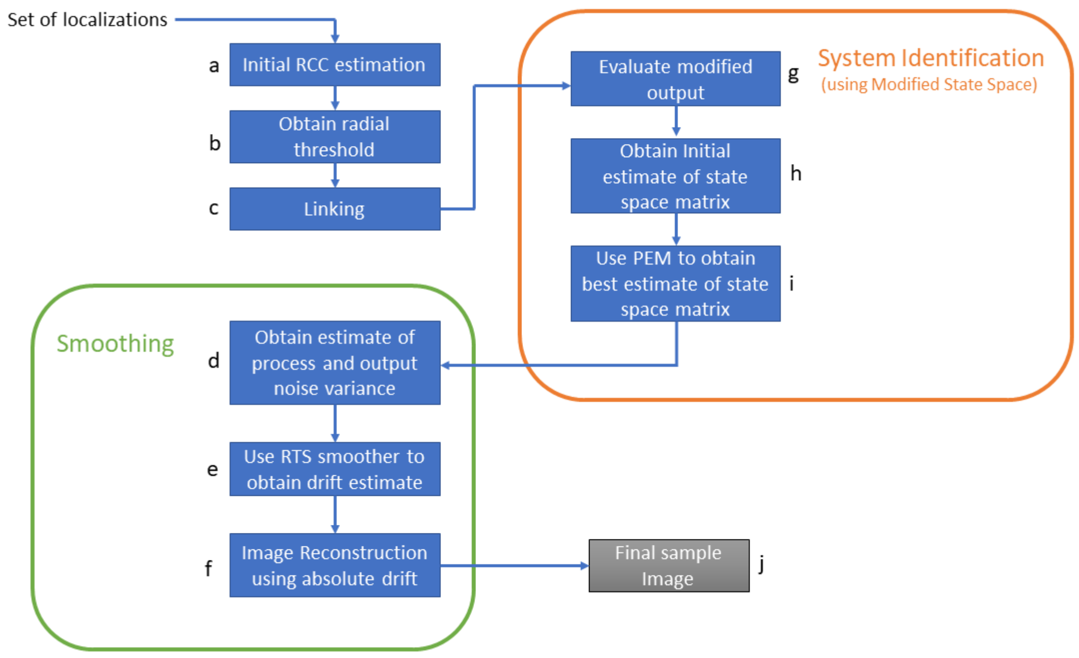
Set of localizations
a
Initial RCC estimation
b
Obtain radial threshold
c
Linking
System Identification
(using Modified State Space)
Evaluate modified output
g
Obtain Initial estimate of state space matrix
h
Use PEM to obtain best estimate of state space matrix
i
Smoothing
d
Obtain estimate of process and output noise variance
e
Use RTS smoother to obtain drift estimate
f
Image Reconstruction using absolute drift
Final sample Image
j

Figure 4-6: Schematic Representation of Drift compensation algorithm using the PLSS. To obtain a fair estimate of radial threshold for performing linking in (c), we obtain a rough estimate of drift using RCC, (a). The resulting drift estimate along with the localization error available from the localization process is used to pick a radial threshold value, (b). Using the output from linking, we obtain our observed data. For the purpose of system identification, we utilize the modified state space which involves evaluating a modified observed data in (g) and then obtaining an initial estimate of system dynamics matrix $\tilde{A}$ in (h) for the non-linear optimization problem formed using PEM to obtain the best estimate, $\hat{A}$ in (i). Using this we apply Kalman (a.k.a RTS) smoother on the original state space to obtain the best estimate of state, or relative drift. (d-e). Using this estimate, we re-construct the sample image in (f) and obtain the final image.

Chapter 5

Performance Comparison and Discussion

In the previous sections, we have defined two new drift estimation algorithms and tested them on simulated data. In this chapter, we compare these algorithms against current state-of-the-art alternatives, on both simulated and experimental data.

To be able to test these algorithms, we require a standard simulation pipeline using which localization data is generated. This would include numerous parameters such as to resemble the experimental data as closely as possible. This is discussed in section 5-1. Using this simulation pipeline we would obtain observations about the performance, computational time as well as flaws of the algorithms which are discussed in succeeding sections. These observations are concluded in Chapter 6 with summarized comments about the potential algorithm development.

5-1 Simulation Pipeline

Simulating the SMLM experiment requires generating binding sites or localization points, contaminating them with drift and then selecting a subset to be represented in each frame. Multiple parameters are required to be selected before simulating the SMLM data such as the variance of localization error, drift variance, the frame density etc. In this section, we describe each of these steps.

5-1-1 Localizations

The binding sites or localizations are supposed to be position of the blinking emitter probes. More information on localization process could be obtained in [2, 20, 16, 9]. How the localizations process works is not of concern here, but what is important is the output obtained. The output is generated using either a sample image or a mathematical function. In the case of using a sample image, we begin with importing a bitmap image of sample. This could be clusters, microtubules or any other shape. These images are used to extract the coordinates of the pixels. Unfortunately with this method, the problem of *pixelation* prevails. These pixels are then used to simulate the process of SMLM. In the case when a mathematical function is used, for e.g. in case of simulating microtubules, we may use spline to represent them and produce binding points along these splines. The points or localizations are used to simulate drift in SMLM.

Along with the process of localization, there is an error involved called *localization error*. This error is due to an inevitable inaccuracy involved whilst finding the exact position of emitter probe through the PSF generated. This error could be fairly accounted in the simulation by contaminating the position of each emitter by a unique ZMWN value. The amount of this localization error is controlled by varying the standard deviation (or equivalently, the variance) of this ZMWN. The standard deviation is measured in nm.

Relative drift, or frame to frame drift is generated using a normal distribution of particular mean and standard deviation (both in nm). The absolute drift is obtained by cumulative summation of the relative drift values. These values are used to contaminate the localization positions of the probes at each frame or time-step. From the set of contaminated localization positions, we select a subset which represents the data in single frame. This selection requires to account for random selection of emitter molecules as well as the *on-time* which is observed in STORM and DNA-PAINT dataset. The on-time refers to the phenomenon due to which the same emitter molecule is observed in succeeding frames for an arbitrary period. STORM data has very short on-times as compared to DNA-PAINT data which has much longer on-times.

5-1-2 Data Generation

The output frames from localizations are used as input to both algorithms. In case of SMSS algorithm, BaSDI and RCC, binning is also required. Binning refers to aggregation of certain number of frames into a single frame. With binning, we can virtually increase the frame density at the cost of losing information about drift. This is discussed thoroughly in subsection 3-4. BaSDI and SMSS requires integer position to work with. This requires slight modification in the frame data. We must first perform a scaling (affine transformation) or zooming and then round the values to closest integer. This produces a suitable integer approximation of frame data but also introduces discretization error. The simulation pipeline is shown in Figure 5-1.

For PLSS model, we also require to test the influence of linking. It is expected that there would be a threshold ratio of correct to incorrect linkings (or jaccard index) below which, considerable estimation error would be encountered. To find this threshold, we also create another additional data generation to specifically control the jaccard index. We use the resulting dataset to test the PLSS model.

5-2 Performance and Computational Time

Measurement of performance is based on RMSE between the ground truth of absolute drift trace used to contaminate the localizations and the estimated absolute drift obtained. The

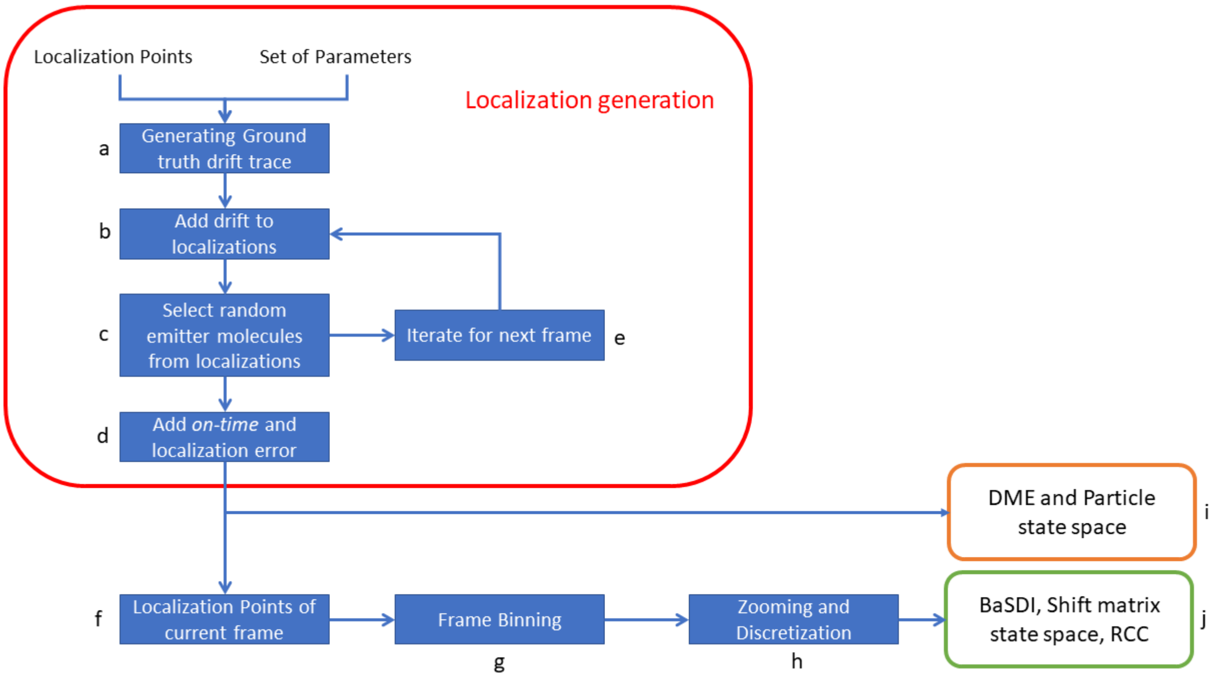
Localization Points
Set of Parameters
Localization generation
a
Generating Ground truth drift trace
b
Add drift to localizations
c
Select random emitter molecules from localizations
Iterate for next frame
e
d
Add *on-time* and localization error
DME and Particle state space
i
f
Localization Points of current frame
Frame Binning
g
Zooming and Discretization
h
BaSDI, Shift matrix state space, RCC
j

Figure 5-1: Schematic representation of Simulation for generating localization data. This consist of Generating the localization points using either image or a mathematical function along with a ground truth drift in (a) and adding these drift traces to subsequent frames in (b) along with introducing *on-time* and localization error in (d). The collective frame data in (f) is passed to perform binning in (g). In case of using BaSDI or shift state space for drift compensation, we also require to discretize the data which is done in (h)

computational time is measured from the point the algorithm begins to evaluate the drift estimate till the point the estimated drift is obtained. These measure could be made for different realizations of data, localization error, binning and other variables to obtain insight into the prowess and weakness of each algorithm.

To benchmark the state space derived drift compensation algorithms against the existing ones including BaSDI, DME and RCC, we generate data using the simulation pipeline discussed and record the performance along with computation time for different binning sizes. The ground truth relative drift is obtained from a normal distribution, $\mathcal{N}\left(\mu = 0nm, \sigma^2 = 4\text{nm}^2\right)$. Localization error is also generated using $\mathcal{N}\left(\mu = 0nm, \sigma^2 = 10\text{nm}^2\right)$. A total of 3467 binding points along 2D splines are used to generate data which consist of total 90482 binding sites. Each of $10,000$ frames have size 100×100px (100nm/px) and consist of 10 emitter per frame on average with an average on-time of 2 frames. Zooming is used to discretize the data for BaSDI and SMSS algorithm. We use a zoom factor of 15 with smoothing factor of 4 and window size 40px. For 2D SMSS, the drift values over which likelihood is evaluated is $\bar{d} = 2$. For 1D shift state space, this value is required to be changed suitably as per the binning size. The reason behind this is explained in next section. Since binning is not required in PLSS model, we vary the linking radius which in turn generates different observed data resulting in different estimate producing a variation in performance. Other parameters for BaSDI and DME algorithm are adjusted accordingly. The results are shown in Figure 5-2.

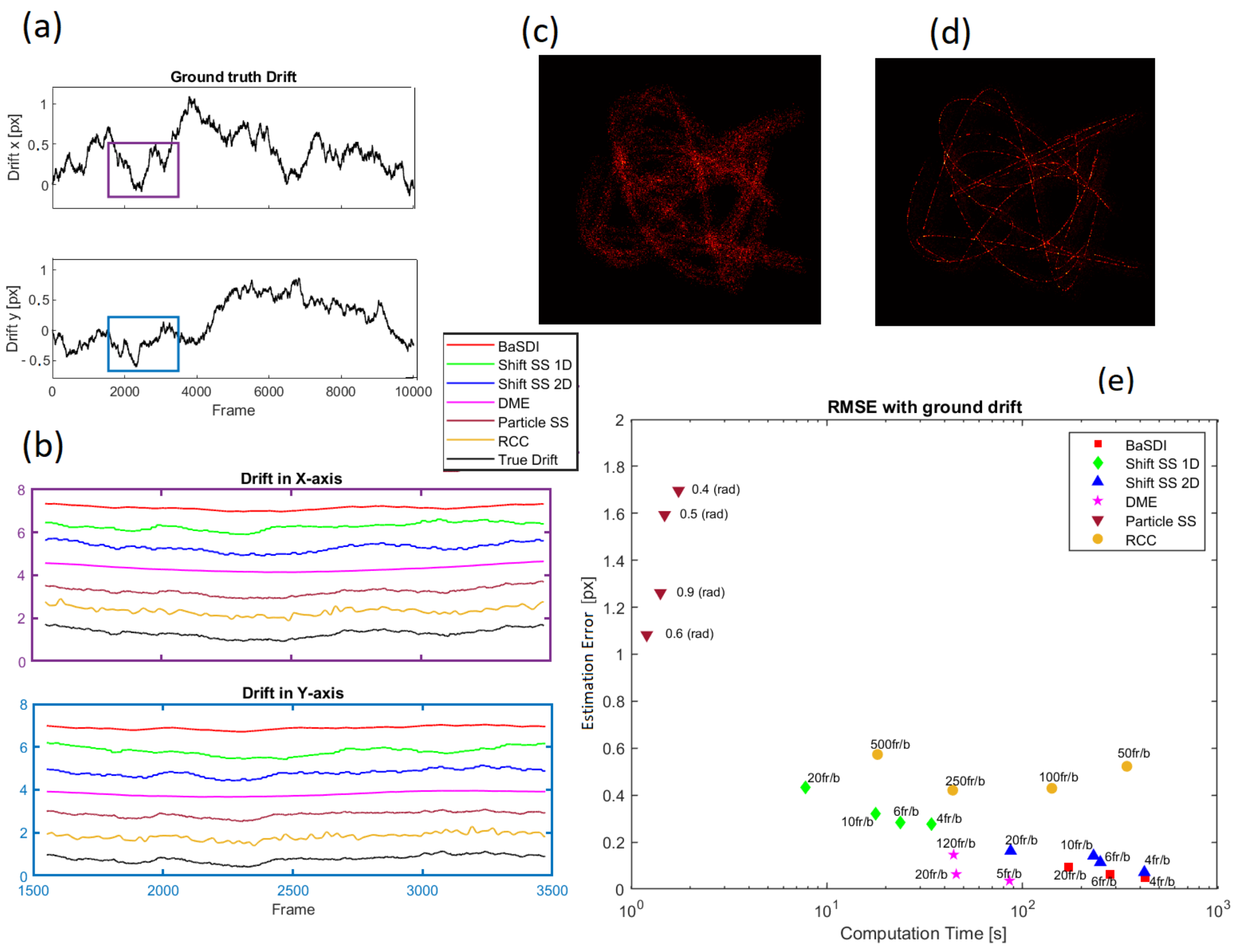

Figure 5-2: Different Passive Drift Compensation methods tested on Simulated Microtubule data. The Data consist of 10 microtubules with 100×100px frame (100nm/px) and $10,000$ frames in total. The frame density is 10m/f and an average on-time of 2 frames. The relative ground truth drift is generated using a normal distribution, $\mathcal{N}(\mu = 0\text{nm}, \sigma = 4\text{nm})$ as shown in (a). The localization error is generated using $\mathcal{N}(\mu = 0\text{nm}, \sigma = 10\text{nm})$. For discretization of data, a zoom factor of 15 was used. (b) shows enlarged section of ground truth along with the estimated drift using different algorithms. Drifted sample image is shown in (c) and (d) shows the undrifted sample image. (e) Shows the comparison graph of computational time against the RMSE precision. The performance of PLSS model is recorded by varying the linking radius, mentioned as 'rad' algorithm has very low computational time but produces large error. 1D Shift algorithm performs adequately whereas the 2D shift algorithm consumes slightly less computational time as compared to BaSDI but also produces higher error.

As shown in Figure 5-2(e), the 1D algorithm gives a fair estimate with smaller computational time as compared to BaSDI and DME. This result varies for more complex sample images and is discussed in the next section. The PLSS model algorithm takes the least computational time but the resulting estimates carry significant error in comparison to other methods. This is primarily due to the number of linked pairs and the cumulative error in relative drift estimates which is discussed further in section 5-4. The performance of 2D shift state space algorithm is slightly less than BaSDI algorithm and also takes lesser time. This is expected since

BaSDI algorithm takes into account both the estimation of absolute drift and the posterior distribution as compared to 2D state space algorithm which only takes into account the likelihood of relative drift. Furthermore, the problem with 2D shift algorithm is that it contains lot of parameters that require to be tuned properly to produce good results. Whereas BaSDI is a global optimizing algorithm, in the sense that it finds the optimum value of the parameters by default.

The results indicate that 1D shift state space and PLSS model have potential for drift compensation methods as they do not use excessive compute time. We investigate these two algorithms in following sections. First we discuss the limitation of 1D algorithm in the next section and then investigate the properties of PLSS model.

5-3 Discussion on 1D SMSS algorithm

It has been discussed in SMSS model that we can solve the problem of 2D drift correction as two separate cases of 1D drift estimation for each lateral axis. This is performed by summing the 2D dataset frames along lateral axis, normalizing it and using the resulting dataset for 1D shift state space model. Solving the 2D drift correction problem as 1D offers a high computational speed, it's performance decreases as the complexity of the sample image increases. How do we define a complex sample image?. The complexity of the image could be defined by how many clusters that sample image consists of and how large is the sample image. To show the increasing error, we form different simulated datasets with increasing number of microtubules and measure the difference between the resulting drift estimate from 1D drift compensation algorithm and that of 2D drift compensation algorithm. As is evident in Figure 5-3 (b), the estimation error in the drift estimate increases as the number of microtubule increase.

The error produced in the drift estimation is due to the loss of information when treating 2D dataset as two separate 1D dataset. We attempt to investigate this loss of information and quantify it between the 1D algorithm and 2D algorithm. A good candidate for such measure is the likelihood function. The likelihood function in (3-11) dictates the likelihood of all drift values in the set, $[-\bar{d}, \bar{d}]$ in both axis, i.e. d_x and d_y. We obtain a likelihood function for 2D drift correction, $\mathcal{L}(d_x, d_y)_{2D}$ and 1D drift correction, $\mathcal{L}(d_x)_{1D}$, $\mathcal{L}(d_y)_{1D}$. In ideal scenario, where no information is lost, the marginalized 2D likelihood in both axis should be equivalent to the 1D likelihood in respective axis.

$$\sum_{d_x} \mathcal{L}(d_x, d_y)_{2D} = \mathcal{L}(d_y)_{1D}$$

$$\sum_{y} \mathcal{L}(d_x, d_y)_{2D} = \mathcal{L}(d_x)_{1D}$$

But in practice, this is not the case since we would encounter some loss of information due to summing up of image along the axis. What we would like to measure is the difference between these two likelihoods.

$$\begin{aligned} e_x &= \sum_{d_x} \mathcal{L}(d_x, d_y)_{2D} - \mathcal{L}(d_y)_{1D} \\ e_y &= \sum_{d_y} \mathcal{L}(d_x, d_y)_{2D} - \mathcal{L}(d_x)_{1D} \end{aligned} \tag{5-1}$$

Unfortunately, one cannot evaluate this analytically. This is primarily because the summing up of image in either axis cannot be described as an analytical function, or in other words, there is no analytical description of the frames or sample image, they are just matrices containing values. Furthermore, it should also be noticed that the likelihood function is also dependent on the current frame, $\boldsymbol{y}(k)$ and also the current iteration i. So, if we aim to measure (5-1) numerically, we should decide suitably how to do so. For the iteration number, it is best to measure the likelihood when convergence is achieved since that gives the best resulting drift estimate. Then, we could compare the two likelihoods for all frames, $(\boldsymbol{y}(k)\ \ \forall k)$ and evaluate the mean.

First, we test the effect binning has on both the algorithms. We generate 10 different microtubules simulation data using the discussed pipeline. By varying the bin size and recording the performance, we are able to observe the effect binning has on 1D algorithm as compared to the 2D algorithm. Furthermore, we also vary the number of micro tubules (thereby the complexity of sample image) and record the RMSE along with the likelihood difference, e_x and e_y as in (5-1) for suitable binning size and drift values. The drift values of 1D algorithm are required to be changed with the binning size. This is because the 1D algorithm relies on marginalizing the actual 2D image, which gives rise to a more condensed image. This over-condensation causes wrong drift values to become more likely which is why it is more important to choose the right set of drift values in 1D algorithm where information is being compressed. Figure 5-3 (e),(f) show different number of microtubules consisting of 10 and 40 respectively. The likelihood error, e_x and e_y also increases as the image complexity increase (by increasing the number of micro-tubules). This makes the 1D algorithm more sensitive to varying parameters such as drift values, $\bar{d}$.

To quantify the performance of 1D and 2D algorithm on experimental data, we used DNA-PAINT origami nanostructure data. The non-bead localizations from this experiment were drift-corrected using BaSDI, 1D and 2D shift algorithm. The result is shown in Figure 5-4. The dataset consist of large number of clusters. Due to this, the sensitivity of 1D SMSS algorithm is very high, as evident by the fluctuations in the drift estimate. Fluctuations are also present in 2D SMSS algorithm.

5-4 Position Linking State Space model

The PLSS model algorithm takes much less time as compared to other algorithms but suffers from problem of cumulative error. The main reason behind this is the *linking* part of the algorithm. The linking determines the richness of the output data. If the data is not rich, neither the system identification algorithm nor the Kalman filter would provide apt results. The data is called rich when it contains a *high* number of *correctly* linked pairs. If either the number of linked pairs is small or the linked pairs are incorrect, i.e. the emitter molecules linked together are not the same molecule, then the resulting drift estimate is poor. We test both of these factors using different simulation data. For purpose of explaining the result, we called the ratio of correctly linked pairs to incorrectly linked pairs as jaccard index. The higher the jaccard index, the more precise are the results. This is shown in Figure 5-5(c) where the linking pairs are generated for given jaccard index values and their performance are measured. Repeated over different realization of simulated data with varying frame density, the results

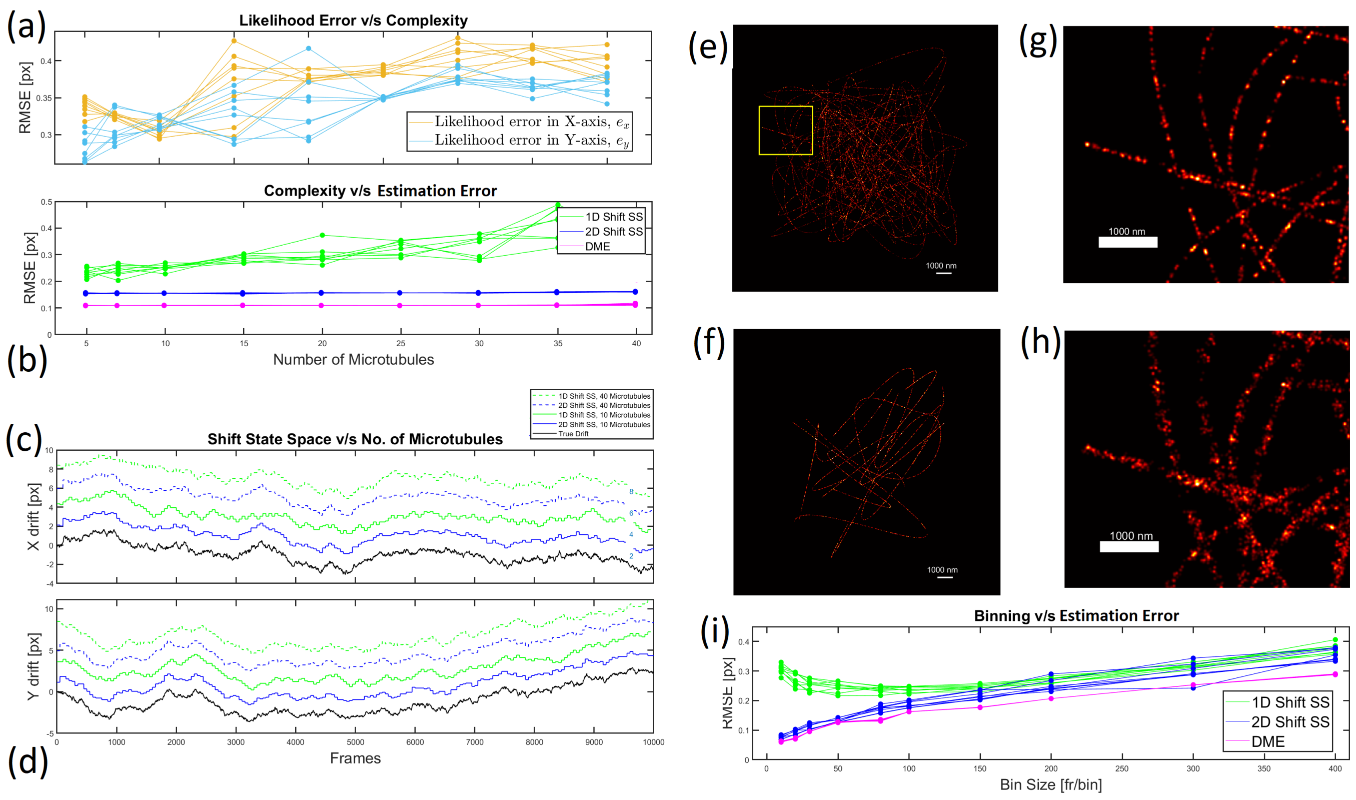

Figure 5-3: Quantitative measure of the difference between the likelihood and the RMSE in drift estimate. The data is created using different number of micro-tubules. The frames are of size 200×200 px with 100nm/px having localization error generated using normal distribution with standard deviation of 10nm. The relative drift is also generated using a normal distribution with standard deviation of 4nm. On average, the frames contain 10m/f and there are $10,000$ frames. The error measured in likelihood as discussed in (5-1) is shown in (a) along with measurement error with increasing number of microtubules. As evident, the likelihood error increases as the number of microtubules increase (i.e. the image complexity) due to which the resulting error in the drift estimate in 1D implementation increases as well. The drift estimate error in 2D algorithm remains almost constant. The error in DME is shown for reference. (e) and (f) shows sample images with 40 and 10 microtubules. (g) and (h) shows the drift corrected image using 1D shift and 2D shift algorithm respectively. (i) shows the effect of binning size on the performance of the algorithm.

confirm the claim. It is clear that as the jaccard index increases, the estimation error become smaller for all frame densities.

We also tested the algorithm with dataset which will give rise to higher number of linked pairs, correct and incorrect both. In principle, at low average on-time, a higher linking radius would result in low jaccard index which will give poor performance. This is shown in Figure 5-5(b) with red curves. Furthermore, low linking radius would give rise to no pairing, and thus empty output dataset which again produces poor performace. Good performance is obtained when we have large average on-time and high linking radius as shown in yellow and green curves in Figure 5-5(b).

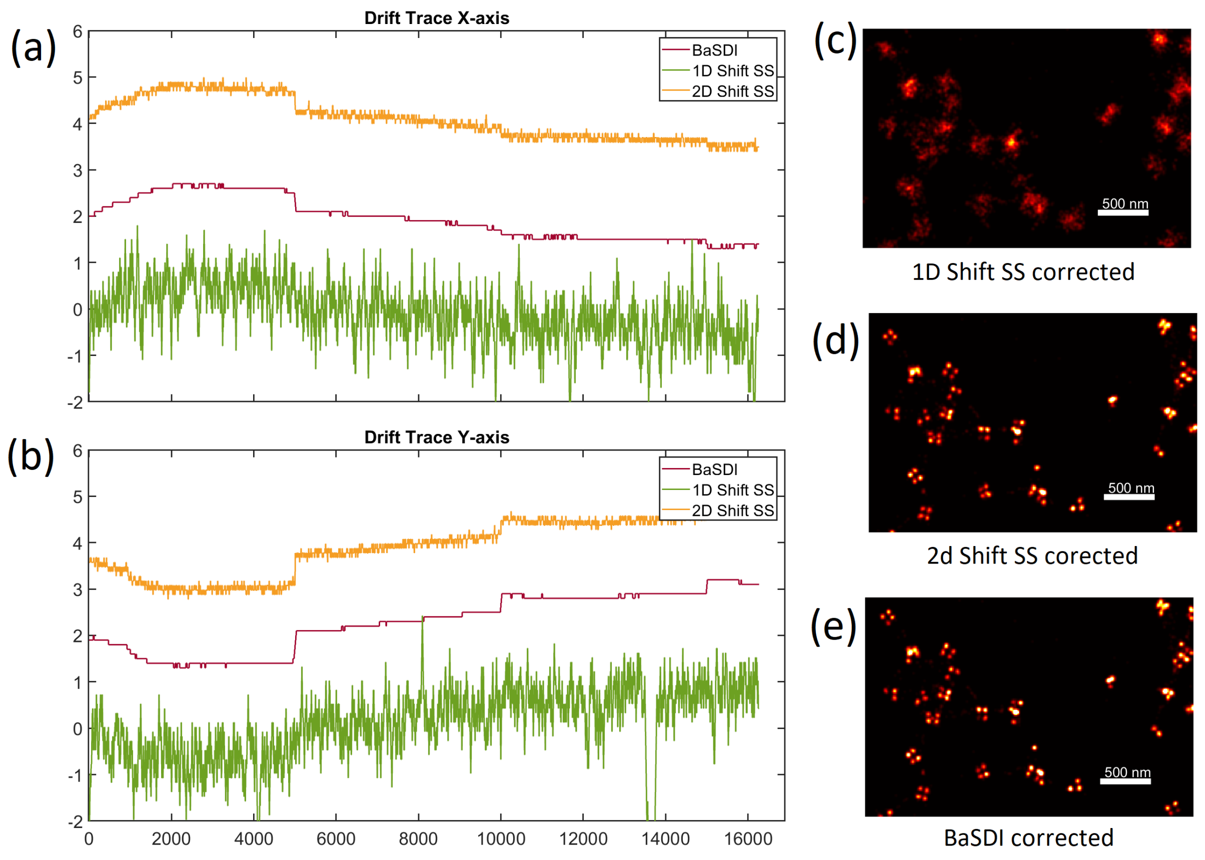

Figure 5-4: Estimated drift using DNA-PAINT nanostructure experimental data using BaSDI, 1D and 2D shift algorithm. (a) and (b) shows the estimated drift trace for X-axis and Y-axis respectively. Fluctuations are present in both 1D and 2D but it is more prominent in the former. (c), (d) and (e) shows the zoomed subsection of the sample image corrected using 1D, 2D and BaSDI.

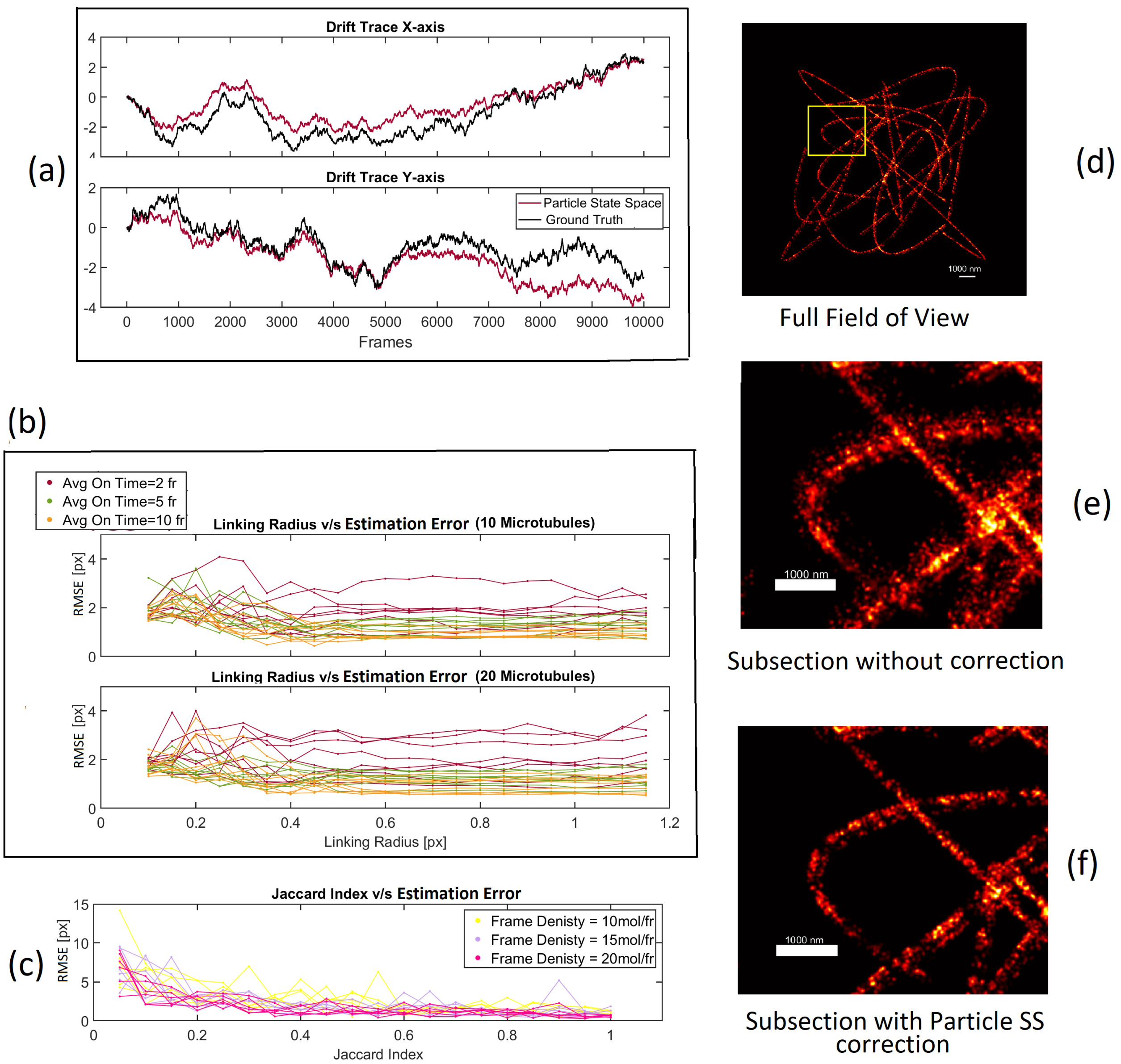

Figure 5-5: Estimated Drift using PLSS model. The simulated data contains 10 microtubule, $10,000$ frames with size 200×200px (100nm/px). The drift is generated using normal distribution $\mathcal{N}(\mu = 0\text{nm}, \sigma = 4\text{nm})$ and the localization error is generated using the normal distribution, $\mathcal{N}(\mu = 0\text{nm}, \sigma = 10\text{nm})$. The average on-time and frame density of molecules is varied to obtain insight into performance of algorithm. (a) shows the resulting drift trace with average on-time of 2 frames and 10m/f. It is evident that the trace diverges due to cumulative error in relative drift for both axis. The effect of linking radius is shown in (b) for different number of microtubules and different average on-times. Low linking radius leads to poor output data since less number of linked molecules are low. As the linking radius is increased or the average on-time is large, the number of pairs also increase which produce rich output data as well as low RMSE. To test how number of linked pairs affect the performance, we specifically generated correct pairs and in-correct pairs controller using Jaccard index and asses the RMSE with varying frame density. (d) Shows the sample image of simulated data. The drift contaminated section of the data is shown in (e) and the drift corrected data using PLSS model is shown in (f)

Chapter 6

Conclusion

6-1 Conclusion

The aim of this thesis was to investigate the application of state space methodologies for drift compensation in SMLM. Two main state space models have been derived in this thesis and used to perform drift correction on simulated and experimental dataset.

The first model utilizes the spatial molecular density as the state of the system. The dynamics of this state is influenced by drift which is introduced using shift matrices. These shift matrices shift the entries of matrix of spatial molecular density in lateral axis (i.e. along rows and columns). The output consist of a matrix containing number of emitters at each spatial coordinate. The number of emitters is dictated by the molecular density at that spatial coordinate, thus we define the output using a Poisson distribution. This is similar to many other applications of state space modelling where the output happens to be strictly integer [26]. The resulting state and output are matrices, and the resulting model is bi-linear. To be consistent with existing state space models, the bi-linear state space is converted to a linear state space model. Furthermore, we also derive the system identification algorithm for this model using maximum likelihood estimation to obtain the shift matrices and therefore the drift taking place in each frame. This estimated drift is used to reconstruct the state and obtain the underlying spatial molecular density. The resulting combination of system identification and state or image reconstruction gives the drift compensation algorithm using SMSS model. This drift compensation algorithm is further improved using tools such as gaussian image smoothing, EM algorithm and binning. We also attempted to treat 2D drift compensation problem as two independent cases of 1D drift compensation by pre-processing the 2D dataset to generate 1D datset using marginalization.

The second model relies on the position of the emitter molecules. Since using the position alone could give rise to number of practical problems such as very large dimensions, we introduced the process of linking, i.e. pairing molecules in succeeding frames. The molecules which are paired represent the same molecules but in succeeding frames. The data resulting from linking is used as output for the particle position state space model. Using Prediction

Error Methods (PEM), we perform system identification using data from linking and furthermore perform smoothing (stochastic signal processing) using RTS smoother to obtain the underlying state, which is relative drift. Since the model has time varying output noise, we also study the non-stationary characteristic of this noise signal and derive suitable condition under which the noise could be safely assumed as stationary.

The drift correction algorithms arising out of these two models are tested for their performance and computational time through bench-marking them with existing drift correction algorithms. We also study individual properties of these algorithms using different simulated and experimental dataset.

6-2 Recommendations

The algorithms developed in this thesis have drawbacks and advantages. These are quantified in the Chapter 5. Here we summarize the results along with suggested improvement in each algorithm. We also conclude with further research that could be performed using the models developed in this thesis study.

The 1D shift state space algorithm is fast and works well with simple sample structures. It achieves apt precision with 10 times less computational time as compared to BaSDI. But, it also fails when the sample image becomes complex. Furthermore, unlike the 2D state space algorithm or BaSDI, the bin size is required to be determined correctly in order to obtain the best possible estimate of drift. Also when working with 1D shift state space algorithm, larger drift value for likelihood, $\bar{d}$ produces more fluctuations as well. Thus it is also required to select a proper value of $\bar{d}$. The 2D shift state space algorithm works comparable to BaSDI algorithm and consumes slightly less time. But nevertheless, the computation time is large and thus it is not suitable for large dataset. Furthermore, at low frame density, the 2D algorithm would breakdown as well. BaSDI on the other hand does not. This is primarily because of BaSDI takes into account the absolute drift as well as prior information of the model of drift which gives better results than using likelihood alone.

One of the ways to improve the speed of 2D state space is to start with a better pre-estimation of drift similar to DME. This pre-estimation could be obtained using either 1D state space algorithm or RCC. There are several more test that could be performed on the algorithm for example investigating the effect of frame density also. *Frame density* is different from *Spatial Molecular Density*. The former is explicitly used to decide how many molecules appear in a single frame on average based on the dimensions of the captured image. Spatial molecular density is used in SMSS model and we limit it to that only. It is implicitly controls the number of molecules since it specifically dictates how many molecules will appear at a single spatial coordinate in a given frame. In a typical SMLM experiment, the value of frame density ranges from 4×10^{-4} to 1×10^{-4}. It is also important to note the difference between the frame density (m/fpx^2 for 2D and m/fpx for 1D, with px as pixels) and the number of molecules per frame (m/f). The relationship between two is given by (6-1).

$$\text{Molecular Density(2D)} = \frac{\text{Average Molecules per frame(m/f)}}{\text{Effect Sample Image Size(px}^2\text{)}} \tag{6-1}$$

The average number of molecules per frame could vary largely. It could be 20m/f, 30m/f or even larger but it is the molecular density which dictates the sparsity of data. Thus it could be insightful to experiment with frame density as well. Although we have measured the effect of different binning size on the SMSS algorithms, it could also be useful to understand the effect of average on-time and binning size on them.

It is also stated that the size of the gaussian smoothing window depends on the effective size of the sample image. This is manually adjusted for the tests performed in Chapter 5. Therefore, it is also possible to create a sub-algorithm which adjusts the window size (as well as the factor) for gaussian smoothing. This could involve finding the effect different window size (and factor) have on the performance of 2D algorithm. Furthermore, the performance of 2D state space algorithm could also be increased by decreasing the smoothing factor of gaussian window at each iteration by a certain amount.

For PLSS, we have used a linking radius to pair the molecules in succeeding frames. If there are multiple molecules lying within the radius, then the closest one is assigned as the pairing molecule. We can also use a more narrow condition to be able to obtain correct linking or pairing. For e.g. we can select a lower and upper radial bound within which we will investigate the pairing instead of searching for pair within specified radius. We can also try to correct the SMLM dataset using fast correction method, such as 1D shift state space algorithm or RCC and then apply the PLSS algorithm to obtain improved drift correction. PLSS also suffers from cumulative error in relative drift due to which the resulting estimate has increasing offset. It is worth noticing that the 2D shift state space algorithm also estimates relative drift but does not suffers from this offset. This is because for each frame, the estimated drift is first corrected and then the drift in next frame is estimated. We can use similar technique of updating information for updating output for single time step and then proceeding to estimate drift for the rest of the data.

Appendix A

Appendix

A-1 Implementation of shift state space

Although we have the dynamic recursion in (3-9), implementing this directly would be computationally expensive. To illustrate this, an SMLM image of 2000px$\times$2000px size would require shift matrices of size 2000×2000 and the resulting Kornecker product matrix, $\boldsymbol{A}_{d_x(k-1),d_y(k-1)}$ would be of size 4000000×4000000. Although the resulting matrix is sparse, multiplication with such large dimension could easily hamper the computational performance of the algorithm. Fortunately, the operation of multiplying with shift matrices could be easily implemented by the slicing operation, which is way faster than the matrix multiplication. We use the command `circshift` in MATLAB. This command shifts the state (spatial molecular density) matrix, $\boldsymbol{\theta}(k)$ along both rows(y) and columns(x) by desired number of times and direction. It also circulates the elements being shifted but due to padding, it has no effect on the implementation.

Furthermore, since the number of pixels having non-zero value is comparatively small, the image is stored in spatial cordinates having non-zero values as compared to matrix of the whole image. Whenever there is a requirement to obtain the complete image, `ij_to_image` function is used to obtain the image matrix from coordinates. Otherwise, when the size of image is too big, storing these in matrix form posses computational challenge from memory point of view. Evaluating the log likelihood in (3-12) is also computationally heavy. This is because of the number of times shifting, product and summation operations are performed in both terms on RHS (right hand side). Fortunately, the stated three operations could be condensed in a single operation performed by correlation function. Furthermore, the correlation function could be computed at much faster speed in Fourier domain using convolution. Although, to obtain equivalent result, we do require to pre-process the arguments.

1. Padding one of the argument by the number of maximum shifts with zeros. For e.g. in 2D case, we would padd the spatial molecular density matrix $\boldsymbol{\theta}(k)$ by $\bar{d}$ using the

function `padding` in MATLAB. In case of 1D, we would only require to padd in single axis.

2. Inverting the other argument, for e.g. the observed frame $\boldsymbol{y}(k)$ would be required to be rotated by 180° using the command `rot90` in MATLAB. For 1D, we could use `flip` instead.

We use the MATLAB function `conv2` to implement this operation. Therefore, (3-12) could be rewritten as the difference between two convolution terms.

$$\begin{aligned}\log\left(\mathcal{L}\left(\boldsymbol{\mathcal{Y}}(k)|\boldsymbol{\Theta}(k-1)', d_x(k-1), d_y(k-1)\right)\right) &= vec\left(\texttt{conv2}\left(\boldsymbol{\alpha},\boldsymbol{\beta}\right) - \texttt{conv2}\left(\boldsymbol{\alpha},\boldsymbol{I}\right)\right), \quad \text{where}\\ \boldsymbol{\alpha} &= \texttt{padding}\left(\log\left(\boldsymbol{\theta}(k)\right), [\bar{d},\bar{d}]\right)\\ \boldsymbol{\beta} &= \texttt{rot90}\left(\boldsymbol{y}(k), 2\right)\\ \boldsymbol{I} &= \texttt{eye}\left(n\right)\end{aligned} \tag{A-1}$$

As evident, the above operates over the whole data at once instead of iterating over each entry. This produces a significant increase in computational speed. In case of 1D, instead of using `conv2`, we simply use `conv` command.

To be able to find the maximizing argument of the optimization problem in 2D, one has to find the drift value corresponding to the maximum value of the resulting output from (A-1). To do so, we must first reshape the output into a matrix of shape $2\bar{d} \times 2\bar{d}$. Then, the index corresponding to this maximum value would yield the most likely value of drift. This is an example of combinatorial programming.

Readers might notice that we jump from using the terms from bi-linear model to the ones in linear model. That is because, from an analytical point of view involving derivation of likelihood algorithm, it is easier to deal with linear model whereas from a practical point of view where we require fast speed and have lot of 2D pre-build functions at hand, it is easier to deal with bi-linear model. Furthermore, as stated previously, the linear model also allows us to derive the likelihood algorithm for both 1D and 2D using almost no difference.

A-2 EM Algorithm

Consider a domain and range sample set $\mathcal{H}$ and $\mathcal{Y}$. Let us also assume a many to one mapping (similar to an invalid function) from $\mathcal{H}$ to $\mathcal{Y}$. Now the article defines an **incomplete set $\boldsymbol{x}$** which is not directly observed, but indirectly observed through $\boldsymbol{y}$. The article also defines sample densities $f(\boldsymbol{x}|\boldsymbol{\Phi})$ and $g(\boldsymbol{y}|\boldsymbol{\Phi})$ parameterized by the vector Φ which defines the shape of density in the sample sets. We assume there is a mapping $\boldsymbol{x} \to \boldsymbol{y}(\boldsymbol{x})$ from $\mathcal{H}$ to $\mathcal{Y}$, and that $\boldsymbol{x}$ is known only to lie in $\mathcal{H}(\boldsymbol{y})$, the subset of $\mathcal{H}$ determined by the equation y = y(x), where y is the observed data.

For a simple numerical example, consider three different bags containing red, green and blue balls $y = \{10, 15, 13\}$ respectively, with a total 38 balls. The corresponding distribution would be $P(y) = \{\frac{10}{38}, \frac{15}{38}, \frac{13}{38}\}$. The different ways in which we may draw 10 red, 15 green and 13 blue balls and putting them back instantly, would be something like $\underline{\text{R}}$,$\underline{\text{R}}$,$\underline{\text{R}}$,$\underline{\text{R}}$... or $\underline{\text{R}}$,$\underline{\text{B}}$,$\underline{\text{R}}$,$\underline{\text{R}}$... and so on and the total number of ways would be $\frac{38!}{10!\times 15!\times 13!}$. The corresponding probability for any sequence of draw for e.g. $\underline{\text{R}}$,$\underline{\text{R}}$,$\underline{\text{R}}$,$\underline{\text{R}}$... would be $(\frac{10}{38})^{10}(\frac{15}{38})^{15}(\frac{13}{38})^{13}$. Therefore, the total probability of drawing would be $\frac{38!}{10!\times 15!\times 13!} \times (\frac{10}{38})^{10}(\frac{15}{38})^{15}(\frac{13}{38})^{13}$. Compare this with the equation (**??**). There θ_{ij} gives the probability/molecular density and f_{ij} gives the number of ways in which each molecule is sort of drawn out.

Consider another example in which 197 animals are distributed multinomially into four categories, so that the observed data consist of

$$Y = (y3, y2, y3y4) = (125, 18, 20, 34)$$

Thus

$$g(\mathbf{y} \mid \pi) = \frac{(y_1 + y_2 + y_3 + y_4)!}{y_1!y_2!y_3!y_4!} \left(\frac{1}{2} + \frac{1}{4}\pi\right)^{y_1} \left(\frac{1}{4} - \frac{1}{4}\pi\right)^{y_3} \left(\frac{1}{4} - \frac{1}{4}\pi\right)^{y_2} \left(\frac{1}{4}\pi\right)^{y}.$$

To illustrate the EM algorithm, we represent $\mathbf{y}$ as incomplete data from a five-category multinomial population where the cell probabilities are $\left(\frac{1}{2}, \frac{1}{4}\pi, \frac{1}{4}(1-\pi), \frac{1}{4}(1-\pi), \frac{1}{4}\pi\right)$, the idea being to split the first of the original four categories into two categories. Thus the complete data consist of $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5)$ where $y_1 = x_1 + x_2, y_2 = x_3, y_3 = x_4, y_4 = x_5$, and the complete data specification is

$$f(\mathbf{x} \mid \pi) = \frac{(x_1 + x_2 + x_3 + x_4 + x_5)!}{x_1!x_2!x_3!x_4!x_5!} \left(\frac{1}{2}\right)^{x_1} \left(\frac{1}{4}\pi\right)^{x_2} \left(\frac{1}{4} - \frac{1}{4}\pi\right)^{x_2} \left(\frac{1}{4} - \frac{1}{4}\pi\right)^{x_4} \left(\frac{1}{4}\pi\right)^{x_5}$$

Note that the integral in (1.1) consists in this case of summing (1.3) over the (x_1, x_2) pairs $(0, 125), (1, 124), \ldots, (125, 0)$, while simply substituting (18,20,34) for (x_3, x_4, x_5)

To define the EM algorithm we show how to find $\pi^{(p+1)}$ from $\pi^{(p)}$, where $\pi^{(p)}$ denotes the value of π after p iterations, for $p = 0, 1, 2, \ldots$. As stated above, two steps are required. The expectation step estimates the sufficient statistics of the complete data $\mathbf{x}$, given the observed data $\mathbf{y}$. In our case, (x_3, x_4, x_5) are known to be (18,20,34) so that the only sufficient statistics that have to be estimated are x_1 and x_2 where $x_1 + x_2 = y_1 = 125$. Estimating x_1 and x_2 using the current estimate of π leads to

$$x_1^{(p)} = 125\frac{\frac{1}{2}}{\frac{1}{2} + \frac{1}{4}\pi^{(p)}} \text{ and } x_2^{(p)} = 125\frac{\frac{1}{4}\pi^{(p)}}{\frac{1}{2} + \frac{1}{4}\pi^{(p)}}$$

The maximization step then takes the estimated complete data $\left(x_1^{(p)}, x_2^{(p)}, 18, 20, 34\right)$ and estimates π by maximum likelihood which involves finding $\frac{\partial f}{\partial \pi}$. It should be noticed that $x_1^{(p)}$ and $x_2^{(p)}$ have no relation with π, thus the partial derivative needs not to perform the derivative action on $x_1^{(p)}$ and $x_2^{(p)}$.

$$\pi^{(p+1)} = \frac{x_2^{(p)} + 34}{x_2^{(p)} + 34 + 18 + 20}$$

Bibliography

[1] Alexander Balinovic, David Albrecht, and Ulrike Endesfelder Endesfelder, David Albrecht. Spectrally red-shifted fluorescent fiducial markers for optimal drift correction in localization microscopy. *Journal of Physics D: Applied Physics*, Mar 2019.

[2] Eric Betzig. Imaging intracellular fluorescent proteins at nanometer resolution. *Science*, 313(5793):1642–1645, 2006.

[3] Ting-Li Chen, Dai-Ni Hsieh, Hung Hung, I-Ping Tu, Pei-Shien Wu, Yi-Ming Wu, Wei-Hau Chang, and Su-Yun Huang. γ-sup: A clustering algorithm for cryo-electron microscopy images of asymmetric particles. *The Annals of Applied Statistics*, 8(1), Mar 2014.

[4] Jelmer Cnossen, Tao Ju Cui, Chirlmin Joo, and Carlas Smith. Drift correction in localization microscopy using entropy minimization. *bioRxiv*, 2021.

[5] W. Colomb, J. Czerski, J. D. Sau, and S. K. Sarkar. Estimation of microscope drift using fluorescent nanodiamonds as fiducial markers. *J Microsc*, 266(3):298–306, 06 2017.

[6] Ecole Polytechnique Fédérale de Lausanne (EPFL). Collection of reference datasets.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[8] A. Elmokadem and J. Yu. Optimal Drift Correction for Superresolution Localization Microscopy with Bayesian Inference. *Biophys J*, 109(9):1772–1780, Nov 2015.

[9] Klaus C Gwosch Arvid H Gynnå Volker Westphal Fernando D Stefani Johan Elf Francisco Balzarotti, Yvan Eilers and Stefan W Hell. Nanometer resolution imaging and tracking of fluorescent molecules with minimal photon fluxes. *Science*, 355(6325):606–612, 2017.

[10] C. Geisler, T. Hotz, A. Schönle, S. W. Hell, A. Munk, and A. Egner. Drift estimation for single marker switching based imaging schemes. *Opt Express*, 20(7):7274–7289, Mar 2012.

[11] Klaus C. Gwosch and Jasmin Pape. Minflux nanoscopy delivers 3d multicolor nanometer resolution in cells. *Nature*, 2022.

[12] Monson H. Hayes. *Statistical Digital Signal Processing and Modeling.* Wiley, 1996.

[13] T. Heil and H. Kohl. Optimization of EFTEM image acquisition by using elastically filtered images for drift correction. *Ultramicroscopy*, 110(7):748–753, Jun 2010.

[14] S. T. Hess, T. P. Girirajan, and M. D. Mason. Ultra-high resolution imaging by fluorescence photoactivation localization microscopy. *Biophys J*, 91(11):4258–4272, Dec 2006.

[15] Robert Resnick Jearl Walker, David Halliday. *Fundamentals of physics.* Wiley, 2014.

[16] Francisco Balzarotti Philipp Hoess Jan Ellenberg Jonas Ries Klaus C Gwosch, Jasmin K Pape and Stefan W Hell. Minflux nanoscopy delivers multicolor nanometer 3dresolution in (living) cells. *bioRxiv*, page 734251, 2019.

[17] S. H. Lee, M. Baday, M. Tjioe, P. D. Simonson, R. Zhang, E. Cai, and P. R. Selvin. Using fixed fiduciary markers for stage drift correction. *Opt Express*, 20(11):12177–12183, May 2012.

[18] H. Ma, J. Xu, J. Jin, Y. Huang, and Y. Liu. A Simple Marker-Assisted 3D Nanometer Drift Correction Method for Superresolution Microscopy. *Biophys J*, 112(10):2196–2208, May 2017.

[19] R. McGorty, D. Kamiyama, and B. Huang. Active Microscope Stabilization in Three Dimensions Using Image Correlation. *Opt Nanoscopy*, 2(1), Apr 2013.

[20] Xiaowei Zhuang Michael J Rust, Mark Bates. Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (storm). *Nature Methods*, (3):793–796, 2006.

[21] M. J. Mlodzianoski, J. M. Schreiner, S. P. Callahan, K. Smolková, A. Dlasková, J. Santorová, P. Ježek, and J. Bewersdorf. Sample drift correction in 3D fluorescence photoactivation localization microscopy. *Opt Express*, 19(16):15009–15019, Aug 2011.

[22] R. Bechsteinc P. Rahea and A. Kühnlea. Vertical and lateral drift corrections of scanning probe microscopy images. *Journal of Vacuum Science Technology*, 2010.

[23] A. Pertsinidis, Y. Zhang, and S. Chu. Subnanometre , registration and distance measurements. *Nature*, 466(7306):647–651, Jul 2010.

[24] G. Shtengel, J. A. Galbraith, C. G. Galbraith, J. Lippincott-Schwartz, J. M. Gillette, S. Manley, R. Sougrat, C. M. Waterman, P. Kanchanawong, M. W. Davidson, R. D. Fetter, and H. F. Hess. Interferometric fluorescent super-resolution microscopy resolves 3D cellular ultrastructure. *Proc Natl Acad Sci U S A*, 106(9):3125–3130, Mar 2009.

[25] Dan Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches.* Wiley-Interscience, USA, 2006.

[26] Jan H van Schuppen. *Control and System Theory of Discrete-Time Stochastic Systems.* Springer, 2021.

[27] C. von Middendorff, A. Egner, C. Geisler, S. W. Hell, and A. Schönle. Isotropic 3D Nanoscopy based on single emitter switching. *Opt Express*, 16(25):20774–20788, Dec 2008.

[28] Y. Wang, E. Cai, J. Sheung, S. H. Lee, K. W. Teng, and P. R. Selvin. Fluorescence imaging with one-nanometer accuracy (FIONA). *J Vis Exp*, (91):51774, Sep 2014.

[29] Y. Wang, J. Schnitzbauer, Z. Hu, X. Li, Y. Cheng, Z. L. Huang, and B. Huang. Localization events-based sample drift correction for localization microscopy with redundant cross-correlation algorithm. *Opt Express*, 22(13):15982–15991, Jun 2014.

[30] L. Xiao and E. S. Yeung. Optical imaging of individual plasmonic nanoparticles in biological samples. *Annu Rev Anal Chem (Palo Alto Calif)*, 7:89–111, 2014.

Glossary

List of Acronyms

SMLM	Single Molecule Localization Microscopy
SRM	Super Resolution Microscopy
EM	Expectation-Minimization
PLSS	Position Linking State Space
PSD	Power Spectral Density
MAP	Maximum a priori
SISO	Single Input Single Output
PEM	Prediction Error Methods
RMSE	Root Mean Squared Error
HMM	Hidden Markov Model
RMS	Root Mean Squared
LLSQ	Linear Least Square
QPD	Quadrant Photodiode
ROI	Region of Interest
SNR	Signal to Noise Ratio
ZMWN	Zero Mean White Noise
DCC	Direct Cross Correlation
MCC	Mean Cross Correlation
RCC	Redundant cross-correlation
CCD	Charge-Coupled device
IR	Infrared
DME	Drift at Minimum Entropy
BaSDI	Bayesian Sample Drift Inference

STED	Structures Illumination Microscopy
STORM	Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy
PALM	Photoactivated Localization Microscopy
FIONA	Fluorescence Imaging with One-nanometer Accuracy
PSF	Point Source Function
RTS	Rauch–Tung–Striebe
SMSS	Shift Matrix State Space