2022

Particle detection, extraction, and state estimation in single particle tracking microscopy

BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Dissertation

PARTICLE DETECTION, EXTRACTION, AND STATE ESTIMATION IN SINGLE PARTICLE TRACKING MICROSCOPY

by

YE LIN

B.S., Beihang University, 2015
M.S., Beihang University, 2017

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2022

Approved by

First Reader

Sean B. Andersson, PhD
Professor of Systems Engineering
Professor of Mechanical Engineering

Second Reader

John Baillieul, PhD
Distinguished Professor of Systems Engineering
Distinguished Professor of Mechanical Engineering
Distinguished Professor of Electrical and Computer Engineering

Third Reader

David Castañón, PhD
Professor of Systems Engineering
Professor of Electrical and Computer Engineering

Fourth Reader

Prakash Ishwar, PhD
Professor of Systems Engineering
Professor of Computer Science
Professor of Electrical and Computer Engineering

Try to become a person of value.

Albert Einstein

Acknowledgments

First of all, I would like to appreciate the tremendous help provided by my advisor Prof. Sean B. Andersson. Without great patience and rigorous guidance from my advisor, I will never have arrived where I am. I'm always grateful for the time and efforts offered by my advisor cultivating me to be a better learner, researcher, presenter, and engineer.

This is a wonderful research journey to me at Boston University, and there are a lot of faculties and staff that I would like to say thank you. Thank you, my committee members: Prof. John Baillieul who teaches me to be a person with a clear goal before taking actions, Prof. Castañón who teaches me to be open to any challenging problem rather than being afraid of it, Prof. Prakash Ishwar who teaches me to hold a growth mindset (as an old Chinese saying goes "one may have changed for the better after an absence of three days"). Thank you, Prof. Christos Cassandras who teaches me not to be rush but calm down when situation is not under control. Thank you, Prof. Yannis Paschalidis and Prof. Michael Caramanis who teach me never be held back by failures. Thank you, Prof. Roberto Tron who serves as the Chair of my PhD defense. Thank you, Prof. Hua Wang, Elizebeth Flagg, and Christine Ritzkowski who offer supports for my progress in both academia and industry. Thank you, Christina Polyzos and Maureen Stanton who teach me to be a thoughtful people for others. Meanwhile, I would like to express my gratitude to my previous advisors and teachers, Prof. Xing Pan, Prof. Liang Ma, Mr. Baiqiang Qi, and Mr. Song Gao, who offered me encouragement and support on my road to Boston University.

Whatever you do in this life, it's not legendary, unless your family and friends are there to see it. Thank you, my closed old friends: Chuan Luo, Chenyu Zhang, and An Zhou who lived far away from me but always offer me warmth and strength. Thank you, my labmates and friends: Dr. Boris I. Godoy, Dr. Samuel Pinto, Shelia

Kang, Sean Sanchez, Zili Wang, Yuhe Chang, Nic Vickers, Yancheng Zhu, Dr. Xi Yu, Dr. Yufan Luo, Dr. Trevor Ashley, Fatemeh Sharifi, for providing help and support for me. Thank you, my PhD colleagues and friends: Dr. Wei Xiao, Mahroo Bahreinian, Yuping Wang, Kasra Ghasemi, Dr. Majid Heidarifar, Ziqi Yang, Rui Liu, Dr. Shirantka Welikala, Dr. Salomon Wollenstein Betech, Waleed Aslam, Qianqian Ma, Mandy Liu, Dr. Xiao Wang, Dr. Tingting Xu, Dr. Xinmiao Sun, Dr. Ruidi Chen, Dr. Nan Zhou, Dr. Arian Houshmand, Dr. Taiyao Wang, Feiyang Kang, Dr. Bee Van, Guanyang Xue, Dr. Yanhe Huang, Dr. Yuyao Wang, Yu Xuan, for adding funs to my PhD life.

Last but not least, I would like to express my gratitude to my beloved family members: both of my paternal and maternal grandmas, my parents, my brother, and my husband Dr. Qi Feng. Their unconditional love and support shed light on my way no matter where I am.

PARTICLE DETECTION, EXTRACTION, AND STATE ESTIMATION IN SINGLE PARTICLE TRACKING MICROSCOPY

YE LIN

Boston University, College of Engineering, 2022

Major Professor: Sean B. Andersson, PhD
Professor of Systems Engineering
Professor of Mechanical Engineering

ABSTRACT

Single Particle Tracking (SPT) plays an important role in the study of physical and dynamic properties of biomolecules moving in their native environment. To date, many algorithms have been developed for localization and parameter estimation in SPT. Though the performance of these methods is good when the signal level is high and the motion model simple, they begin to fail as the signal level decreases or model complexity increases. In addition, the inputs to the SPT algorithms are sequences of images that are cropped from a large data set and that focus on a single particle. This motivates us to seek machine learning tools to deal with that initial step of extracting data from larger images containing multiple particles. This thesis makes contributions to both data extraction question and to the problem of state and parameter estimation.

First, we build upon the Expectation Maximization (EM) algorithm to create a generic framework for joint localization refinement and parameter estimation in SPT. Under the EM-based scheme, two representative methods are considered for generat-

ing the filtered and smoothed distributions needed by EM: Sequential Monte Carlo - Expectation Maximization (SMC-EM), and Unscented - Expectation Maximization (U-EM). The selection of filtering and smoothing algorithms is very flexible so long as they provide the necessary distributions for EM. The versatility and reliability of EM based framework have been validated via data-intensive modeling and simulation where we considered a variety of influential factors, such as a wide range of signal-to-background ratios, diffusion speeds, motion blur, camera types, image length, etc.

Meanwhile, under the EM-based scheme, we make an effort to improve the overall computational efficiency by simplifying the mathematical expression of models, replacing filtering/smoothing algorithms with more efficient ones (trading some accuracy for reduced computation time), and using parallel computation and other computing techniques. In terms of localization refinement and parameter estimation in SPT, we also conduct an overall quantitative comparison among EM based methods and standard two-step methods. Regarding the U-EM, we conduct transformation methods to make it adapted to the nonlinearities and complexities of measurement model. We also extended the application of U-EM to more complicated SPT scenarios, including time-varying parameters and additional observation models that are relevant to the biophysical setting.

The second area of contribution is in the particle detection and extraction problem to create data to feed into the EM-based approaches. Here we build Particle Identification Networks (PINs) covering three different network architectures. The first, $\mathrm{PIN}_{\mathrm{CNN}}$, is based on a standard Convolutional Neural Network (CNN) structure that has previously been successfully applied in particle detection and localization. The second, $\mathrm{PIN}_{\mathrm{ResNet}}$, uses a Residual Neural Network (ResNet) architecture that is significantly deeper than the CNN while the third, $\mathrm{PIN}_{\mathrm{FPN}}$, is based on a more advanced Feature Pyramid Network (FPN) that can take advantage of multi-scale

information in an image. All networks are trained using the same collection of simulated data created with a range of SBRs and fluorescence emitter densities, as well as with three different Point Spread Functions (PSFs): a standard Born-Wolf model, a model for astigmatic imaging to allow localization in three dimensions, and a model of the Double-Helix engineered PSF. All PINs are evaluated and compared through data-intensive simulation and experiments under a variety of settings.

In the final contribution, we link all above together to create an algorithm that takes in raw camera data and produces trajectories and parameter estimates for multiple particles in an image sequence.

Contents

List of Tables

List of Figures

List of Abbreviations

AP	Average Precision
BSPS	Backward Simulation Particle Smoother
BW	Born-Wolf
CCD	Charge-Coupled Device
CNN	Convolutional Neural Network
DH	Double-Helix
EM	Expectation Maximization
EMCCD	Electron-Multiplying Charge-Coupled Device
FFBS	Forward Filtering Backward Smoothing
FPN	Feature Pyramid Network
GF	Gaussian Fitting
GPF	Gaussian Particle Filter
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MSD	Mean Squared Displacement
NA	Numerical Aperture
OU	Ornstein-Uhlenbeck
PF	Particle Filter
PIN	Particle Identification Network
PS	Particle Smoother
PSF	Point Spread Function
ResNet	Residual Neural Network
ROC	Receiver Operating Characteristic
SBR	Signal-to-background Ratio
sCMOS	Scientific Complimentary Metal-Oxide Semiconductor
sGPF	Simplified Gaussian Particle Filter
SIR	Sampling Importance Resampling
SMC	Sequential Monte Carlo
SNR	Signal-to-noise Ratio
SPT	Single Particle Tracking
UKF	Unscented Kalman Filter
URTSS	Unscented Rauch-Tung-Striebel Smoother

Chapter 1

Introduction

1.1 Overview of Single Particle Tracking

Single particle tracking (SPT) is an important class of techniques for studying the motion of nanometer-scale biomolecules moving in their native environment. With the ability to localize particles with an accuracy far below the diffraction limit of light and to track particles across time, SPT continues to be an invaluable tool in understanding biology at the nanometer-scale by revealing details about particle dynamics and their local environment such as diffusion rates, confinement length, and other parameters (Shen et al., 2017). SPT has been applied to a wide variety of molecules, including proteins (Simson et al., 1995; Holcman et al., 2018), mRNA molecules (Park et al., 2010), DNA (Rösch et al., 2018), viruses (Ewers et al., 2005; Peerboom et al., 2018), growth factor receptor (Clarke and Martin-Fernandez, 2019), Janus colloids (Kurzthaler et al., 2018), and more (Zhong and Wang, 2020).

Typically, SPT analysis of a sequence of images begins with an image segmentation step where the raw images are post-processed to extract image sequences that each contain information about a single particle of interest. These sequences are then further processed to determine particle trajectories and motion model parameters. Under the standard paradigm, a two-step process is applied to each image sequence. In the first step, the location of the particle in each segmented image frame is determined and linked across frames to form a trajectory (Chenouard et al., 2014) (we refer to this as "localization refinement" since the initial segmentation is a coarse lo-

calization step). In the second step, trajectories are analyzed to extract information about the dynamic process, such as the value of the diffusion coefficient or other motion parameters. Localization refinement is often done using Gaussian Fitting (GF) (Thompson et al., 2002; Anthony and Granick, 2009) while model parameters are extracted from the trajectories using the Mean Squared Displacement (MSD) (Saxton and Jacobson, 1997; Michalet, 2010) or Maximum Likelihood Estimation (MLE) (Berglund, 2010; Calderon, 2016). Regardless of the algorithms used, this two-step paradigm separates trajectory estimation from model parameter identification despite the fact that these two problems are coupled.

One of the assumptions of the standard approach is that the localized positions represent a simple linear observation of the true particle position corrupted by additive white Gaussian noise. The actual data, however, are usually the segmented camera images. The photon detection process in each pixel during imaging can be well modeled as a Poisson-distributed random variable with a rate that depends on the true location of the particle as well as on experimental realities, including background intensity noise and the details of the optics used in the instrument. This already nonlinear model becomes even more complicated at the low signal intensities that are often found in SPT data.

To handle nonlinear measurement models and to jointly perform localization and parameter estimation, one can take advantage of nonlinear system identification and optimal estimation as in (Ashley and Andersson, 2015). Further, since the analysis is off-line, one can use non-causal estimation methods. This general approach, known as *Sequential Monte Carlo-Expectation Maximization* (SMC-EM), can handle nearly arbitrary nonlinearities in both the motion and observation models and has been shown to work as well as current state-of-the-art methods in the simple settings of 2-D diffusion and to work under more complicated motion and observation scenarios

including estimating 3-D motion from wide field images.

Overview of the thesis

SMC-EM applies a basic Sequential Importance Resampling (SIR) combined with a Forward Filtering Backward Smoothing (FFBS) to produce the posterior distributions needed by the EM algorithm. While estimation results on SPT data have been quite good (Ashley and Andersson, 2015), these SMC elements come with a high computational workload. To alleviate this, we developed a simplified scheme that replaced the SMC methods with an Unscented Kalman Filter (UKF) and an Unscented Rauch-Tung-Striebel Smoother (URTSS); we refer to this as U-EM (Lin and Andersson, 2019). The U-EM has a much lower computational burden than SMC-EM, allowing the method to be applied to larger data sets with little loss of estimation performance. Both the estimation accuracy and computational efficiency have been validated by applications on Brownian motion and Ornstein-Uhlenbeck (OU) motion, however, the standard UKF approximates model noise to be Gaussian for mean and variance propagation. In addition, the specific type of UKF we consider in this work assumes the model noise be additive (Särkkä, 2013). This restriction limits the application of U-EM to a class of motion models that, notably, does not include confined diffusion.

Confined diffusion is a particularly important model in biophysics, describing, for example, the dynamics of particles inside vesicular compartments, motion in membranes, and the motion of T-cells (Hilzenrat et al., 2020). Note that confined diffusion is a nonlinear motion model driven by non-Gaussian noise (Kusumi et al., 1993). To simplify the motion process, some existing work approximates confined motion using an Ornstein-Uhlenbeck model that describes the diffusion of a particle attached to an elastic tether; this model can be expressed in a form with additive Gaussian noise and the parameters connected to the confinement lengths in a confined model (Calderon,

2016). However, using a true confined motion model provides both more flexibility in terms of the shape of the confinement region as well as a hope for more accurate estimation of the relevant parameters. As discussed above, this setting is not amenable to U-EM. It can, however, be handled using the more general SMC-EM technique. As noted above the computational complexity of SMC-EM is high, a situation made worse by the complexity of the one-step probability distribution under the confined model. We address this computational complexity through three modifications to SMC-EM. The first two of these use approximations to the state transition density for the confined diffusion model as well as simplifications to the Bessel function that describes the optical measurements. These approximations significantly reduce the computational burden with little to no loss in achieved estimation accuracy. The third modification is to replace the SIR filter and FFBS with more efficient versions to produce the posterior densities for EM. Specifically, we use an alternative Gaussian Particle Filter (aGPF) (Kotecha and Djuric, 2003) and a Backward Simulation Particle Smoother (BSPS) (Godsill et al., 2004).

With this computationally efficient version of SMC-EM, we are able to extend the method into more realistic but also more complicated models. For example, the original camera model was suitable for Charge-Coupled Devices (CCDs) or Electron Multiplying CCDs (EMCCDs) where the readout noise statistics were common across the device. However, the use of scientific Complementary Metal-Oxide Semiconductor (sCMOS) devices has become extremely common in recent years due to their sensitivity, detection efficiency, large image size, and high frame rates. Unlike CCD/EMCCD cameras, CMOS/sCMOS devices have pixel-dependent readout noise. To extend the impact of our SMC-EM method, we have incorporated these models into our estimation.

In addition to moving to sCMOS cameras, the field of SPT is increasingly turning

towards engineered PSFs. The standard PSF of a sub diffraction-limit sizes particle is the Born-Wolf Point Spread Function (BW-PSF). This PSF is symmetric about the focal plane of the instrument and reveals very little information about the axial (z) location of the particle. Engineered PSFs have been developed to encode more information in the image about the axial direction. Two of the most popular are the the Double-Helix PSF (DH-PSF) (Pavani et al., 2009b) and Astigmatic PSF (A-PSF) (Kao and Verkman, 1994). The DH-PSF is created by inserting a phase plate into the Fourier plane of the output light and is designed to produce a PSF with a pair of lobes in the image plane. The particle is located at the center of the two lobes and the orientation of the line between those lobes gives the axial location. The A-PSF is generated by introducing a cylindrical lens into the light path, generating an elliptical bright spot whose major and minor axes change as a function of the axial position of the particle. Incorporating both the DH-PSF and Astigmatic PSF into our framework allows for the joint estimation of particle trajectory and model parameters for full 3D motion from a sequence of 2D images.

Because SPT experiments are often photon-impoverished and subject to significant background, it is important to consider the impact of signal and noise levels when comparing different analysis algorithms. For example, (Saunter, 2010) investigated the performance of an experimental method in error estimation techniques across a variety of signal and noise values, the comparison work in (Newby et al., 2018) included the signal level as a core factor in their simulations, (Newby et al., 2018) generated simulated videos at various levels of signal to noise ratios to validate the use of convolutional neural networks on SPT data, and (Granik et al., 2019) applied deep learning to analyze particle trajectories based on simulated data over a large range of signal to noise ratios. Though the standard SPT methods perform well at high signal levels, many begin to fail as the signal level decreases or noise level

increases. This motivates us to compare our EM based methods (e.g., SMC-EM and U-EM) to the standard methods across a wide variety of Signal-to-background ratios (SBRs), defined as the ratio between peak signal intensity and background noise.

The SMC-EM algorithm, and, in fact nearly all existing SPT analysis algorithms, assume an input that is a sequence of images that are cropped from a large data set and that focus on a single particle. To generate an end-to-end tool, we therefore must provide a technique for extracting these sequences from the raw images. As this is essentially a pattern recognition problem, we seek to apply machine learning tools to detect particles in an image and extract cropped images for each particle that can be linked into those sequences suitable for downstream analysis.

With the rapid development of machine learning, many researchers have leveraged machine learning tools in both superresolution imaging and SPT, see, e.g., (Nehme et al., 2018; Newby et al., 2018; Helgadottir et al., 2019; Cheng et al., 2021). Most of the published approaches take in raw data and return either localization results, trajectories, model parameters, or some combination of these. While one can certainly use the localizations to extract cropped images in each frame, we hypothesize that a network trained specifically for the detection task may outperform one repurposed for that role. Moreover, most existing approaches rely on a Convolutional Neural Network (CNN). In the field of computer vision however, object detection is often done using more complex networks, such as a Residual Network (ResNet) (He et al., 2016) or the Feature Pyramid Network (FPN) (Lin et al., 2017a). For object detection in computer vision, we modified the aforementioned neural network architectures to construct Particle Identification Networks (PINs) so as to detect particles of interest in an image and extract a small image around each particle.

In this work, we develop three different Particle Identification Networks (PINs) and train them on simulated data to do particle detection and extraction of cropped

images. The first, PIN_{CNN}, uses a plain CNN architecture and serves as a baseline by which to compare the other two. The second, $\text{PIN}_{\text{ResNet}}$, uses the ResNet structure while the final, PIN_{FPN}, uses an FPN. We consider data based on three different optical PSFs: a standard PSF modeled by a simple Gaussian profile (see, e.g. (Zhang et al., 2007)), an astigmatic PSF (Kao and Verkman, 1994), and a Double Helix PSF (Pavani et al., 2009a). We train, validate, and test using simulated data generated across a wide range of SBRs. To validate the approach on experimental data, we deploy the same networks on images of fluorescently labeled AMPA receptors in live primary rat hippocampal neurons, imaged at different intensities to create different SBR settings.

1.2 State of the Art and Weaknesses

In this section, we describe the current state-of-the-art for localization, parameter estimation, and particle detection and discuss their weaknesses, focusing on those that motivated the work in this thesis.

1.2.1 Methods for Localization Refinement

As previously noted, SPT localization algorithms typically assume that their input is sequence of images where each image contains a single particle. The input images are obtained via post process where coarse localization estimation has been done to segment the raw images and extract regions around individual particles. Recall that we use the phrase "localization refinement" to refer to the process of localizing a particle from one of these extracted regions.

Two of the most common approaches to localization refinement are GF (Thompson et al., 2002; Anthony and Granick, 2009) and MLE (Smith et al., 2010; Huang et al., 2013). When the Signal-to-Noise Ratio (SNR) is high and the PSF of the instrument can be described by a Gaussian model or Gaussian mixture model, both GF and MLE

return very accurate localization results. The MLE, however, is based on optimal estimation theory and is more easily extended to more complicated observation models such as those based on images from an sCMOS camera. It has been demonstrated that, as expected, when the SNR is very high, the localization precision returned by MLE is very close to that by Cramer-Rao Lower Bound (CRLB), establishing that under these conditions the MLE (nearly) achieves the best possible precision (Abraham et al., 2009; Mortensen et al., 2010; Smith et al., 2010; von Diezmann et al., 2017). However, when at lower SNR levels (particularly when it is less then four), or when the observation model is more complex (Smal et al., 2009; Chenouard et al., 2014), the MLE no longer returns reliable estimates. However, SPT experiments are typically photon-impoverished and subject to significant background levels, these results motivate us to seek more accurate tools to do localization refinement.

1.2.2 Methods for Parameter Estimation

Once a trajectory has been determined from the image sequence, it is then analyzed to estimate model parameters, most commonly using a MSD analysis (Saxton and Jacobson, 1997; Michalet, 2010), though maximum likelihood techniques have also been applied (Berglund, 2010; Calderon, 2016). While the MSD remains more popular due in large part to its simplicity, the MLE has consistently been shown to yield more accurate results (Smal et al., 2009; Berglund, 2010; Chenouard et al., 2014).

These standard methods are most effective when there is a large signal and a low background intensity in the images with a SNR of at least five. As with localization algorithms, the performance of parameter estimation fails outright when the signal intensity gets low. Because SPT experiments almost always work with low signal levels, developing methods that work well under these conditions is a key challenge.

1.2.3 Machine Learning for Particle Detection

As noted above, machine learning techniques have begun to find application in SPT and in super-resolution imaging (Nehme et al., 2018; Newby et al., 2018; Helgadottir et al., 2019; Cheng et al., 2021), taking in raw camera images to return either localization results, trajectories, model parameters, or some combination of these. Still in their infancy, most of these methods apply simple (though deep) Convolutional Neural Networks (CNNs) and do not take advantage of more advanced structures that have been developed in the field of computer vision.

1.3 Contribution of the Thesis

Overall, this thesis makes contributions to the problem of particle detection and extraction from raw camera images, as well as to those of state and parameter estimation. These contributions can be summarized in three groups.

In the first group, we build upon the existing SMC-EM method to create a generic EM-based framework for joint localization and parameter estimation from a sequence of images of a single particle.

The second group addresses the particle detection and extraction problem to create data to feed into downstream analysis algorithms. Unlike existing ML methods in SPT, we focus on the particle detection problem and leverage better performing network structures.

The final group links together the particle detection and extraction methods with the SMC-EM framework to produce and end-to-end method to take in raw camera data and produce particle trajectories and model parameter estimates for multiple particles in an image sequence.

Below we detail specific contributions in these three groups.

1. **Creation of U-EM**

While the previously established SMC-EM approach can handle nearly arbitrary nonlinearities in both the motion and observation models, the computational complexity of the particle filtering scheme and numerical methods used in the maximization step severely limit its applicability. We address this issue by replacing the particle-based methods with an UKF and URTSS (Merwe and Wan, 2004; Särkkä, 2013). This Sigma Points based EM scheme, which we call U-EM, significantly reduces the computational burden, allowing it to be applied to larger data sets and to more complicated models. This reduction in complexity comes, of course, at the cost of generality in the posterior distribution describing the position of the particle at each time point since the UKF-URTSS approximates this distribution as a Gaussian while the particle-based approaches can represent other distributions (Särkkä, 2013).

2. **Comparison of noise model transformation approaches for U-EM**

One of the challenges in applying the standard UKF is that it approximates noise in both the state update and measurement equations as Gaussian. The observation models concerned in this work, however, involve Poisson distributions where the Poisson rate includes noise whose parameters depend upon the state and experimental settings. Thus, to apply the UKF, the model must be transformed into one where the measurement noise is Gaussian distribution instead of Poisson. We explore two possible approaches: the use of a variance stabilizing transformation, such as the Anscombe or Freeman-Tukey transform, that yields a measurement model with additive Gaussian noise with unity variance, and a straightforward replacement of the Poisson distribution by a Gaussian with a mean and variance equal to the rate of the original distribution. Our results indicate that for SPT data, the Anscombe transformation provides the best performance across a wide range of SBRs.

3. **Generic framework as a guidance for SPT studies**

   A third contribution of this thesis is the creation of a generic EM based framework that allows users to select a variety of filtering and smoothing algorithms based on the specific problem at hand, choosing computationally light schemes (such as U-EM) when allowed for by the motion models, all the way up to full nonlinear filtering and smoothing. We also design and implement quantitative comparisons among different versions of our EM based methods against standard SPT methods such as GF combined with MSD (GF-MSD) or with MLE (GF-MLE or GF-$\text{MLE}_{\text{sCMOS}}$ when the images are acquired by an sCMOS camera). This comparison looks at performance across a wide range of influential factors, including signal intensity, background noise level, type of readout noise, and diffusion speed. These studies demonstrate that, if there are enough photons and a good SBR, then GF-MLE (or $\text{MLE}_{\text{sCMOS}}$), SMC-EM, and U-EM perform similarly well and all outperform GF-MSD. Given the additional computational complexity of the EM-based methods over GF-MLE/$\text{MLE}_{\text{sCMOS}}$, it makes more sense to apply these more standard algorithms in this setting. At low signal levels, however, the EM-based methods outperform the others. The choice between the different EM schemes is dictated in large part by the computation time but also by the ratio of the (expected) diffusion coefficient and the sampling rate. If this ratio is low, U-EM offers good results at significantly less computation time than the SMC-EM methods. Based on the quantitative investigation and analysis, we summarized a guideline for users to choose an appropriate algorithm according to their requirement for time or accuracy.

4. **Application of EM based framework on sCMOS data**

   A fourth contribution is an extension of the existing algorithms to data captured using an sCMOS camera. Due to their relatively low cost, high speed,

and performance, sCMOS cameras are becoming popular tools for SPT data acquisition and including them in our EM-based approach extends the impact our algorithms can have.

5. **Application of EM based framework on multiple PSFs**

   The use of engineered PSFs is becoming important in SPT and analysis algorithms must be adapted to take advantage of the additional information they provide. Our fifth contribution is to include models of two common PSFs, namely the DH-PSF and the A-PSF, demonstrating performance in simulation and showing how additional PSFs could be incorporated in future work.

6. **Incorporation of time-varying parameters in SPT**

   Our sixth contribution is the incorporation of an optimal estimation algorithm for systems with time-varying model parameters. Our approach combines our U-EM-based approach to estimation in the SPT setting with a sliding window scheme to estimate parameters over time. Results from simulations show that our approach is successful in tracking time-varying diffusion constants at a range of physically relevant signal levels.

7. **Computationally efficient variants of SMC-EM**

   Our seventh contribution is the development of three modifications to SMC-EM aimed at improving its computational efficiency. The first two modifications use approximation methods to reduce the complexity of the original motion and measurement models without significant loss of accuracy. The third modification replaces the previous SMC methods with a simplified Gaussian particle filter combined with a backward simulation particle smoother, trading off some level of generality for improved computational performance. We take advantage of the improved efficiency to investigate the effect of data length on performance

in localization and parameter estimation and demonstrate it through analysis of simulated SPT data of a particle in a three dimensional confined environment.

8. **Creation, testing, and comparison of three deep neural networks for particle detection**

   Our eighth contribution is the creation, testing, and comparison of three different deep networks for particle detection in raw camera images with SPT data. The first, PIN_{CNN}, is based on a standard CNN structure that has previously been successfully applied in particle detection and localization. The second, $\text{PIN}_{\text{ResNet}}$, uses a ResNet architecture that is significantly deeper than the CNN while the third, PIN_{FPN}, is based on a more advanced FPN that can take advantage of multi-scale information in an image. All networks are trained using the same collection of simulated data created with a range of SBRs and fluorescence emitter densities, as well as with three different PSFs, a standard Born-Wolf model, the A-PSF, and the DH-PSF. Our results show that at high signal levels, all three networks perform extremely well, indicating that the simpler and smaller PIN_{CNN} is sufficient when the image quality is high. At low SBR levels, however, the PIN_{CNN} is significantly outperformed by both the $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN}. All three PINs were first demonstrated using simulated images with a variety of emitter densities and then using experimental images of labeled AMPA receptors in rat hippocampal neurons imaged under both high and low light signal levels.

9. **Combination of the generic EM based method and deep neural network**

   Our final contribution is to combine our generic approach to localization and parameter estimation with $\text{PIN}_{\text{ResNet}}$ to produce an end-to-end scheme for SPT

data analysis. We demonstrate this approach through both simulations and the AMPA receptor data.

1.4 Organization of the Thesis

The remainder of this thesis is organized as follows. In the next chapter, we introduce the fundamental models behind the observed particles moving under the fluorescent microscopy. In terms of the motion model, three most common types are taken into consideration: Brownian Motion, Ornstein-Uhlenbeck Motion, and Confined Motion. In terms of the observation models, we describe the general model and its specialization to sCMOS cameras, as well as the incorporation of multiple PSFs. In the third chapter, based on conventional tools for SPT analysis, we summarize the traditional two-step procedure to localization and physical parameter estimations in SPT and summarize our generic framework built upon EM.

In Chapter 4, we develop U-EM. To adapt U-EM to the complicated measurement models that are important in SPT, we introduce three model transformation approaches and make a quantitative comparison among them. Then, we conduct an overall comparison among SPT methods in estimation problems. The compared SPT methods include GF, MSD, MLE, U-EM, and SMC-EM.

In Chapter 5, we start to describe the limitations of U-EM and the prior SMC-EM methods and then develop modifications to SMC-EM to speed up the computation without significantly affecting the estimation accuracy. We demonstrate these methods through simulations of a particle diffusing in a confined environment. We improve the computational efficiency through two mathematical modifications: a simplification of motion model and a simplification of measurement model. These modifications are validated through simulations using models of both EMCCD cameras and sCMOS cameras.

Chapter 6 describes our approach to using deep learning to address the problem of the particle detection and image extraction problem in order to obtain the desired input to EM framework. These networks are compared and then validated against both simulation and experimental data.

In Chapter 7, we combine the EM based framework and PINs to do automatic particle detection, extraction, and physical parameter estimation. The combined method is validated based on both simulation and experimental data. In the final chapter, we briefly discuss about open questions and future directions of the research work.

Chapter 2

Relevant Models in Single Particle Tracking Microscopy

2.1 Motion Models

Assuming the motion process for each axis is independent and distinct, we first consider a generic motion model in a given direction given by

$$x_{t+1} = f(x_t, w_t), \tag{2.1}$$

where x_t denotes the state of the system at time t in one direction, and w_t denotes a stochastic process. The model (2.1) can describe a variety of models important to biomolecular motion, including pure Brownian diffusion, Ornstein-Uhlenbeck dynamics, confined diffusion, and more. Note that the motion of the biological macromolecules of interest in SPT occurs in fluids with a length scale of nanometers, the dynamics are in the Stokes regime, meaning that inertial effects can be ignored. As a result the state is almost always simply the position of the particle.

In what follows, we describe three fundamental motion models which are special types of (2.1). Throughout this work, we assume the motion in each of the three dimensions is independent of the others. For simplicity of presentation, we describe the motion models below in one dimension for the purpose of easy presentation.

2.1.1 Brownian Motion

A standard discrete time Brownian motion is given by

$$x_{t+1} = x_t + w_t, \quad w_t \sim \mathcal{N}(0, Q). \tag{2.2}$$

where $x_t \in R$ represents the one dimensional position of the particle in the lateral plane at time t and Q is a covariance given by

$$Q = 2D\Delta t. \tag{2.3}$$

Here Δt is the discretized period of time given by the frame rate of the camera, and D is the (unknown) diffusion coefficient that we would like to estimate. The one transition probability distribution is given by

$$p(x_{t+1}|x_t) = \frac{1}{\sqrt{4\pi D\Delta t}} \exp\left[-\frac{(x_{t+1} - x_t)^2}{4D\Delta t}\right]. \tag{2.4}$$

2.1.2 Ornstein Uhlenbeck Motion

The Ornstein-Uhlenbeck (O-U) process can capture the motion of a biomolecule tethered to a substrate using a flexible linker. The dynamics of an O-U process are

$$x_{t+1} = ax_t + w_t, \quad w_t \sim \mathcal{N}(0, Q). \tag{2.5}$$

Motivated by the model presented in (Zhang et al., 2007), we set a and Q in (2.1) as

$$a = e^{-A\Delta t} \tag{2.6a}$$

$$Q = \frac{D(1 - e^{-2A\Delta t})}{A} \tag{2.6b}$$

where $A > 0$, the stiffness coefficient of the linker and D, the diffusion coefficient are

the unknown parameters to be estimated.

From (2.6), the state transition probability density between two successive images in one direction is

$$p(x_{t+1}|x_t) = \sqrt{\frac{A}{2\pi D(1-e^{-2A\Delta t})}} \exp\left\{ -\frac{A}{2D}\left[\frac{(x_{t+1}-x_t e^{-A\Delta t})^2}{1-e^{-2A\Delta t}}\right]\right\} \tag{2.7}$$

2.1.3 Confined Motion

We consider the situation where a particle of interest is diffusing in a rectangular "corral", with each axis bound to an interval. We assume the x position is confined within the interval $[-L_x/2, L_x/2]$ and that the dynamics follow a diffusion with reflective boundary conditions (Saxton and Jacobson, 1997). The state transition probability function in the x direction is then

$$\begin{aligned} p(x_{t+1}|x_t) =& \frac{1}{L_x} \\ &+ \frac{2}{L_x}\sum_{n=1}^{N_p=\infty} \exp\left[\frac{-D\Delta t n^2\pi^2}{L_x^2}\right] \cos\left[\frac{n\pi}{L_x}(x_{t+1}+\frac{L_x}{2})\right] \cos\left[\frac{n\pi}{L_x}(x_t+\frac{L_x}{2})\right]. \end{aligned} \tag{2.8}$$

The y (and z if necessary) directions follow analogous dynamics. The diffusion coefficient and confinement length (in each axis) are the unknown parameters to be estimated.

2.2 Observation Models

2.2.1 CCD/EMCCD Camera

Because the particle of interest in SPT is smaller than the diffraction limit of light, the image shape on the camera arising from the emitted fluorescence is largely determined by the PSF of the instrument. Assuming segmentation has already been done, the image acquired by the camera is composed of P^2 pixels arranged into a $P \times P$ square

array (note that the value of P s typically between 5 to 20 with the specific value depending on the particular imaging setting). The pixel size is Δx by Δy with the actual dimensions determined both by the physical size of the camera elements on the camera and by the magnification of the optical system. At time step t, the expected photon intensity measured for the p^{th} pixel, $\lambda_{p,t}$, can be calculated via

$$\lambda_{p,t} = G \cdot \int_{x_{p,t}^{min}}^{x_{p,t}^{max}} \int_{y_{p,t}^{min}}^{y_{p,t}^{max}} PSF\left(x_t - \xi, y_t - \xi'\right)\, d\xi d\xi', \tag{2.9}$$

where G denotes the peak signal intensity, (x_t, y_t) is the position of the particle, and the integration bounds $(x_{p,t}^{min}, x_{p,t}^{max},\ y_{p,t}^{min}, y_{p,t}^{max})$ are over the boundaries of the p-th pixel. The details about the PSF and the corresponding mathematical expression will be discussed in Chapter 2.2.3 - 2.2.5.

In addition to the signal, there is always a background intensity rate arising from out-of-focus fluorescence and autofluorescence in the sample. This can be modeled locally as a constant rate N_{bgd} over the small $P \times P$ array of the segmented images (Ashley and Andersson, 2015). Usually, the value of the backgroud noise is measured experimentally and for the rest of the work, we assume it is known (though its value can be estimated using the EM algorithm). Due to the shot noise inherent to the photon generation process brought by CCD/EMCCD cameras, the measured photons in the p^{th} pixel at time t can be described by a Poisson process

$$I_{p,t} \sim \text{Poiss}(\lambda_{p,t} + N_{bgd}), \tag{2.10}$$

where Poiss$(\cdot)$ represents a Poisson distribution. Throughout this work, we define the SBR as the ratio of the peak rate G to the background rate N_{bgd}. In simulation studies, these terms are known exactly while in experimental work they must be estimated from the data.

2.2.2 sCMOS Camera

To maximize the signal level, experiments often use sensitive detectors such as Electron Multiplied Charge Coupled Device (EM-CCD) or scientific Complementary Metal-Oxide Semiconductor (sCMOS) cameras. sCMOS cameras in particular have become quite popular due to their (relatively) low cost, high resolution, frame rate, and quantum efficiency, and their large field of view (Huang et al., 2013; Watanabe et al., 2017). These cameras, however, bring pixel-dependent noise characteristics that, together with the Poisson-process nature of photon generation, must be accounted for, particularly at low signal levels.

To simulate an sCMOS camera, we include pixel-dependent readout noise in the measurement model through the choice of distributions for $\epsilon_{p,t}$ in Eq (2.10). We base our measurement model on a Hamamatsu ORCA Flash 4.0 camera described in (Huang et al., 2013).

From (Huang et al., 2013), the probability density function (PDF) of the measured ADU (analog-to-digital units) counts $C_{p,t}$ in the p^{th} pixel at time t is a combination of the Poisson photon noise and the Gaussian read-out noise and can be expressed as

$$P(C_{p,t}) = A \sum_{q=0}^{\infty} \frac{1}{q!} \exp^{-\mu_{p,t}} \mu_{p,t}^{q} \frac{1}{\sqrt{2\pi \mathrm{Var}_{p,t}}} \exp^{-\frac{\left[\frac{C_{p,t}-O_{p,t}}{g_{p,t}} - q\right]^2}{2\mathrm{Var}_{p,t}/g_{p,t}^2}} \tag{2.11}$$

where A is a normalizing constant, $\mu_{p,t}$ is the number of expected photon electrons arising from signals and background noise, $g_{p,t}$ is the amplification gain, $O_{p,t}$ and $\mathrm{Var}_{p,t}$ are the offset and variance of the readout noise in pixel p at time t, respectively.

From this distribution, the measured photon counts $I_{p,t}$ for the p^{th} pixel at time

t is given by

$$\frac{I_{p,t}}{QE} \triangleq \frac{C_{p,t} - O_{p,t}}{g_{p,t}} \approx \text{Poiss}(\mu_{p,t}) + \epsilon_{p,t}, \tag{2.12a}$$

$$\epsilon_{p,t} \sim \mathcal{N}(0, \sigma^2_{p,t}), \tag{2.12b}$$

$$\sigma^2_{p,t} = \frac{\text{Var}_{p,t}}{g^2_{p,t}}. \tag{2.12c}$$

where QE is the quantum efficiency of the camera pixels. In this work, for simplicity, we ignore the offsest as it is just a constant shift, and set QE as 1 for simplicity. Therefore, in our case, the measured photon counts $I_{p,t}$ for the p^{th} pixel at time t can be expressed as

$$I_{p,t} \sim \text{Poiss}(\mu_{p,t}) + \epsilon_{p,t} \tag{2.13}$$

with

$$\mu_{p,t} = \lambda_{p,t} + N_{bgd} \tag{2.14}$$

where $\lambda_{p,t}$ is the expected photon intensity arising from signals, following the double integral form as (2.9), N_{bgd} is the photon counts arising from background noise.

The corresponding PDF of the measured photon counts in pixel p at time t is given by

$$P(I_{p,t}) = \sum_{q=0}^{\infty} \frac{1}{q!} \exp\left[-(\lambda_{p,t} + N_{bgd})\right] (\lambda_{p,t} + N_{bgd})^q \frac{1}{\sqrt{2\pi\sigma^2_{p,t}}} \exp\left[-\frac{(I_{p,t} - q)^2}{2\sigma^2_{p,t}}\right]. \tag{2.15}$$

where all values of $\text{Var}_{p,t}$ and $g_{p,t}$ that define $\sigma_{p,t}$ depend on the type of camera sensors used for imaging. If the readout noise is neglected, we set $\epsilon_{p,t} = 0$; if a CCD/EMCDD camera sensor is modeled, we set $\sigma_{p,t}$ as a constant; if a CMOS/sCMOS is modeled, the value depends on the pixel properties.

Our specific model for an sCMOS camera is based on that of (Huang et al., 2013).

We fit the variance data in that work with the model

$$\log_{10}\text{Occurance} = a \cdot \exp\left(-b \cdot \text{variance}\right), \tag{2.16}$$

finding the best fit parameters to be $a = 4.734, b = -0.001799$. Similarly, we fit the gain data to the model

$$\log_{10}\text{Occurance} = a_1 \cdot \exp -(\frac{g - b_1}{c_1})^2 + a_2 \cdot \exp -(\frac{g - b_2}{c_2})^2, \tag{2.17}$$

leading to the values

$$a_1 = 2.215, b_1 = 2.19, c_1 = 0.07661, a_2 = 2.68, b_2 = 2.249, c_2 = 0.216.$$

Fig. 2·1 shows the gain and variance maps of the pixel-dependent readout noise for a 512×512 image, as well as the curve fitting results.

2.2.3 Born-Wolf Point Spread Function

The Born-Wolf Point Spread Function (PSF) at a three dimensional position (x, y, z) can be described as

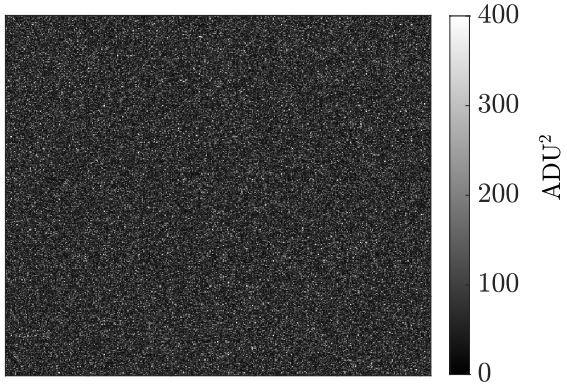
$$PSF(x, y, z) = \left| C \int_0^\alpha \sqrt{cos(\theta)} \mathcal{J}_0(\kappa \sin\theta \sqrt{x^2 + y^2}) \exp(-i\kappa z \cos\theta) \sin\theta d\theta \right|^2 \tag{2.18}$$

where C is a complex constant that normalizes the model, α and κ are physical constants, and $\mathcal{J}_0$ denotes a zeroth-order Bessel function of the first kind.

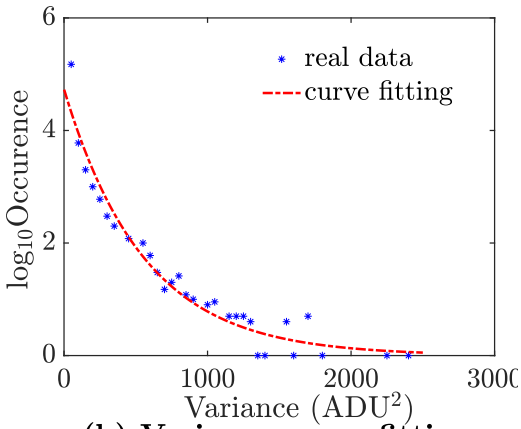
For images acquired in 2D, the PSF can be simplified as

$$PSF_{BW}(x, y; x_o, y_o) = h_{BW} \cdot \exp\left(-\frac{(x-x_o)^2}{2\sigma_x^2} - \frac{(y-y_o)^2}{2\sigma_y^2}\right), \sigma_x = \sigma_y = \frac{\sqrt{2}\lambda}{2\pi \text{NA}}, \tag{2.19}$$

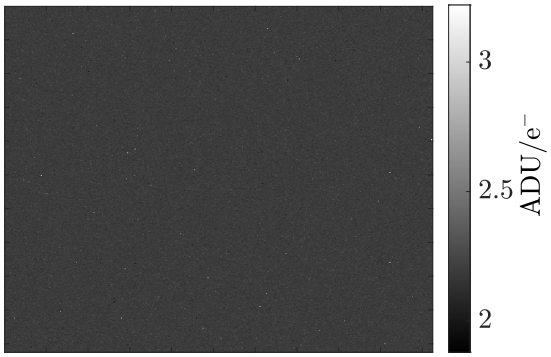
where (x_o, y_o) is the position of the particle, h_{BW} is the normalization factor used in two ways in this thesis. The photon rate in a given pixel is then given by inserting



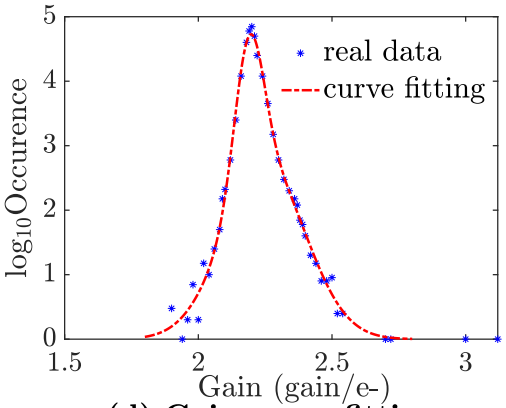
(a) Simulated variance map.



(b) Variance curve-fitting.



(c) Simulated gain map.



(d) Gain curve-fitting.

Figure 2·1: Simulated maps and histograms of the pixel-dependent readout noise.

(2.19) into (2.9). Prior Chapter 6, we set$h_{BW} = \frac{1}{\Delta x \Delta y}$ to normalize by the area of the pixel. From Chapter 6 on, we set h_{BW} to

$$h_{BW} = \int_A PSF_{BW}(u, v; 0, 0) du dv = 1 \tag{2.20}$$

where A is the area of a pixel region determined by integration bounds ($x_{p,t}^{min}, x_{p,t}^{max}$, $y_{p,t}^{min}, y_{p,t}^{max}$) to normalize the PSF to one in the given pixel containing the particle. Note that formally, the expression in (2.19) is a convolution of the PSF with a Dirac delta function at the position of the particle; for simplicity of exposition throughout this thesis, we refer both as a PSF; the meaning should be clear from context.

A typical image formed with the Born-Wolf PSF at a low signal level (here, $N_{bgd} = 1$, $G = 10$) is shown in the left-side image of Fig. 2·2, while an image at a higher signal level ($N_{bgd} = 10$, $G = 100$) is shown in the right-side image.

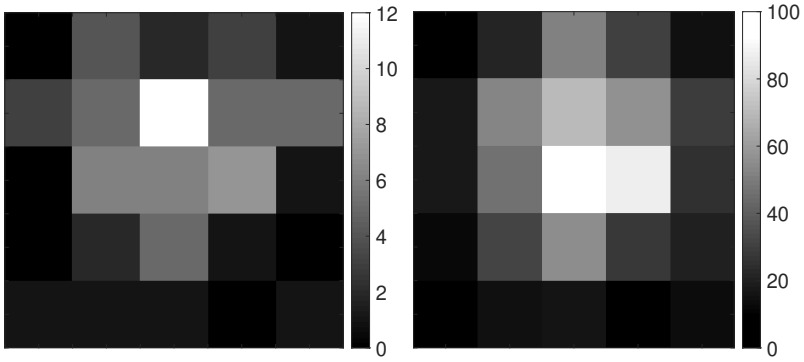


Figure 2·2: Typical images at low and high signal levels. (left) $N_{bgd} = 1$ and $G = 10$, (right) $N_{bgd} = 10$ and $G = 100$. Notice the different scaling in the two images.

2.2.4 Double-Helix Point Spread Function

The DH-PSF, illustrated in Fig. 2·3, uses a phase plate in the output path of the microscope to generate a PSF such that the particle is at the center between two lobes that rotate around each other as the particle moves along the axial direction

(von Diezmann et al., 2017). While the true shape of the DH-PSF is non-trivial, it

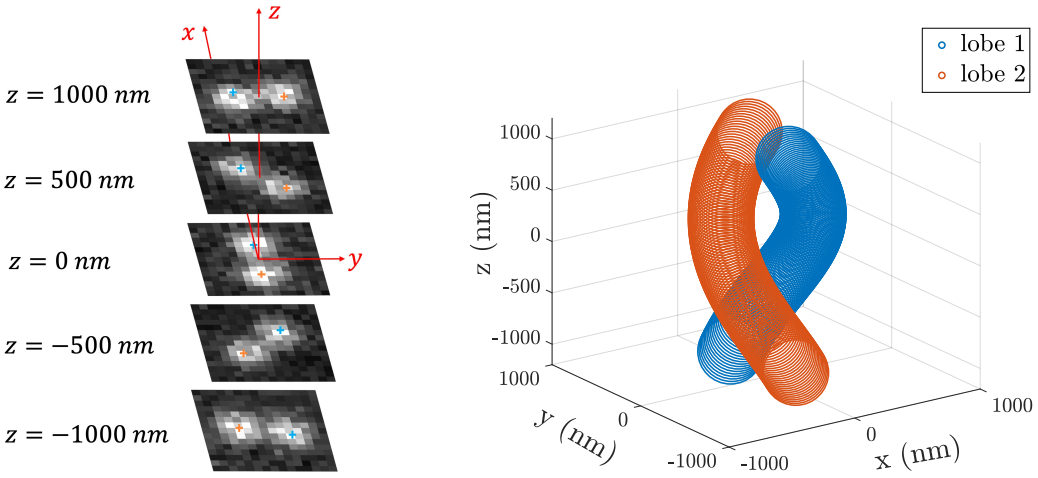


Figure 2·3: The DH-PSF generates two lobes that rotate in the $x-y$ plane as the particle moves in the z direction. (left) A sequence of simulated images of a particle at different z locations. (right) an equi-intensity profile of the DH-PSF in (x, y, z), demonstrating the choice of the "double-helix" name.

can be well-approximated by a pair of Gaussian lobes (Pavani et al., 2009b),

$$PSF(x,y) = h_{DH} \cdot \left(\exp\left[-\frac{(x-x_1)^2+(y-y_1)^2}{2\sigma_{xy}^2} \right] + \exp\left[-\frac{(x-x_2)^2+(y-y_2)^2}{2\sigma_{xy}^2} \right] \right), \quad (2.21)$$

where h_{DH} is again a normalization factor used in the same two ways as before, and (x_i, y_i) are the centers of the lobes ($i = 1$ or 2), given by

$$\begin{bmatrix} x_{1,2} \\ y_{1,2} \end{bmatrix} = \begin{bmatrix} x_p \\ y_p \end{bmatrix} \pm r \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}, \quad (2.22)$$

where r is a constant and (x_p, y_p) is the location of the particle. The angle of the lobes, θ, is a monotonic function of the axial position with a specific relationship that depends on the instrument.

This relationship is typically obtained by generating a calibration curve by scanning a single feature along the axial direction and measuring the resulting PSF (von

Diezmann et al., 2017). In this work, for simplicity we take a linear model

$$\theta = -kz, \tag{2.23}$$

where coefficient k can be found via fitting to data (Schechner et al., 1996). Note that the k can be simplified to be a constant when the axial distance is not very large (von Diezmann et al., 2017; Lin et al., 2021).

2.2.5 Astigmatic Point Spread Function

The Astigmatic PSF (A-PSF) is generated by inserting a cylindrical lens into the light path. The resulting PSF is an elliptical spot whose major and minor axes change as a function of the axial position of the particle. This PSF is well-approximated by

$$PSF_{Astig}(x, y, z; x_o, y_o) = h_{Astig} \cdot \exp\left(-\frac{(x-x_o)^2}{2\sigma_x^2(z)} - \frac{(y-y_o)^2}{2\sigma_y^2(z)}\right), \tag{2.24}$$

where h_{Astig} is a normalization factor such that $\int_A PSF_{Astig}(u, v, 0; 0, 0)dudv = 1$, A is the area of a pixel region; (x_o, y_o) is the location of the particle in the $x-y$ plane, and the PSF width σ_x and σ_y as a function of z can be fit into a defocusing curve given by

$$\sigma_{x,y}(z) = \sigma_0\sqrt{1 + (\frac{z-c}{d})^2 + A(\frac{z-c}{d})^3 + B(\frac{z-c}{d})^4}. \tag{2.25}$$

We select parameters from (Smith et al., 2010), giving

$A_x = -0.0708, B_x = -0.073, c_x = 0.389, d_x = 0.531, A_y = 0.164, B_y = 0.0417, c_y = -c_x, d_y = d_x, \sigma_{x0} = 1.08, \sigma_{y0} = 1.01.$

Fig. 2·4 shows the simulated A-PSF width $\sigma_{x,y}$ as a function of z.

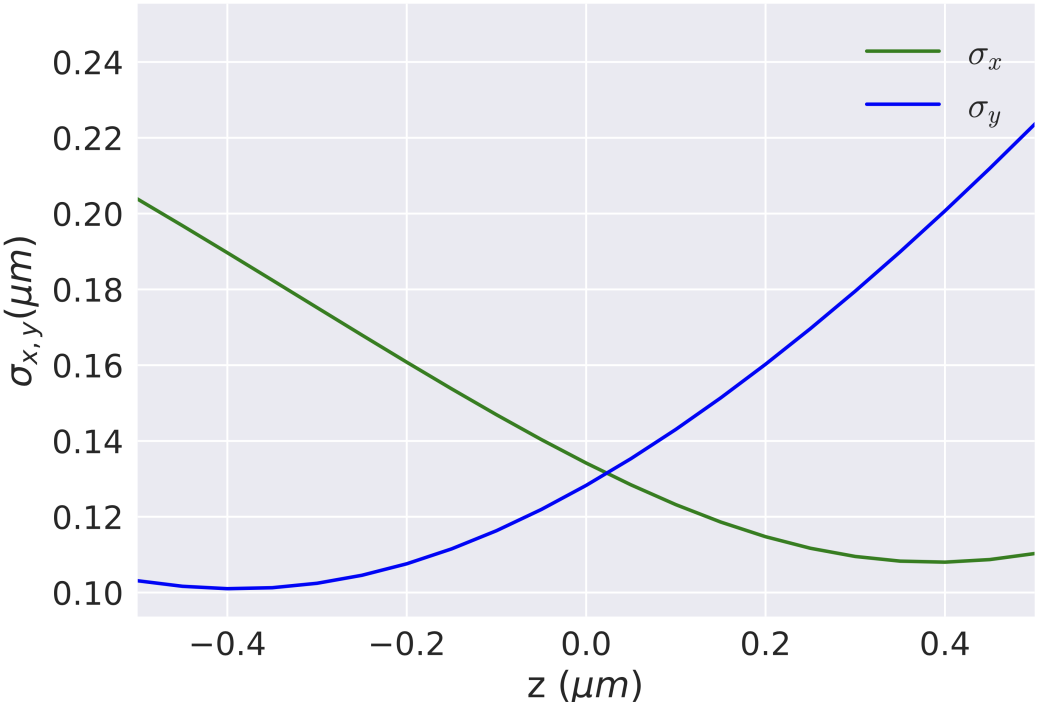


Figure 2·4: Astigmatic PSF width as a function of z.

Chapter 3

Generic Framework for State Estimation

3.1 Standard Two-step Framework

As described in Chapter 1, a typical SPT analysis of a sequence of images begins with an image segmentation step where the raw images are post-processed to extract image sequences that each contain information about a single particle. These sequences are processed to determine particle trajectories and motion model parameters, enabling the ability to do non-causal estimation.

Under the standard paradigm, a two-step process is applied in which the particle is first localized in each image and these positions linked across frames to create a trajectory; this trajectory is then analyzed to extract information about physical parameters. This two-step paradigm, summarized in Fig. 3·1, separates trajectory estimation from model parameter identification despite the fact that these two problems are coupled.

While there are a variety of algorithms in the SPT literature for localization, one of the most common is that of Gaussian Fitting (GF). Similarly, while there are many methods for estimating parameters, most biophysicists apply a Mean Squared Displacement (MSD) analysis or, less commonly, Maximum Likelihood Estimation (MLE). As we use these three techniques throughout this thesis as a baseline for comparison to our own methods, below we briefly describe each of the them in the SPT context.

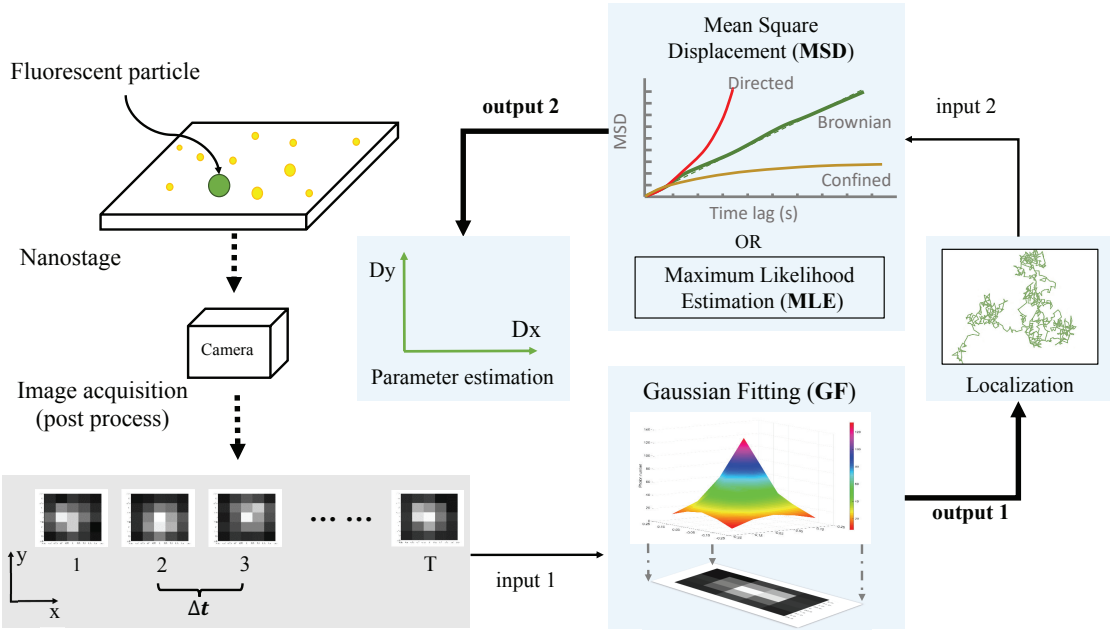


Figure 3·1: Illustration of the standard two-step approach. Segmented image data is first passed through a localization step where an algorithm such as GF determines the position of the particle in each frame. The resulting trajectory is then analyzed using, e.g., the MSD or MLE to determine model parameters.

3.1.1 Gaussian Fitting (GF)

In the 2-D setting, the PSF of the instrument is well approximated by a Gaussian. As a result the measured intensity, I_{xy} can be described as

$$I_{xy} = G \exp\left(-\frac{(x-x_o)^2}{2{\sigma_x}^2} - \frac{(y-y_o)^2}{2{\sigma_y}^2}\right) + N_{bgd}, \tag{3.1}$$

where G is the peak amplitude of the intensity, (x, y) are the lateral coordinates in the image frame, (x_o, y_o) are the position of the particle, (σ_x, σ_y) are physical parameters describing the width of the PSF, and N_{bgd} is the background intensity. Fitting the measured data to this model allows one to estimate the particle position as well the other model parameters in (3.1).

3.1.2 Mean Squared Displacement (MSD)

MSD is one of the most frequently used methods for estimating diffusion coefficients from trajectory data. Following (Saxton and Jacobson, 1997), the single-axis MSD is given by

$$\text{MSD}(n) = \frac{1}{N-n}\sum_{i=1}^{N-n}\left(r_{i+n} - r_i\right)^2, \quad n = 1, ..., N-1, \tag{3.2}$$

where N is the data length and r_i is the position of the fluorescent particle in either x or y in frame i. For a particle moving with a diffusion coefficient of D, the expectation of the MSD is given by

$$\mathbb{E}\left[\text{MSD}(n)\right] = 2Dn\Delta t, \tag{3.3}$$

where Δt is the time interval between frames of the image sequence. In this work we fit calculated MSD curves to the model in (3.3) using the nonlinear least-squares curve fitting solver *lsqnonlin* in MATLAB (MathWorks, Natick, MA).

3.1.3 Maximum Likelihood Estimation (MLE)

While the MSD remains popular and is simple to use, it relies on several user choices and is known to lack robustness with respect to measurement noise. An alternative is to use optimal estimation theory. In particular, the MLE has good statistical properties as it is both efficient and consistent, achieving the Cramer-Rao lower bound (so long as sufficient data is available) (Michalet and Berglund, 2012). Consider the problem of identifying an unknown parameter $\theta \in \mathbb{R}^{n_\theta}$ for an arbitrary state space model

$$X_{t+1} = f_t(X_t, w_t, \theta), \tag{3.4a}$$

$$Y_t = h_t(X_t, v_t, \theta), \tag{3.4b}$$

where X_t is the (vector) state of the system at time t, Y_t is the (vector) observation at time t, w_t and v_t are independent white noise processes, and θ_t is the unknown (vector) parameter to be estimated. In the context of SPT, the state is the two or three dimensional position of the particle, the function f_t is defined by a particular motion model (see Chapter 2.1), and the observation vector is the collection of pixel values from the current image, reshaped into a P^2-length vector. The parameter vector may include unknown motion model parameters (e.g. diffusion coefficients and confinement lengths) as well as unknown parameters defining the observation model (e.g. peak intensity or PSF width). The MLE determines an estimate of this parameter by maximizing the log likelihood of the observed data $Y_{1:N} \triangleq \{Y_1, \ldots, Y_N\}$,

$$\hat{\theta} = \arg\max_{\theta} \log p_\theta(Y_{1:N}), \tag{3.5}$$

where $p_\theta(Y_{1:N})$ is the joint probability density of the observations $Y_{1:N}$ defined by the model in (3.4). We use a computationally efficient version of the ML estimator developed in (Berglund, 2010) for estimating the diffusion coefficient and the

variance of observation noise v_t under the assumption of a simple diffusion motion model and a linear observation of the position corrupted by zero-mean Gaussian noise. See (Berglund, 2010; Michalet and Berglund, 2012) for details.

3.2 Expectation Maximization Framework

The basic tool behind our approach of simultaneous localization and parameter estimation is the EM algorithm (see, e.g., (Särkkä, 2013)), an iterative approach for finding an ML estimate. Based on EM, we created a generic framework for SPT analysis shown in Fig 3·2.

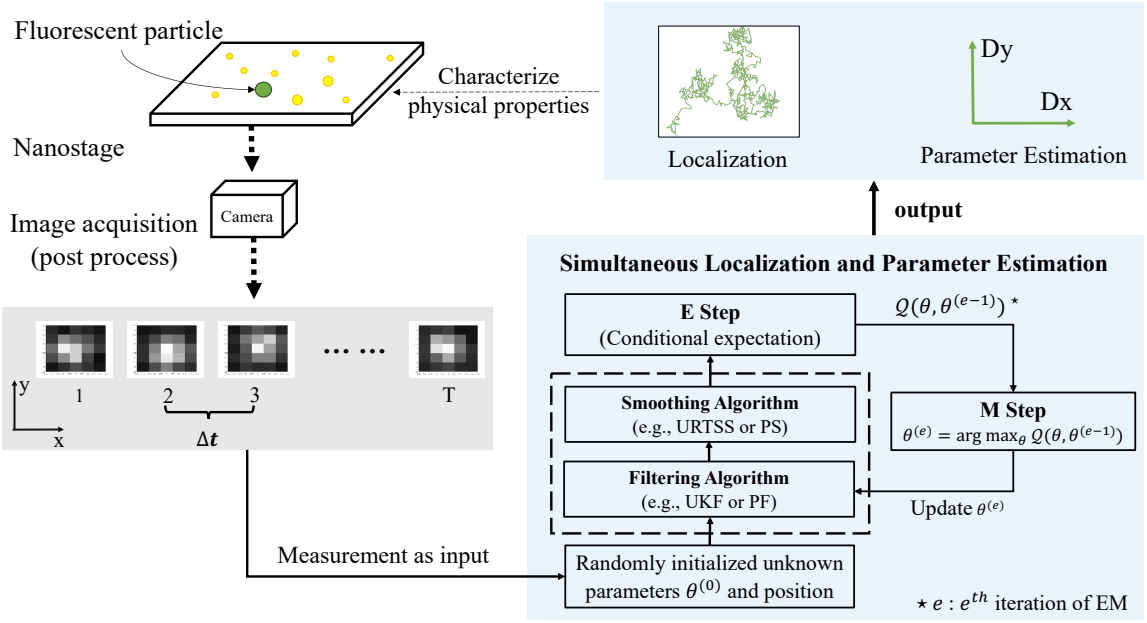


Figure 3·2: Illustration of the EM-based framework for simultaneous localization and parameter estimation. Segmented image data is passed directly to the estimation routine where EM alternates between filtering/smoothing to find the distribution of the particle trajectory and estimation of the parameter based on that distribution.

In what follows, we briefly describe our approach and the two main flavors of it used in this thesis (additional variants that improve computational performance

are presented in later chapters). Note that the EM approach does not produce a point estimate for the particle location in each frame but rather an estimate of the smoothed probability distribution of its location and thus provides more information than the GF. For the purposes of this work, we obtain the particle location by taking the mean of this distribution in each frame; however, other estimators could be used.

Consider once again the state space model in (3.4). In general, the log-likelihood of the observations, $\log p_\theta(Y_{1:N})$, is intractable or cannot be written analytically. As a result, (3.5) cannot be solved directly. The EM algorithm handles this through an iterative approach, forming an approximation to the likelihood function at the e^{th} step, named $\mathcal{Q}\left(\theta, \theta^{(e)}\right)$, based on a current estimate of the parameter $\theta^{(e)}$, and then optimizing this to find the next estimate $\theta^{(e+1)}$, stepping towards the MLE (Dempster et al., 1977). The approximation is given by the conditional expectation of the joint log likelihood function,

$$\mathcal{Q}\left(\theta,\ \theta^{(e)}\right) = \mathbb{E}_{\theta^{(e)}}\left[L_\theta(X_{0:N}, Y_{1:N})|Y_{1:N}\right], \tag{3.6}$$

where θ is the unknown parameter, $X_{0:N} = \{X_0, X_1, \cdots, X_N\}$ is known as a *hidden state* that, in the context of SPT, is given by the unknown particle locations, and $L_\theta(X_{0:N}, Y_{1:N})$ is the joint log likelihood function of the trajectory and observations.

This function is given by

$$L_\theta(X_{0:N}, Y_{1:N}) = \log p_\theta(X_0) + \sum_{t=1}^{N} \log p_\theta(X_t|X_{t-1}) + \sum_{t=1}^{N} \log p_\theta(Y_t|X_t). \tag{3.7}$$

Using (3.7) in (3.6) yields

$$\mathcal{Q}(\theta,\ \theta^{(e)}) = I_1(\theta, \theta^{(e)}) + I_2(\theta, \theta^{(e)}) + I_3(\theta, \theta^{(e)}), \tag{3.8}$$

where

$$I_1(\theta, \theta^{(e)}) = \mathbb{E}\left[\log p(X_0|\theta)|Y_{1:N}, \theta^{(e)}\right], \tag{3.9a}$$

$$I_2(\theta, \theta^{(e)}) = \sum_{t=1}^{N} \mathbb{E}\left[\log p(X_t|X_{t-1})|Y_{1:N}, \theta^{(e)}\right], \tag{3.9b}$$

$$I_3(\theta, \theta^{(e)}) = \sum_{t=1}^{N} \mathbb{E}\left[\log p(Y_t|X_t)|Y_{1:N}, \theta^{(e)}\right]. \tag{3.9c}$$

The calculation of $\mathcal{Q}\left(\theta, \theta^{(e)}\right)$ is called the *Expectation (E) step* at the e^{th} iteration. It has been shown that any choice of $\theta^{(e+1)}$ such that $\mathcal{Q}(\theta^{(e+1)}, \theta^{(e)}) \geq \mathcal{Q}\left(\theta^{(e)}, \theta^{(e)}\right)$ ensures the EM algorithm converges to a local maximum of the likelihood function. Thus, the expectation step is followed by a *Maximization (M) step* to produce the next estimate of the parameter,

$$\theta^{(e+1)} = \arg\max_{\theta} \mathcal{Q}\left(\theta, \theta^{(e)}\right) \tag{3.10}$$

Despite the fact that convergence is only guaranteed to a local optimum, EM has been shown to work well in practice (Dempster et al., 1977; McLachlan and Krishnan, 2007). To implement the E step (that is, to calculate $\mathcal{Q}$) by carrying out the expectations in (3.9), it is necessary to know the posterior densities $p(X_t|Y_{1:N})$ and $p(X_t, X_{t-1}|Y_{1:N})$. If the underlying model in (3.4) is linear with Gaussian noise, then these distributions are easily obtained using a Kalman filter and a Kalman smoother (Gibson and Ninness, 2005). For nonlinear observations, however, these distributions must often be approximated in some way. The choice of how to handle these nonlinearities defines the flavor of our method.

3.2.1 Unscented - EM (U-EM)

U-EM approximates the posterior densities in (3.9) as Gaussians using a UKF and a URTSS. The UKF was developed in (Julier and Uhlmann, 1997) as an alternative to the Extented Kalman Filter, capturing (an approximation to) the mean and covariance of a nonlinear stochastic process without relying on linearization or a Jacobian computation. U-EM starts with the UKF to get the estimated state and covariance, and then uses the URTSS to return the posterior probability densities required for the EM algorithm.

Consider a generic dynamic state space system depending on a parameter θ, of which generic form is described the same as (3.4). Note that the UKF in requires a model in which both the process and measurement noise terms appear in an additive fashion. To make this explicit, we replace the general state space dynamic model (3.4) with

$$X_{t+1} = f_t(X_t, \theta) + w_t(\theta), \tag{3.11a}$$

$$Y_t = h_t(X_t, \theta) + v_t(\theta). \tag{3.11b}$$

The U-EM scheme starts with a UKF to get the filtered estimate of the state X and then applies a URTSS to return the posterior probability densities $p(X_t|Y_{1:N})$ needed for the EM algorithm where N denotes the total number of observations and $t = 1, ..., N$. These steps are described below.

Unscented Kalman Filter (UKF)

The goal of the UKF is to form a Gaussian approximation of the distribution of the state X (the location of the particle being tracked in the SPT data). Generically, this distribution is given by

$$p(X_t|Y_{1:t}) \simeq \mathcal{N}(\mathbf{m}_t, \mathbf{P}_t), \quad t = 1, ..., N \tag{3.12}$$

where $\mathcal{N}(\mathbf{m}_t, \mathbf{P}_t)$ denotes a Gaussian (also known as a Normal) distribution of mean $\mathbf{m}_t$ and covariance $\mathbf{P}_t$. These distributions at each time step are found as follows.

1. Prediction step: First calculate the deterministic *sigma points*, $\mathcal{X}$, according to

$$\mathcal{X}_{t-1}^{(0)} = \mathbf{m}_{t-1} \tag{3.13a}$$

$$\mathcal{X}_{t-1}^{(i)} = \mathbf{m}_{t-1} + \sqrt{(n+\zeta)}\left[\sqrt{\mathbf{P}_{t-1}}\right]_i, \ (i = 1, \ldots, n) \tag{3.13b}$$

$$\mathcal{X}_{t-1}^{(i+n)} = \mathbf{m}_{t-1} - \sqrt{(n+\zeta)}\left[\sqrt{\mathbf{P}_{t-1}}\right]_i, \ (i = 1, \ldots, n) \tag{3.13c}$$

where n is the dimension of the state, $\mathbf{m}_0$ and $\mathbf{P}_0$ are randomly initialized, $[\cdot]_i$ denotes the i^{th} column of the matrix, $\sqrt{A}$ is the matrix square root of A, and ζ is a scaling parameter defined by $\zeta = \alpha^2(n+\kappa) - n$. The parameters α and κ are determined by the user to tune the algorithm performance. The parameter α determines the spread of the sigma points around the mean and is usually taken in the interval $(0, 1]$, while κ is a secondary scaling parameter which is usually set to $3 - n$ (see (Särkkä, 2013) for details).

The sigma points are then propagated through the motion model

$$\hat{\mathcal{X}}_t^{(i)} = f(\mathcal{X}_{t-1}^{(i)}, \theta), \ i = 0, ..., 2n \tag{3.14}$$

and combined to produce the predicted mean and covariance at time t given data up to time $t-1$ according to

$$\mathbf{m}_t^- = \sum_{i=0}^{2n} W_i^{(m)} \hat{\mathcal{X}}_t^i, \quad \mathbf{P}_t^- = \sum_{i=0}^{2n} W_i^{(c)} (\hat{\mathcal{X}}_t^{(i)} - \mathbf{m}_t^-)(\hat{\mathcal{X}}_t^{(i)} - \mathbf{m}_t^-)^T + Q_{t-1} \tag{3.15}$$

where Q_{t-1} is the covariance matrix of the process noise w_t in (3.4a). The

weights in (3.15) are given by

$$W_0^{(m)} = \frac{\zeta}{n+\zeta} \tag{3.16a}$$

$$W_0^{(c)} = \frac{\zeta}{n+\zeta} + (1-\alpha^2+\beta),\ i = 1,\ldots,2n \tag{3.16b}$$

$$W_i^{(m)} = W_i^{(c)} = \frac{1}{2(n+\zeta)},\ i = 1,\ldots,2n \tag{3.16c}$$

Here β is used to incorporate prior knowledge of the distribution of state X_t (with $\beta = 2$ used for Gaussian distributions (Särkkä, 2013)).

2. Update and filter: A new set of sigma points, $\mathcal{X}_t^-$, are formed from the predicted mean and covariance according to (3.13) using $\mathbf{m}_t^-$ and $\mathbf{P}_t^-$ in place of $\mathbf{m}_{t-1}$ and $\mathbf{P}_{t-1}$. These sigma points are then propagated through the measurement model (3.4b)

$$\hat{\mathcal{Y}}_t^{(i)} = h(\mathcal{X}_t^{-(i)}, \theta),\ i = 0, ..., 2n. \tag{3.17}$$

and combined to form

$$\mu_t = \sum_{i=0}^{2n} W_i^{(m)} \hat{\mathcal{Y}}_t^{(i)}, \tag{3.18a}$$

$$S_t = \sum_{i=0}^{2n} W_i^{(c)} (\hat{\mathcal{Y}}_t^{(i)} - \mu_k)(\hat{\mathcal{Y}}_t^{(i)} - \mu_k)^T + \mathbf{R}_t, \tag{3.18b}$$

$$C_t = \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{X}_t^{-(i)} - \mathbf{m}^-)(\hat{\mathcal{Y}}_t^{(i)} - \mu_t)^T, \tag{3.18c}$$

$$K_t = C_t S_t^{-1}, \tag{3.18d}$$

where $\mathbf{R}_t$ is the covariance matrix of the measurement noise v_t in (3.4b). Finally, these are used to produce the filtered estimates of the mean and covariance of the process at time t using the data up to time t through

$$\mathbf{m}_t = \mathbf{m}_t^- + K_t \left[Y_t - \mu_t \right], \quad \mathbf{P}_t = \mathbf{P}_t^- - K_t S_t {K_t}^T. \tag{3.19a}$$

Unscented Rauch-Tung-Striebel Smoother (URTSS)

The URTSS iterates backwards from $t = N, N-1, ..., 0$, beginning with the final results of the UKF, $\mathbf{m}_T^s = \mathbf{m}_T$ and $\mathbf{P}_T^s = \mathbf{P}_T$, as follows.

1. Prediction and update: Form the sigma points $\mathcal{X}_t$ from (3.13) using $\mathbf{m}_t$ and $\mathbf{P}_t$, propagate them through the motion model

$$\hat{\mathcal{X}}_{t+1}^{(i)} = f(\mathcal{X}_t^{(i)}, \theta), \ i = 0, 1, ..., 2n, \tag{3.20}$$

and then combine the predictions by using

$$\mathbf{m}_{t+1}^- = \sum_{i=0}^{2n} W_i^{(m)} \hat{\mathcal{X}}_{t+1}^{(i)}, \quad \mathbf{P}_{t+1}^- = \sum_{i=0}^{2n} W_i^{(c)} (\hat{\mathcal{X}}_{t+1}^{(i)} - \mathbf{m}_{t+1}^-)(\hat{\mathcal{X}}_{t+1}^{(i)} - \mathbf{m}_{t+1}^-)^T + Q_t, \tag{3.21a}$$

$$D_{t+1} = \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{X}_t^{(i)} - \mathbf{m}_t)(\hat{\mathcal{X}}_{t+1}^{(i)} - \mathbf{m}_{k+1}^-)^T, \tag{3.21b}$$

where the weights are given in (3.16).

2. Produce the smoothed estimate: The mean and covariance of the smoothed Gaussian density at time t are calculated from

$$\mathcal{G}_t = D_{k+1} \left[P_{t+1|N}^- \right]^{-1}, \tag{3.22a}$$

$$\mathbf{m}_{t|N}^s = \mathbf{m}_t + \mathcal{G}_t (m_{t+1|N}^s - \mathbf{m}_{t+1}^-), \tag{3.22b}$$

$$\mathbf{P}_{t|N}^s = \mathbf{P}_t - \mathcal{G}_t (\mathbf{P}_{t+1|N}^s - \mathbf{P}_{t+1}^-) \mathcal{G}_t^T. \tag{3.22c}$$

E-step via UKF and URTSS

From the results of the UKF and URTSS, the (approximate) posterior densities needed for the EM algorithm are

$$p(X_t|Y_{1:N}) \sim \mathcal{N}(\mathbf{m}^s_{t|N}, \mathbf{P}^s_{t|N}), \tag{3.23a}$$

$$p(X_t, X_{t-1}|Y_{1:N}) \sim \mathcal{N}\left(\begin{bmatrix}\mathbf{m}^s_{t|N}\\ \mathbf{m}^s_{t-1|N}\end{bmatrix}, \begin{bmatrix}\mathbf{P}^s_{t|N} & \mathbf{P}^s_{t|N}\mathcal{G}^T_{t-1}\\ \mathcal{G}_{t-1}\mathbf{P}^s_{t|N} & \mathbf{P}^s_{t-1|N}\end{bmatrix}\right). \tag{3.23b}$$

This gives the approximation of the $\mathcal{Q}$ function to be

$$\begin{aligned}\mathcal{Q}(\theta, \hat{\theta}^{(i)}) \approx & -\frac{1}{2}\log(2\pi\mathbf{P}_0) - \frac{1}{2}\log(2\pi\mathbf{Q}) - \frac{1}{2}\log(2\pi\mathbf{R})\\ & - \frac{1}{2}tr\{\mathbf{P}_0^{-1}\left[\mathbf{P}^s_{0|N} + (\mathbf{m}^s_{0|N} - \mathbf{m}_0)(\mathbf{m}^s_{0|N} - \mathbf{m}_0)^T\right]\}\\ & - \frac{1}{2}\sum_{t=1}^{N} tr\{\mathbf{Q}^{-1}\mathbb{E}\left[(x_t - f(x_{t-1}))(x_t - f(x_{t-1}))^T|Y_N\right]\}\\ & - \frac{1}{2}\sum_{t=1}^{N} tr\{\mathbf{R}^{-1}\mathbb{E}\left[(y_t - h(x_t))(y_t - h(x_t))^T|Y_N\right]\}.\end{aligned} \tag{3.24}$$

where $\mathbf{Q}, \mathbf{R}$ are the covariance matrices for the motion model and observation model respectively, $\mathbf{P}_0, \mathbf{m}_0$ are initial estimate of the motion covariance and mean state, $\mathbf{P}^s_{0|N}, \mathbf{m}^s_{0|N}$ are smoothed estimates of the motion covariance and mean state at the initial time, and tr denotes the trace operation. As previously noted, in the context of the SPT the unknown parameter vector θ may contain terms from the motion model (such as diffusion coefficients or confinement lengths) as well as from the observation model (such as peak intensity).

M-step for Parameter Estimation

The application of the maximization step under the U-EM scheme yields an analytical expression for the estimate of θ at the e^{th} EM iteration, according to (3.10). The specifics of this step depend on the particular choice of model and will be defined as

needed throughout the remainder of the thesis.

3.2.2 Sequential Monte Carlo - EM (SMC-EM)

For the SMC-EM algorithm, the posterior densities in (3.9) are calculated using a PF and PS, allowing for arbitrary distributions to be estimated. Note that the term "particles" in SMC refers to the random samples used to represent a distribution rather than the fluorescently labeled objects being tracked. In the remainder of the thesis, the meaning of the word "particles" should be clear from context. The details of SMC-EM are as follows.

Consider a generic dynamic state space system as in (3.4). The SMC-EM can use arbitrary number of randomly generated Monte Carlo samples (often referred to as *particles*) to approximate the posterior needed by the EM algorithm. Under the SMC-EM scheme, a PF provides the filtered state estimates and importance weights. These are then passed to a PS that works backwards to produce the smoothed distribution. These steps are described below.

Particle Filter (PF)

For simplicity in exposition, we use a basic sequential importance resampling PF in our basic SMC flavor. Alternative PFs with different numerical properties could also be used (see (Doucet et al., 2009) and later sections of this thesis).

1. Randomly generate M initial particles at time $t = 0$ by drawing from the initial distribution $P_\theta(x_0)$,

$$x_0^i \sim P_\theta(x_0), \quad i = 1, ..., M. \tag{3.25}$$

2. Propagate these M particles through the motion model (3.4a),

$$\tilde{x}_t^i = f_{t-1}(\tilde{x}_{t-1}, w_{t-1}, \theta), \quad i = 1, 2, ..., M. \tag{3.26}$$

3. Compute the importance weights,

$$w_t^i \triangleq \tilde{w}_t^i = \frac{P_\theta(Y_t|\tilde{x}_t^i)}{\sum_{j=1}^M P_\theta(Y_t|\tilde{x}_t^j)}, \quad i,j = 1,...,M. \tag{3.27}$$

where Y_t is the observed measurement at time t and $P_\theta(Y_t|\tilde{x}_t^i)$ is determined by the measurement model in (3.4b).

4. Resample M new particles x_t^j from the discrete distribution,

$$p(x_t^j = \tilde{x}_t^i) = w_t^i, \ j = 1,\ldots,M. \tag{3.28}$$

5. Increment $t \leftarrow t+1$ and iterate from step 2 until $t = N$.

Particle Smoother (PS)

The PS used in our plain SMC flavor is a forward-filtering backward smoothing method. As with the PF, other particle smoothing schemes can be selected.

1. Initialize the importance weights w_t^i at $t = N$ as $w_{N|N}^i = w_t^i$, $i = 1,...,M$.

2. Decrement $t \leftarrow t-1$ and calculate the smoothed weights $w_{t|N}^i$ by backward computation,

$$w_{t|N}^i = \sum_{j=1}^M w_{t+1|N}^j \frac{w_t^i P_\theta(\tilde{x}_{t+1}^j|\tilde{x}_t^i)}{\sum_{l=1}^M w_t^l P_\theta(\tilde{x}_{t+1}^j|\tilde{x}_t^l)}, \tag{3.29}$$

where $P_\theta(\tilde{x}_{t+1}^j|\tilde{x}_t^i)$ is determined by the motion model in (3.4a).

3. Iterate from Step 2 until $t = 0$.

E-step via PF and PS

Using the results of the PF and PS, the $\mathcal{Q}$ function can be approximated by

$$\mathcal{Q}\left(\theta, \theta^{(e)}\right) = I_1(\theta, \theta^{(e)}) + I_2(\theta, \theta^{(e)}) + I_3(\theta, \theta^{(e)}) \tag{3.30}$$

where

$$I_1(\theta, \theta^{(e)}) \approx \sum_{i=1}^{M} w_{1|N}^{i} log\ P_\theta(\tilde{x}_1^i), \tag{3.31a}$$

$$I_2(\theta, \theta^{(e)}) \approx \sum_{t=1}^{N-1} \sum_{i=1}^{M} \sum_{j=1}^{M} w_{t|N}^{ij} log P_\theta(\tilde{x}_{t+1}^j | \tilde{x}_t^i), \tag{3.31b}$$

$$I_3(\theta, \theta^{(e)}) \approx \sum_{t=1}^{N} \sum_{i=1}^{M} w_{t|N}^{i} log P_\theta(y_t | \tilde{x}_t^i). \tag{3.31c}$$

where $w_{t|N}^{ij}$ are given by

$$w_{t|N}^{ij} = \frac{w_t^i w_{t+1|N}^j P_\theta(\tilde{x}_{t+1}^j | \tilde{x}_t^i)}{\sum_{l=1}^{M} w_t^l P_\theta(\tilde{x}_{t+1}^j | \tilde{x}_t^l)}. \tag{3.32}$$

One benefit of SMC-EM over U-EM is its ability to represent arbitrary posterior distributions rather than approximating them as Gaussians.

M-step for Parameter Estimation

The application of the maximization step under the SMC-EM scheme depends on the particular motion model used. For pure diffusion, an analytical solution to the maximization problem can be found but in general numerical methods are needed in the optimization.

Chapter 4

Performance of SPT Methods on Joint Localization Refinement and Parameter Estimation in Two Dimensional Diffusion

In this chapter we describe the U-EM flavor of our general scheme. We first compare different methods of transforming the observation models into a form suitable for the UKF and then compare U-EM both to baselines standard from the field as well as to the more general SMC-variant of our own approach.

4.1 Model Transformations in U-EM

One of the challenges in applying the UKF is that it assumes Gaussian noise in both the state update and measurement equations. In this work we focus on a pure diffusion motion model to focus the discussion on a concrete setting, though the method can handle other motion models. As the dynamic model for diffusion is linear with additive Gaussian noise, applying the UKF in terms of the state update equations is straightforward. The observation model, however, involves Poisson distributed noise whose parameters depend upon the state and experimental settings. Thus, to apply the UKF, the model must be transformed into one where the measurement noise is Gaussian instead of Poisson. Two possible approaches are considered: one is a choice of a variance stabilizing transformation, in particular the Anscombe and Freeman-Tukey transform, that yields a measurement model with additive Gaussian

noise with unity variance; the other is a straightforward replacement of the Poisson distribution by a Gaussian with a mean and variance equal to the rate of the original distribution.

Given the diffusion model described in (2.2) and the description of U-EM in Chapter 3.2.1, the maximization step under the U-EM scheme yields an analytical expression for the estimate of D_x at the e^{th} EM iteration, given by

$$\begin{aligned}\hat{D}_{x,e} = \frac{1}{2N\Delta t}\cdot[&\sum_{t=1}^{N}(\hat{x}^2_{t|N,e} + P_{t|N,e}) + \sum_{t=1}^{N}(\hat{x}^2_{t-1|N,e} + P^2_{t-1|N,e}) \\ &- 2\sum_{t=1}^{N}(\hat{x}_{t|N,e}\cdot\hat{x}_{t-1|N,e} + P_{t,t-1|N,e})]\end{aligned} \tag{4.1}$$

where $\hat{x}_{t|N,e}$ is the estimate of the particle location at time t given data over all N images. The analytical expression for $\hat{D}_{y,e}$ is analogous to (4.1).

4.1.1 Direct Gaussian Approximation

For a sufficiently high rate, a Poisson distribution of rate λ is well approximated by a Gaussian of mean and covariance equal to that rate (Gnedenko, 2017). One approach, then, is to replace (2.10) with

$$\tilde{I}_{p,t} = \tilde{\lambda}_{p,t} + v_k,\ v_k \sim \mathcal{N}\left(0, \tilde{\lambda}_{p,t}\right). \tag{4.2}$$

where $\tilde{\lambda}_{p,t} = \lambda_{p,t} + N_{bgd}$. This approach requires no modification to the measured data. However, the noise term v_k itself depends upon the state variable since the rate $\lambda_{p,t}$ is a function of the position of the particle.

4.1.2 Anscombe Transformation

The Anscombe transformation is a variance-stabilizing transformation that (approximately) converts a Poisson-distributed random variable into a unit variance Gaussian one (Anscombe, 1948). Under this approach, the actual measurements are first trans-

formed by

$$\tilde{I}_{p,t} = 2\sqrt{I_{p,t} + \frac{3}{8}}. \tag{4.3}$$

The measurement model (2.10) is then replaced by

$$\tilde{I}_{p,t} \simeq 2\sqrt{\tilde{\lambda}_{p,t} + \frac{3}{8}} + v_k,\ v_k \sim \mathcal{N}(0,1). \tag{4.4}$$

4.1.3 Freeman Tukey Transformation

An alternative variance stabilizing transform is the Freeman and Tukey (Freeman and Tukey, 1950). Under this approach, the measurements are first transformed by

$$\tilde{I}_{p,t} = \sqrt{I_{p,t} + 1} + \sqrt{I_{p,t}}. \tag{4.5}$$

and the measurement model is replaced by

$$\tilde{I}_{p,t} \simeq \sqrt{\tilde{\lambda}_{p,t} + 1} + \sqrt{\tilde{\lambda}_{p,t}} + v_k,\ v_k \sim \mathcal{N}(0,1). \tag{4.6}$$

Note that a similar transformation can be applied for noise models which are a combination of Poisson noise and Gaussian readout noise. This is particularly important when considering data from an sCMOS camera and will be discussed in Sec. 4.3.

4.1.4 Demonstration and Analysis

To demonstrate the performance of the U-EM algorithm with different transformation methods, we performed several simulations. 40 different ground truth trajectories were generated from the diffusion motion model (2.2) and used to create simulated images according to the observation model in (2.10). The optical parameters and other fixed constants used in these simulations are shown in Table 4.1.

To explore the difference among these transformations, we fixed the background

Table 4.1: Parameter settings in two dimensional Brownian diffusion.

Symbol	Parameter	Values
Δt	Image period (discrete time step)	100 ms
N	Number of images per dataset	100
P	Number of pixels per squared image	25
D_x	Diffusion coefficient in x direction	0.005 μm^2/s
D_y	Diffusion coefficient in y direction	0.01 μm^2/s
Δx	Length of unit pixel	100 nm
Δy	Width of unit pixel	100 nm
λ	Emission wavelength	540 nm
NA	Numerical aperture	1.2

rate $N_{bgd} = 10$ and the peak signal intensity $G = 100$, representing a strong but not atypical signal in actual SPT experiments. The estimated position at each time was taken as the mean value of the smoothed distribution. The resulting root mean square errors (RMSE) are shown in Fig. 4·1. As can be seen, all approaches perform well with an estimation error of approximately 6.25 nm in x position and 7.30 nm in y position. Both the similarity and the actual error level is as expected given that the signal level is high.

Performance of parameter estimation over the 40 runs and with the three different transformation choices is shown in Table 4.2.

Table 4.2: Parameter estimation of D_x and D_y on 40 datasets

Method	D_x (μm^2/s)	D_y (μm^2/s)
Gaussian	0.0047 ± 7.3e-4	0.009 ± 0.0011
Anscombe	0.0046 ± 7.3e-4	0.009 ± 0.0011
Freeman Tukey	0.0046 ± 7.3e-4	0.009 ± 0.0011

The primary differences among the different observation model transformations become meaningful only when the rate of the Poisson distribution is low (as determined by the combination of signal level and background). We performed two sets of simulations at different noise levels. In the first set, the noise N_{bgd} was fixed at

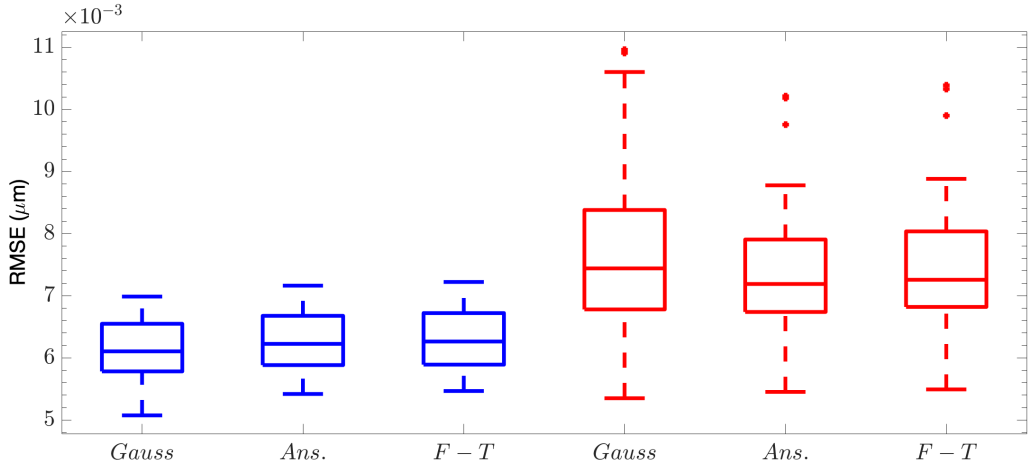


Figure 4·1: Box plots of 2-D position estimation error using the (Gauss) Gaussian approximation, (Ans.) Anscombe transform, and (F-T) Freeman-Tukey transform. Blue and red box correspond to RMSE in x and y position respectively.

one and the signal G increased from one to 10. In the second, SBR was fixed to 10 and N_{bgd} increased from 1 to 10. Other imaging and model parameters were kept the same.

The localization results are shown in Fig. 4·2 with the top graph corresponding to $N_{bgd} = 1$ (and thus extremely low signal levels) and the bottom to simulations for a fixed SBR. In both plots, red corresponds to Gaussian approximation, blue to Anscombe transform and green to Freeman-Tukey transform.

It is clear that differences only appear at the low signal levels. Note that in the first plot with a peak intensity of $G = 6$, the rate in the pixel at the center of the PSF is still only 7 counts. At the lowest signal levels, the Anscombe transform outperforms the other two. While the Gaussian approach is close, the difference between the mean and the center quartiles indicates that it has several large outliers. To put these estimation errors in context, note that for the given imaging parameters, the diffraction limit of light is approximately 270 nm.

The corresponding results for the estimation of the diffusion coefficients are shown

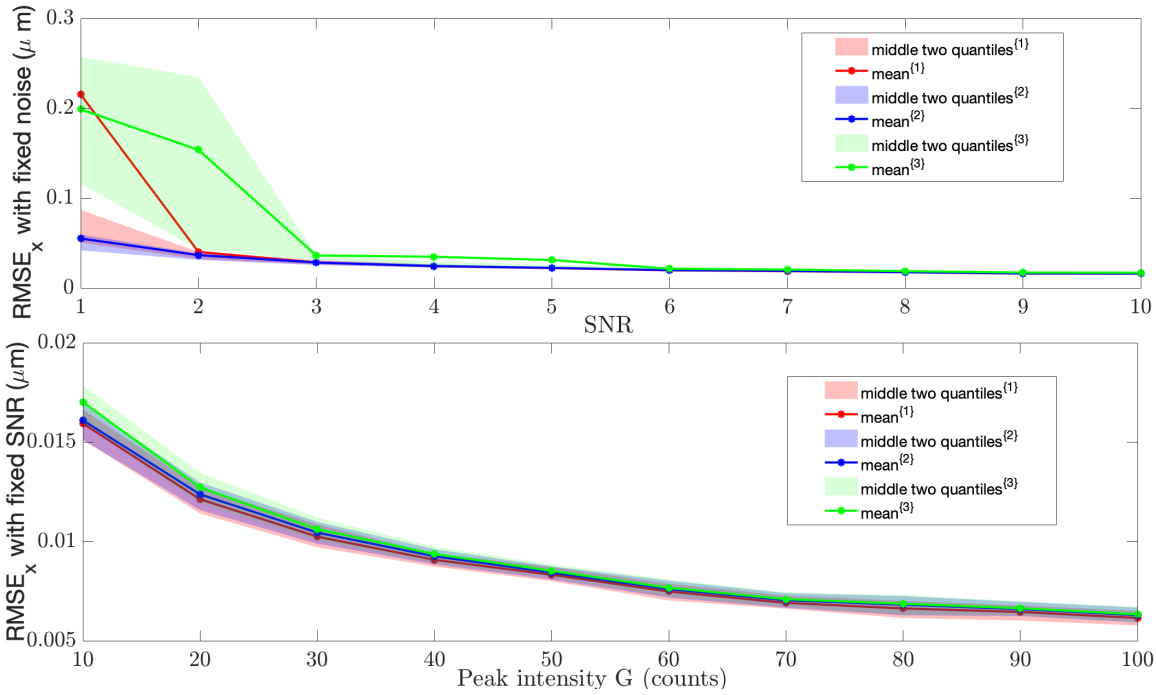


Figure 4·2: RMSE of x position estimation with different $\{N_{bgd},\ G\}$. The superscript $\{1\}$, $\{2\}$ and $\{3\}$ indicates results based on the Gaussian approximation, Anscombe transform, and Freeman-Tukey transform, respectively. (top) Results holding N_{bgd}=1 fixed and varying G, showing the behavior at very low signal levels. (bottom) Results holding the ratio $SBR = 10$ fixed while varying G. Note that for space reasons, only results for D_x are shown; estimation of D_y is similar.

in Fig. 4·3. As before, red corresponds to Gaussian approximation, blue to Anscombe transform, and green to Freeman-Tukey transform. The true value is $D_x = 0.005$ μm/s^2. These results parallel the trajectory estimation, with all transformations of the observation equation being essentially equivalent at high signal levels and Freeman-Tukey failing at the lowest signals.

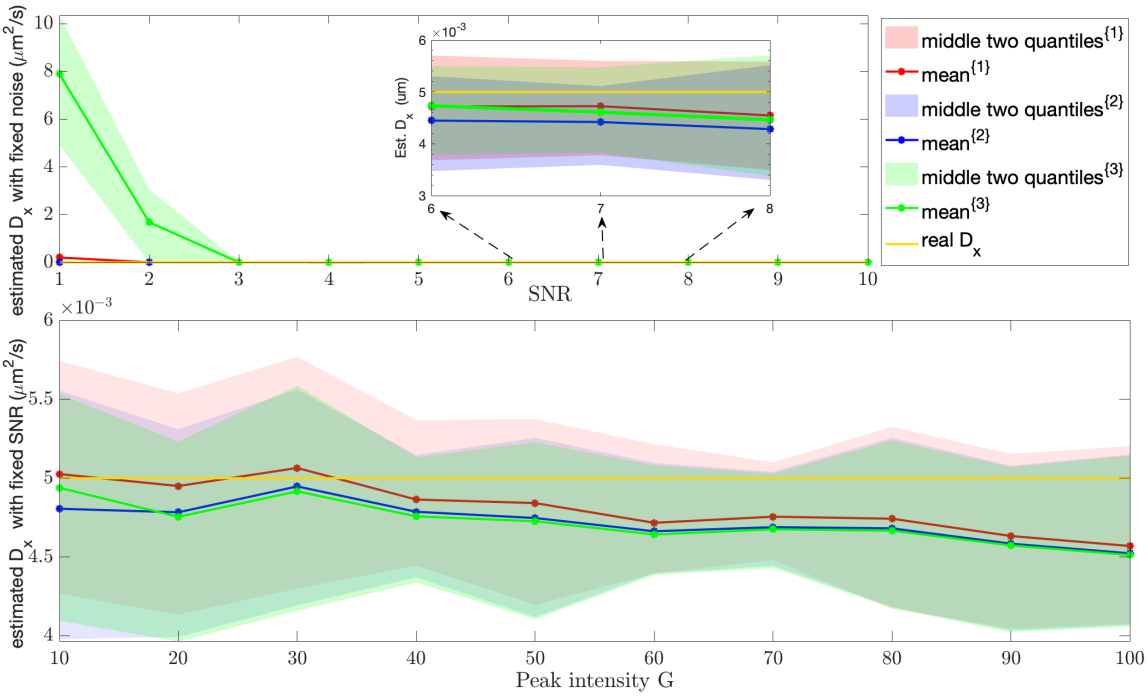


Figure 4·3: Results of estimation of D_x with a true value of $D_x = 0.005$ μm/s^2. (top) Results holding $N_{bgd} = 1$ fixed while varying SBR. (bottom) Results holding the ratio SBR fixed while varying G.

As noted in Chapter 3.2, the primary motivation for U-EM is the reduced computational load and indeed the method is faster then vanilla SMC-EM using any of the three transformation methods (on the order of a few minutes using unoptimized code in Matlab on a standard laptop with 10 EM iterations as opposed to hours with SMC-EM). It is important to note, however, that among the three, the Gaussian approximation runs the slowest (approximately 10-15% slower). This is easily explained from the equations (4.2) - (4.6) where we see that under the Gaussian approxima-

tion, the variance $\lambda_{p,t}$ must be calculated at each time step while both Anscombe and Freeman-Tukey avoid this since they are variance stabilized to one. Since the Anscombe transform outperforms at low signal level and has lower computational load, it should be the preferred approach.

4.2 Overall Comparison among SPT Methods on CCD or EMCCD Data

In this section, we are interested in comparing EM based algorithms against standard SPT methods, such as GF-MSD and GF-MLE, under the influence of different SBR levels, motion blurs and diffusion speeds. We begin by considering the scenario where the motion and observation models are following (2.2) and (2.10) respectively. While we include background noise and a basic model of a pixelated image, we ignore other camera-specific issues such as readout noise. Under this scenario we consider three cases. In Case 1, we evaluate the performance of the algorithms in an experimental setting with a relatively large signal and a low background. In Case 2, we investigate algorithm performance across a range of signal intensity and background levels. In Case 3, we explore the effect of the value of the diffusion coefficient on algorithm performance, considering both an idealized setting with no motion blur and a more realistic setting where motion blur is presented.

Images were simulated for $N = 100$ frames at an imaging period of $\Delta t = 100$ ms (i.e, a frame rate of 10 frames/s) for a total of 10 s. To generate each sequence of images, independent trajectories of length $N \times N_{sub}$ were generated where N_{sub} represents a sub-sampling factor. In practice, cameras accumulate photons over an integration period, and the motion of the particle during the exposure period may affect the estimation accuracy. To replicate this motion blur effect, we assumed the camera accumulated photons continuously during the first $\delta t = 10$ ms of each imaging

period Δt and produced each final frame by averaging the first 10 consecutive images in the period and ignoring the rest. To generate statistics on algorithm performance, $K = 100$ image sequences were generated for every parameter setting. Without particular presentation, the optical settings should remain the same as previous ones. In this section, the particular settings for optical parameters are given in Table 4.3.

Table 4.3: Fixed parameters used in the simulations.

Symbol	Parameter	Value
N	Image sequence length	100
δt	Shutter period	10 ms
N_{sub}	Sub-sampling factor	100
K	Number of sequences	100

4.2.1 Case 1: Observation at High Light Conditions

For this first case, the peak signal level was set to $G = 100$ and the background noise to $N_{bgd} = 10$. (Note that these are the rates in each image after accumulating over the shutter period.) Other imaging parameters were set as described in Table 4.3. The diffusion coefficients were fixed to $D_x = 0.005$ μm^2/s and $D_y = 0.01$ μm^2/s. For all EM based methods, the initial position is usually generated from a predefined distribution. For ease of calculation, we have $x_0 \sim \mathcal{N}(0, 2D\Delta t)$ which is analogous to y_0.

Then the data were analyzed using GF-MSD, GF-MLE, U-EM, and three versions of SMC-EM: SMC^{100}, SMC^{500}, and SMC^{1000} where the superscript denotes the number of Monte Carlo samples used in the PF and PS algorithms. A typical trajectory, together with the position estimates produced by the U-EM algorithm as a typical estimation result, is shown in Fig. 4·4. Under the U-EM scheme demonstrated in Chapter 3.2.1, the analytical solution to diffusion coefficients on Brownian motion is same to (4.1).

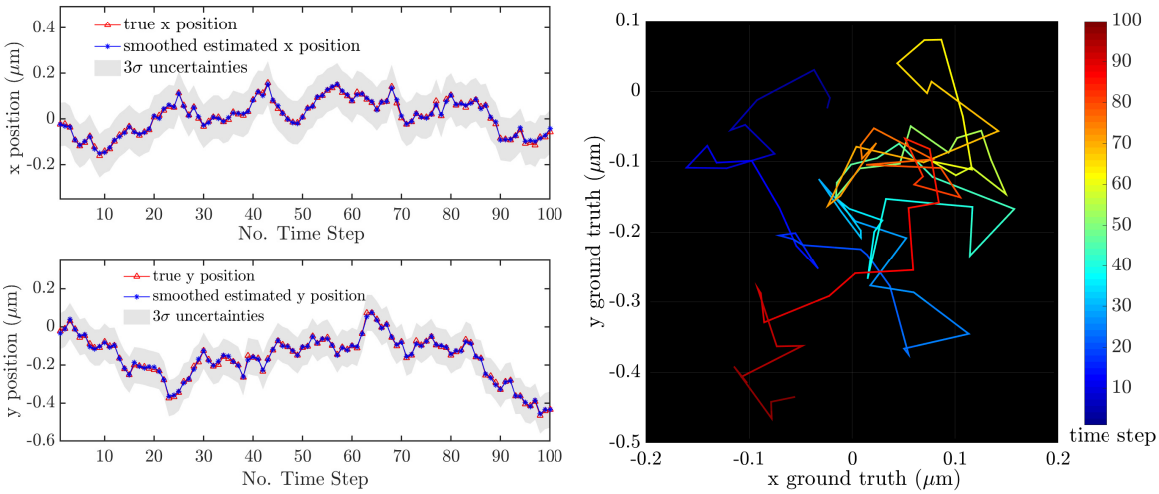


Figure 4·4: A typical trajectory with $D_x = 0.005$ μm^2/s and $D_y = 0.01$ μm^2/s. (left) x and y trajectories together with the position estimates from U-EM and the 3σ error bounds. (right) The ground truth trajectory in the plane with color indicating time.

As described in Chapter 3.2, the EM algorithm at the heart of the simultaneous approach is an iterative scheme, improving the estimate at each iteration as it moves towards a local optimal of the log likelihood function. The application of the maximization step under the SMC-EM scheme in Chapter 3.2.2 yields an analytical expression for the estimate of D_x at the e^{th} EM iteration, given by

$$\hat{D}_{x,e} = \frac{1}{2N\Delta t}(D^a_{x,e} + D^b_{x,e}), \tag{4.7}$$

with

$$D^a_{x,e} \triangleq \sum_{i=1}^{M} w^i_{1|N,e}(x^i_{1|N,e})^2, \quad D^b_{x,e} \triangleq \sum_{k=1}^{N-1}\sum_{i=1}^{M}\sum_{j=1}^{M} w^{ij}_{k|N,e}(x^j_{k+1|N,e} - x^i_{k|N,e})^2.$$

where $x^i_{t|N,e}$ denotes the smoothed state for the i^{th} sampling particle at time t, and M is the number of samples used in the PF/PS. The analytical expression for estimated D_y is analogous to (4.7).

The resulting evolution of the diffusion coefficient parameter estimates over the

100 different trajectories for the three versions of SMC-EM and for U-EM are shown in the box plots in Fig. 4·5. These plots show that the EM algorithm generally converges in a small number of steps and that, as expected, the performance of SMC-EM improves as the number of Monte Carlo samples used in the PF and the PS increases.

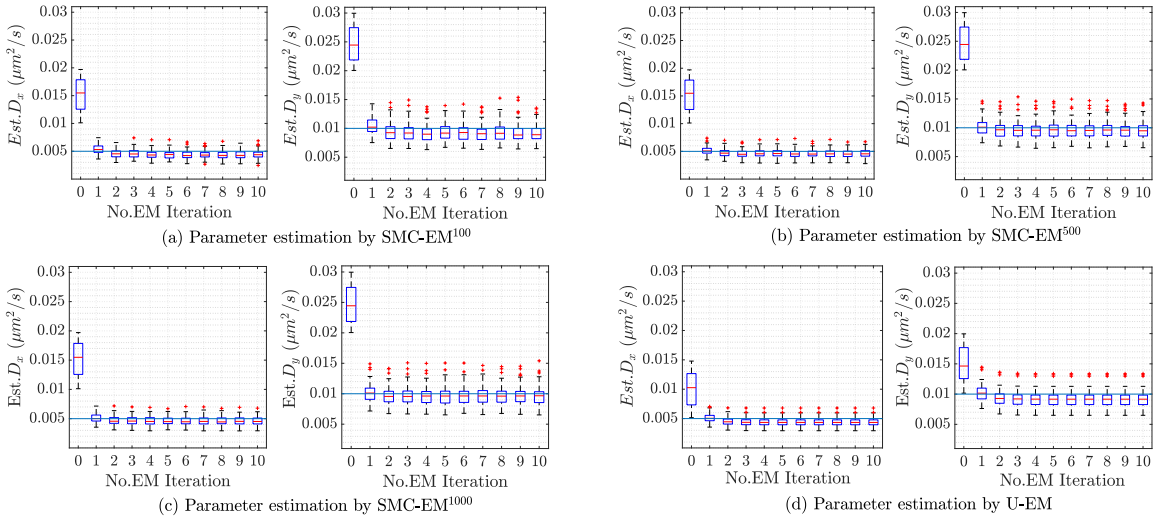
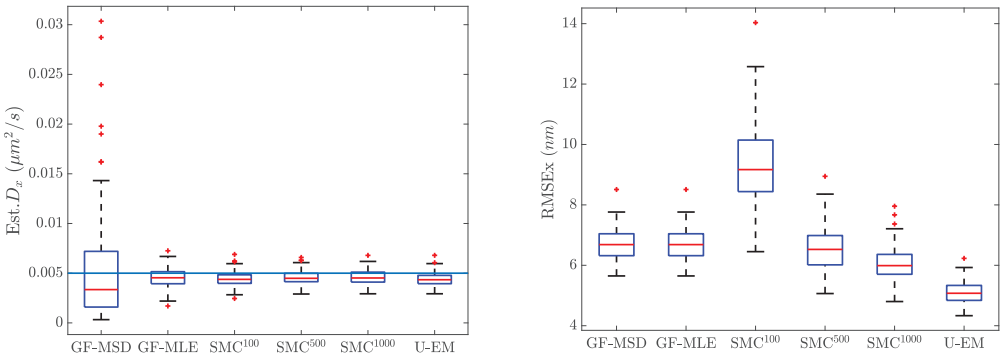


Figure 4·5: Box plots of estimated D_x and D_y by SMC-EM and U-EM. The true values of the diffusion coefficients are shown as solid horizontal lines in each plot. Note that the red line inside the box is the median, the edges of the box represent the first and third quartiles, the vertical dashed line indicates the bounds for data within 1.5 times the interquartile range, and the red + symbols are data points outside this range.

The comparison between the final results across the 100 trajectories for all the algorithms are shown in the box plots in Fig. 4·6 and recapitulated in Table 4.4. Results in y are similar and are omitted for space reasons. Note that there is a clear bias in the diffusion coefficient estimation in the GF-MLE and in our EM based methods. This is likely driven by a variety of factors, including the length of the data set (since MLE methods are only guaranteed to be consistent, meaning that they converge to the true value as the amount of data becomes large) and nonlinearities in

the models. A close examination of the SMC-EM results shows that bias reduces as we go from SMC-EM^{100} to SMC-EM^{500} and then to SMC-EM^{1000}. The SMC techniques more faithfully represent the nonlinear nature of the system as the number of MC samples. By contrast, the UKF at the heart of U-EM is accurate only to second-order. This supports the argument that the nonlinearities in the observation models are at least partially responsible for driving the bias.



(a) Estimation of D_x **(b)** RMSE$_x$ for localization in x.

Figure 4·6: Performance comparison among the different analysis methods. With $G = 100$, $N_{bgd} = 10$, $D_x = 0.005$ μm^2/s, and $D_y = 0.01$ μm^2/s. (a) Box plot results for the estimate of D_x. (b) RMSE for $x-$localization.

Table 4.4: Algorithm performance at $G = 100$, $N_{bgd} = 10$, $D_x = 0.005$ μm^2/s, $D_y = 0.01$ μm^2/s. Note that the estimates in table are in the form of *mean* $\pm$ *Std.* while the boxplots in Fig. 4·6 indicate the median.

Approach	Est.D_x (μm^2/s)	Est.D_y (μm^2/s)	RMSE$_x$(nm)	RMSE$_y$ (nm)
GF-MSD	0.0055 $\pm$ 0.0059	0.0102 $\pm$ 0.0096	6.7 $\pm$ 0.524	6.7 $\pm$ 0.506
GF-MLE	0.0046 $\pm$ 9.72e-4	0.0092 $\pm$ 0.0019	6.7 $\pm$ 0.524	6.7 $\pm$ 0.506
SMC-EM^{100}	0.0044 $\pm$ 7.76e-4	0.0092 $\pm$ 0.0015	9.2 $\pm$ 1.100	9.8 $\pm$ 1.300
SMC-EM^{500}	0.0046 $\pm$ 7.23e-4	0.0097 $\pm$ 0.0015	6.6 $\pm$ 0.757	6.5 $\pm$ 0.629
SMC-EM^{1000}	0.0046 $\pm$ 7.15e-4	0.0097 $\pm$ 0.0015	6.0 $\pm$ 0.580	5.9 $\pm$ 0.488
U-EM	0.0044 $\pm$ 7.0013e-4	0.0091 $\pm$ 0.0013	6.3 $\pm$ 0.475	7.7 $\pm$ 1.200

These results show that at these high signal levels, GF-MLE, SMC-EM, and U-EM all perform similarly in terms of diffusion coefficient estimation. GF-MSD, however,

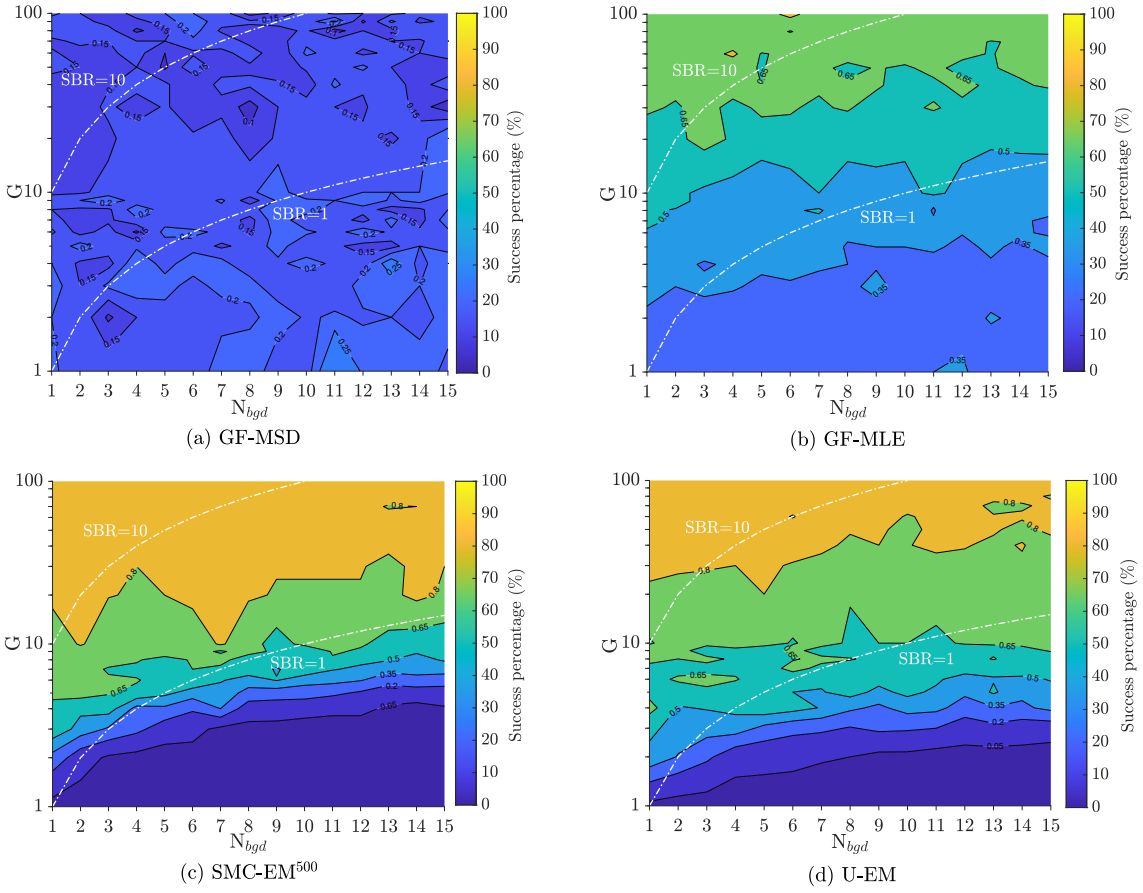
while having a similar mean, has a much larger variance and many more outliers than the others. Localization performance is evaluated in terms of the RMSE over an entire trajectory. Both GF and the EM-based schemes yield accurate localization with mean errors of below 7 nm for all but SMC^{100} where the small number of Monte Carlo samples used to represent the location distribution leads to both a larger error and a larger variance relative to the other schemes. Table 4.4 also shows that the performance improvement is minimal when the number of Monte Carlo samples increases from 500 to 1000. In the remainder of this work, then, we use 500 particles in SMC-EM.

4.2.2 Case 2: Performance at Different Signal-to-Background Ratios

In this second case, the diffusion coefficients were again fixed at $D_x = 0.005\ \mu\text{m}^2/\text{s}$, $D_y = 0.01\ \mu\text{m}^2/\text{s}$ and the imaging parameters set as in Table 4.3. The peak intensity, G, was varied across two decades, from $1 - 100$, and the background noise, N_{bgd}, was varied from 1 to 15. As before, 100 datasets of 100 images each were simulated at each pair of $\{G,\ N_{bgd}\}$ and the performance of the four algorithms, GF-MSD, GF-MLE, SMC^{500}, and U-EM compared.

To evaluate the parameter estimation performance among the different approaches, we followed the approach set out in (Michalet and Berglund, 2012) and defined a successful estimate as one which was within 25% of its true value. The success maps for each of the algorithms are shown in Fig. 4·7.

In these plots, color corresponds to the percentage of runs where successful estimation was achieved. Results for D_y were similar and are omitted for space reasons. These results show that GF-MSD has the worst performance of all four algorithms across all settings of intensity and background noise level with very low rates of success even at the highest SBR and signal levels considered. At the absolute lowest signal levels, GF-MLE shows the highest success rate (though that rate is still very



(a) GF-MSD (b) GF-MLE

(c) SMC-EM^{500} (d) U-EM

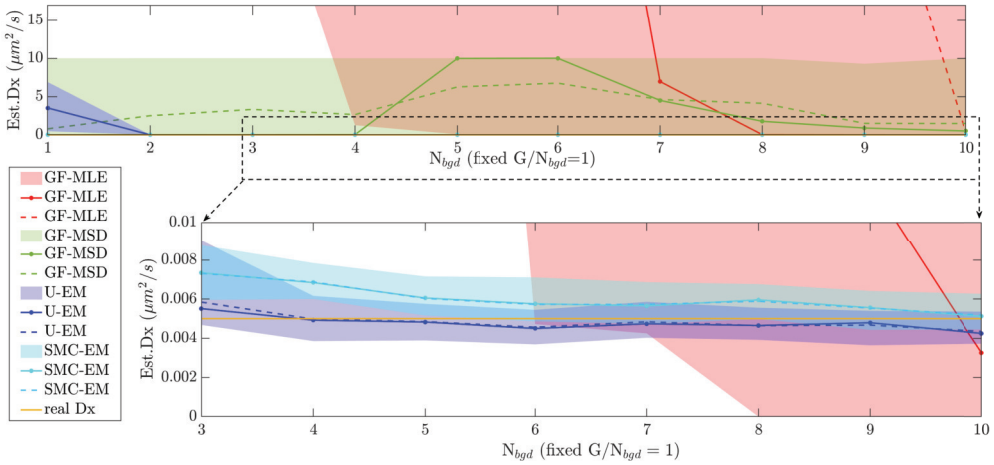
Figure 4·7: Success maps of the four algorithms. The percentage of trajectories resulting in an estimated D_x within 25% of the true value as a function of peak intensity G and background level N_{bgd} is shown. Yellow indicates 100% success while blue represents 0%. Results along the two white curves are shown in Fig. 4·8.

low). The two EM-based methods, however, show the highest level performance when the entire range of SBRs is considered.

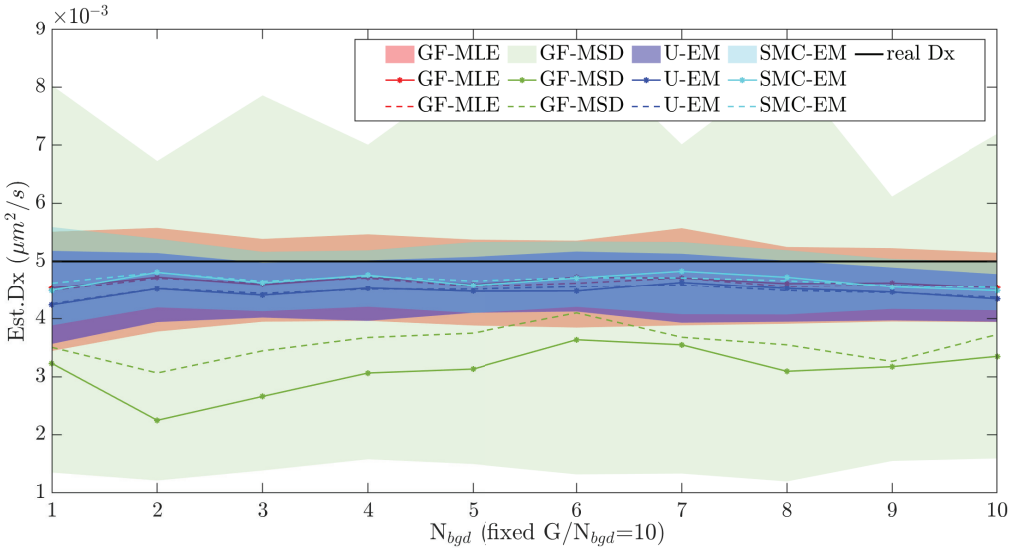
To dive more deeply into these results, we compared the performance in the accuracy of diffusion coefficient estimation at SBRs of $G/N_{bgd} = 10$ and $G/N_{bgd} = 1$ along the two white curves on the success maps in Fig. 4·7. The parameter estimation results are shown in Fig. 4·8 for both SBR = 1 (representative case at low signal intensities) and SBR = 10 (representative case at high signal intensities) . The figures show the mean and median for all algorithms as well as the middle two quantile (50%) range.

Note that at SBR = 1, the GF-MSD approach essentially fails while GF-MLE needs an intensity of $G = 10$ before its estimates are reasonable. By contrast, our EM based methods return reasonable results beginning at an intensity of $G = 3$. With an SBR of 10, all four algorithms yield reasonable results, though the GF-MSD algorithm has the worst performance with a median value that significantly underreports relative to the true value and with quantiles that are much larger than those of the other schemes. The other three algorithms all have similar performance, though the EM-based methods do yield tighter quantiles.

We also compared the localization accuracy of the different algorithms for the same data. The results for both SBR = 1 and SBR = 10 are shown in Fig. 4·9. As before, we show the center two quantiles, mean, and median of the estimates over the 100 trials at each value of G and N_{bgd}. (Note that since both the GF-MSD and GF-MLE algorithms use GF for localization, their results are combined as they are equivalent.) SMC-EM^{500} outperforms the other algorithms at all signal levels. Except at the very lowest signal level, U-EM outperforms GF. As the signal level increases, GF eventually catches up to match the results of SMC-EM and U-EM.



(a) SBR=1



(b) SBR=10

Figure 4·8: Diffusion coefficient estimation performance over 100 simulation runs at different signal and background levels. With fixed (a) SBR = 1 and (b) SBR = 10. Shown are the middle two quantiles (colored, shaded area), median (solid line), and mean (dashed line) using GF-MLE (red), GF-MSD(green), U-EM (purple), and SMC-EM^{500} (cyan). The true value was $D_x = 0.005$ μm^2/s.

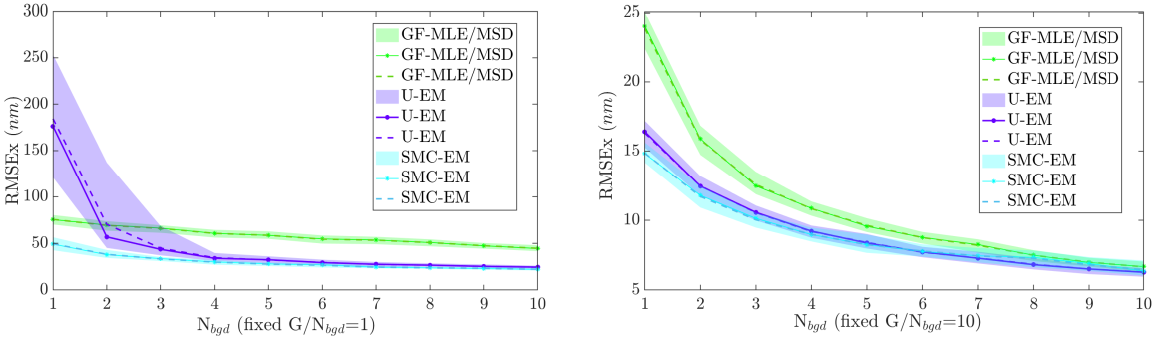


Figure 4·9: Localization performance (RMSE) over 100 simulation runs at different signal and background levels. With fixed (left) SBR = 1 and (right) SBR = 10. Shown are the middle two quantiles (colored, shaded area), median (solid line), and mean (dashed line). GF (green), U-EM (purple), and SMC-EM^{500} (cyan).

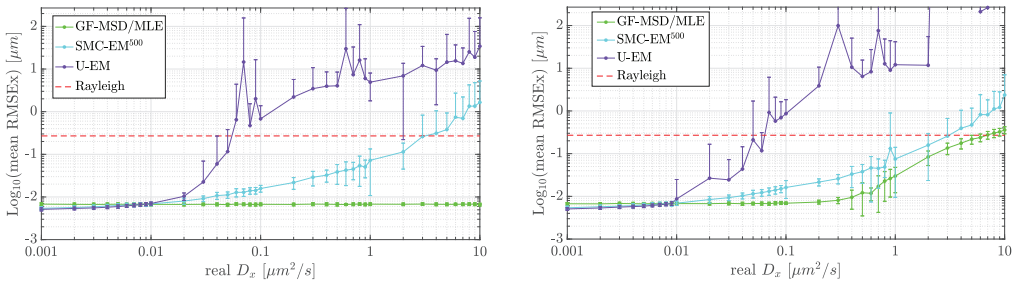
4.2.3 Case 3: Performance as a Function of Diffusion Coefficient

In the presence of motion blur, the performance of both localization and parameter estimation will depend on the diffusion coefficient. In addition, because the EM-based schemes jointly estimate the trajectory and the model parameters, it is reasonable to expect that performance will depend on motion model parameters even in the absence of motion blur (representing the limit of instantaneous image acquisition). To study this, we fixed the signal levels at $G = 100$, $N_{bgd} = 10$ (where all algorithms perform similarly) and ran simulations with $D_x = D_y$ over the range of 0.001 μm^2/s to 10 μm^2/s, considering both the case with motion blur (with $N_{sub} = 100$) and without (with $N_{sub} = 1$). We set a threshold for localization failure as the diffraction limited resolution given by the Rayleigh criterion. For the imaging parameters used here this leads to

$$\Delta L_{Rayleigh} = \frac{0.61\lambda}{NA} = 270\,\text{nm}. \tag{4.9}$$

The results for localization in the absence of motion blur are shown in Fig. 4·10a. As expected, if the measurements can be obtained instantaneously then the perfor-

mance of the GF method is independent of the diffusion coefficient since in each frame the particle is motionless. The EM-based schemes, however, do show degraded performance as the diffusion coefficient increases with the resulting thresholds shown in Table 4.5. It is perhaps somewhat surprising that U-EM and SMC-EM have such drastically different thresholds given that they use the same observation model. However, the problem in U-EM arises primarily from the the breakdown of the unscented transform at large noise variances. Thus, while U-EM is much better than SMC-EM in terms of computational complexity, it is limited to small values of the ratio of diffusion coefficient to sample rate due to the inability of the UKF to handle large variances in the noise inputs.



(a) Without motion blur **(b)** With motion blur

Figure 4·10: Localization performance in terms of RMSE across varying diffusing speeds. (a) without and (b) with motion blur using GF-MSD/MLE (green), SMC-EM^{500} (cyan), and U-EM (purple). The failure threshold is defined as the Rayleigh resolution criterion (red, dashed).

The case with motion blur is shown in Fig. 4·10(b). U-EM and SMC-EM perform similarly to the setting without motion blur. Now, however, estimation based on GF also shows a limit on the diffusion coefficient beyond which localization fails, likely driven by the fact that motion blur causes the PSF to diverge from a simple Gaussian shape. The resulting limits are shown in Table 4.5. It is important to note that when using GF, we take advantage of the prior information available in the segmentation and limit the estimate to be within the segmented image.

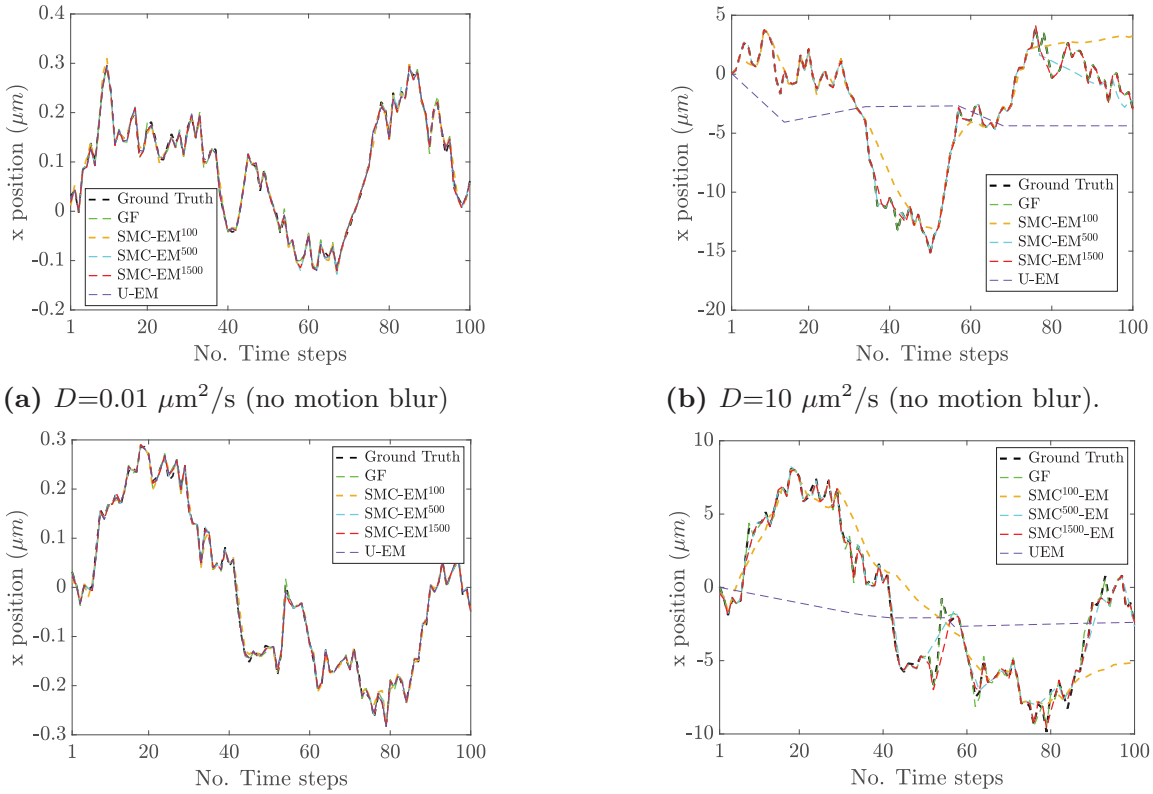
Table 4.5: Diffusion coefficient threshold before localization failure with $G = 100$, $N_{bgd} = 10$.

Condition	GF (μm^2/s)	U-EM (μm^2/s)	SMC-EM^{500} (μm^2/s)
No motion blur	∞	0.05	3.0
With motion blur	6.0	0.05	3.0

The values of the thresholds for the diffusion coefficient depend, of course, on the specific imaging parameters. In general, in the absence of motion blur, increasing the peak intensity G or the shutter time δt will increase the SBR and thus improve localization performance and one would expect the SMC-EM methods to work at higher diffusion coefficient values. Since U-EM depends on the unscented transform, increasing the imaging rate (that is, decreasing Δt) will reduce the process noise and thus increase the diffusion coefficient threshold. In the presence of motion blur, decreasing the shutter time will mitigate its effects but at the cost of reducing the number of acquired photons and thus the SBR. This can be compensated for somewhat by increasing the intensity parameter G by increasing the power of the excitation, though the ability to do so is limited by phototoxicity issues.

To better understand the degradation in localization as the diffusion coefficient increases, we show in Fig. 4·11 typical runs both with and without motion blur. These results show that at larger diffusion coefficients, the U-EM scheme simply fails while the others degrade more smoothly, particularly in the absence of motion blur.

For SMC-EM, we show results using different numbers of sampled particles in the PF methods. With a small number of samples (e.g., SMC-EM^{100}) and at large D, the SMC-EM tends to track the particle well in most frames but occasionally to lose that track. Because segmentation ensures that the data is never too far from ground truth, the algorithm is often able to pick up the location again. Increasing the number of sampling particles in the SMC-EM mitigates this effect at any given value of D and thus the threshold on the diffusion coefficient increases with increasing number



(a) D=0.01 μm^2/s (no motion blur)

(b) D=10 μm^2/s (no motion blur).

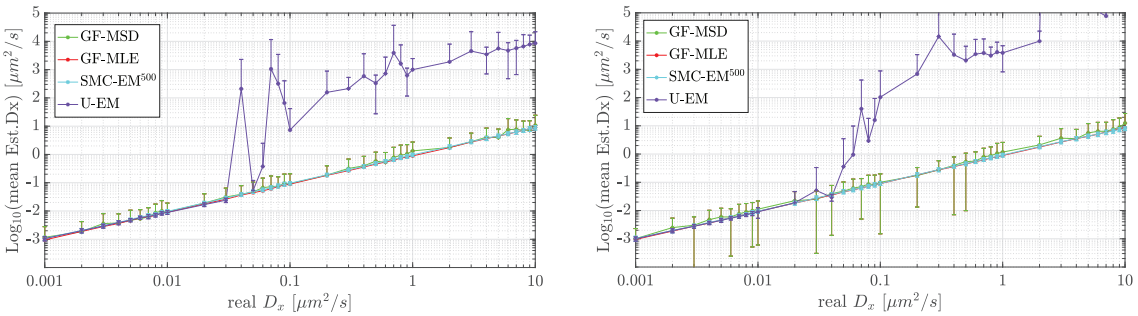
(c) D=0.01 μm^2/s (with motion blur).

(d) D=10 μm^2/s (with motion blur).

Figure 4·11: Typical localization performance of GF, SMC-EM, and U-EM. (green) GF, (orange) SMC-EM with 100 Monte Carlo samples, (cyan) 500 Monte Carlo samples, (red) 1500 Monte Carlo samples, and (purple) U-EM at a diffusion coefficient of (a,c) 0.01 μm^2/s and (b,d) 10 μm^2/s, both (a,b) without and (c,d) with motion blur.

of particles in the filter. To further demonstrate this, we also show typical results when using 1500 sampling particles; SMC-EM is then able to track the particle even at 10 μm^2/s.

The results for the estimation of D_x as a function of the diffusion coefficient are shown in Fig. 4·12, both with and without motion blur. As noted before, the UKF element of the U-EM algorithm fails as the covariance of the process noise, defined by the value of the diffusion coefficient, gets large. As seen in the localization performance results in Fig. 4·11, this leads to complete loss of tracking which in turn leads to failed diffusion coefficient estimation. By contrast, SMC-EM, GF-MSD, and GF-MLE continue to produce good estimates throughout the entire considered range (though, of course, GF-MSD has much larger variance than the other approaches. It is perhaps surprising that SMC-EM produces good diffusion coefficients at large D even though, as seen in Fig. 4·10, localization performance degrades significantly. The likely reason is that, as illustrated in Fig. 4·11, for a large part of any given trajectory, tracking is good with only a few large outliers. These outliers have a serious impact on the RMSE but a smaller effect on the diffusion coefficient.



(a) Without motion blur **(b)** With motion blur

Figure 4·12: Mean estimates of D_x by GF-MSD, GF-MLE, SMC-EM, and U-EM as a function of the true diffusion coefficient. (a) without and (b) with motion blur.

In two dimensional diffusion cases, we described two versions of our EM-based

framework for simultaneous localization and parameter estimation from SPT data. Our algorithms indicate that, at least in the two-dimensional setting considered, if there are enough photons and a good SBR, then GF-MLE (or $\text{MLE}_{\text{sCMOS}}$), SMC-EM, and U-EM perform similarly well and all outperform GF-MSD. Given the additional computational complexity of the EM-based methods over GF-MLE/$\text{MLE}_{\text{sCMOS}}$, it makes more sense to apply these more standard algorithms in this setting. At low signal levels, however, the EM-based methods outperform the others. The choice between the different EM schemes is dictated in large part by the computation time but also by the ratio of the (expected) diffusion coefficient and the sampling rate. If this ratio is low, U-EM offers good results at significantly less computation time than the SMC-EM methods. These conclusions are summarized in Fig. 4·13.

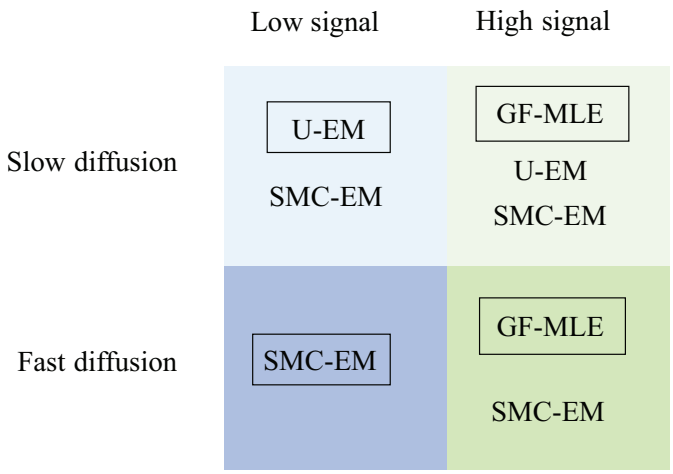


Figure 4·13: Qualitative guidance for choice of SPT localization and parameter estimation algorithm. Shown are the algorithms that produce similar results in each of the domain with the method. The boxed algorithm in each quadrant has the lowest computational load.

4.3 Application of EM based Methods on sCMOS Data

In this section we incorporate the sCMOS model described in Chapter 2.2.2 into our scheme. Simulations were performed using the settings in Table 4.3 and at two different signal levels, one low ($G = 10$) and one high ($G = 100$). A typical image frame at $G = 100$ and $N_{bgd} = 10$, together with the pixel-by-pixel variance and gain maps for this frame, is shown in Fig. 4·14.

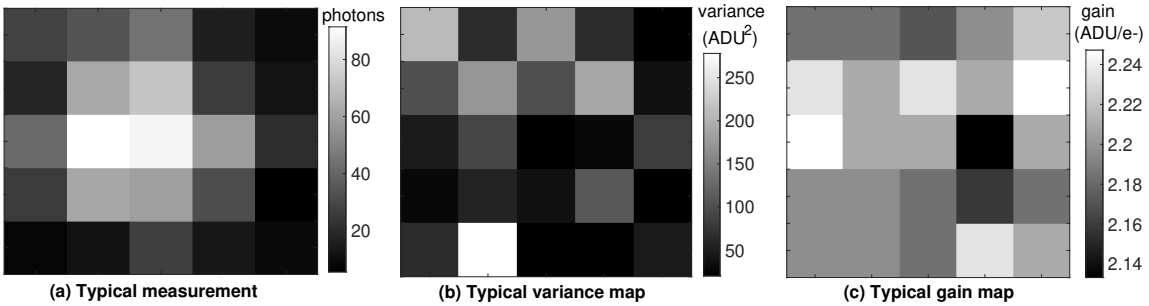


Figure 4·14: Typical frames related to sCMOS camera model. (a) Observation with $N_{bgd} = 10$, $G = 100$. (b) Variance and (c) Gain maps of the pixels in the frame shown in (a).

The work in (Huang et al., 2013) showed that GF can yield poor results on sCMOS data and, motivated by this, developed a localization algorithm specific to the sCMOS model using ML estimation. Following (Huang et al., 2013), the probability distribution function for a pixel value with both shot noise and read-out noise can be approximated by

$$\begin{aligned} P_{\text{sCMOS}}(z = I_{p,t} + \sigma^2_{p,t} | \lambda_{p,t}(\theta_{xy,t}), N_{bgd}, g_{p,t}, \text{Var}_{p,t}) \\ = \frac{e^{-(\lambda_{p,t}+N_{bgd}+\sigma^2_{p,t})}(\lambda_{p,t} + N_{bgd} + \sigma^2_{p,t})^z}{\Gamma(z+1)}, \end{aligned} \tag{4.10}$$

where $\Gamma(\cdot)$ is the standard Gamma function. The $\text{MLE}_{\text{sCMOS}}$ for localization at time

t can be expressed as

$$\hat{\theta}_{xy,t} = \underset{\theta_{xy,t}}{\arg\min} \left\{ -\ln \left[\prod_{p=1}^{25} P_{\text{sCMOS}}(z = I_{p,t} + \sigma_{p,t}^2 | \lambda_{p,t}(\theta_{xy,t}), N_{bgd}, g_{p,t}, \text{Var}_{p,t}) \right] \right\}, \tag{4.11}$$

where $\hat{\theta}_{xy,t}$ is the maximum log-likelihood estimation of localization at time t.

Using the form of the PSF for $\lambda_{p,t}$ given in the main paper, this MLE can be expressed as

$$\hat{\theta}_{xy,t} = \underset{\theta_{xy,t}}{\arg\min} \sum_{p=1}^{25} \left[(\lambda_{p,t} + N_{bgd} + \sigma_{p,t}^2) - z \cdot \ln(\lambda_{p,t} + N_{bgd} + \sigma_{p,t}^2) + \ln \Gamma(z+1) \right], \tag{4.12}$$

where $z = I_{p,t} + \sigma_{p,t}^2$. In the remainder of this subsection, then, we use that approach to localize the particle in each frame. We combine those localization results with the MLE approach to parameter estimation from (Berglund, 2010) and refer to this combined algorithm as $\text{MLE}_{\text{sCMOS}^+}$.

Combining the photon detection process of the microscope with the read-out noise of the detector leads to a nonlinear, non-Gaussian measurement model. While this can be handled directly using SMC methods, the UKF requires Gaussian distributed noise. Therefore when using U-EM scheme, we need to transform the model in (2.13) into a form amenable to the UKF. There are different approaches for variance stabilization, such as the Anscombe (Anscombe, 1948) or the Freeman and Tukey (Freeman and Tukey, 1950) transformations, or direct approximation by a Gaussian model (when measured intensities are sufficiently large) (Gnedenko, 2017). These different approaches have been discussed and compared in (Lin and Andersson, 2019) where it was found that in general the Anscombe transform performs the best in the SPT setting. Therefore, we use the generalized Anscombe transformation (Makitalo and

Foi, 2012) to transform the observation model (2.13) into

$$\tilde{I}_{p,t} = 2\sqrt{\lambda_{p,t} + N_{bgd} + \frac{3}{8} + \sigma_{p,t}^2} + v_t, v_t \sim \mathcal{N}(0,1). \tag{4.13}$$

To apply this transformation, the observed measurements $I_{p,t}$ should be first expressed as

$$\hat{I}_{p,t} = 2\sqrt{I_{p,t} + \frac{3}{8} + \sigma_{p,t}^2}. \tag{4.14}$$

We then take the transformed observation as the input for the U-EM method described in Chapter 3.2.1.

4.3.1 Diffusion with Brownian Motion

First, we focus on two dimensional Brownian motion (2.2). The comparison between the final results across all 100 simulation runs among all the algorithms at the high signal level are listed in Table 4.6, and they are summarized in box plots shown as Fig. 4·15.

Table 4.6: Algorithm performance at SBR = 10 on sCMOS data.

Approach	Est.D_x (μm^2/s)	Est.D_y (μm^2/s)	RMSE$_x$(nm)	RMSE$_y$ (nm)
MLE$_{\text{sCMOS+}}$	0.00472 ± 0.00105	0.00922 ± 0.00199	6.72 ± 0.513	6.82 ± 0.514
SMC-EM^{100}	0.00450 ± 0.00084	0.00947 ± 0.00142	9.97 ± 1.338	11.19 ± 1.857
SMC-EM^{500}	0.00470 ± 0.00079	0.00964 ± 0.00144	7.35 ± 0.711	7.65 ± 0.756
SMC-EM^{1000}	0.00473 ± 0.00078	0.00968 ± 0.00145	6.90 ± 0.620	7.04 ± 0.560
U-EM	0.00448 ± 0.00074	0.00894 ± 0.00135	7.36 ± 0.547	8.99 ± 1.22

These results indicate that when the signal level is high, all methods perform well in parameter estimation and localization, though when using only 100 particles in SMC-EM the RMSE is higher than with the other methods. In addition, SMC-EM^{500} and SMC-EM^{1000} show fewer outliers than the other methods.

The comparison between the final results across all 100 simulation runs for all the algorithms for the low signal level are shown in the box plots in Fig. 4·16 and

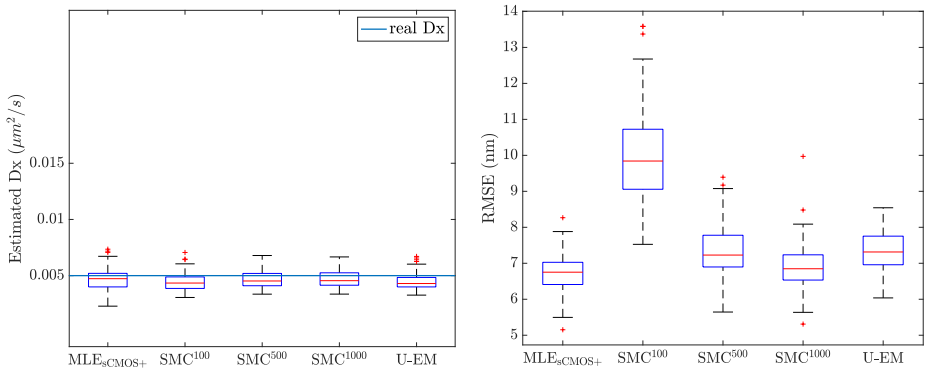


Figure 4·15: Estimation performance of different SPT methods on sCMOS camera model at $G = 100$ and $N_{bgd} = 10$.

recapitulated in Table 4.7. The EM-based schemes show significant improvement over $\text{MLE}_{\text{sCMOS}}$ in this setting in terms of both reduced variance in the parameter estimates and smaller RMSE in localization.

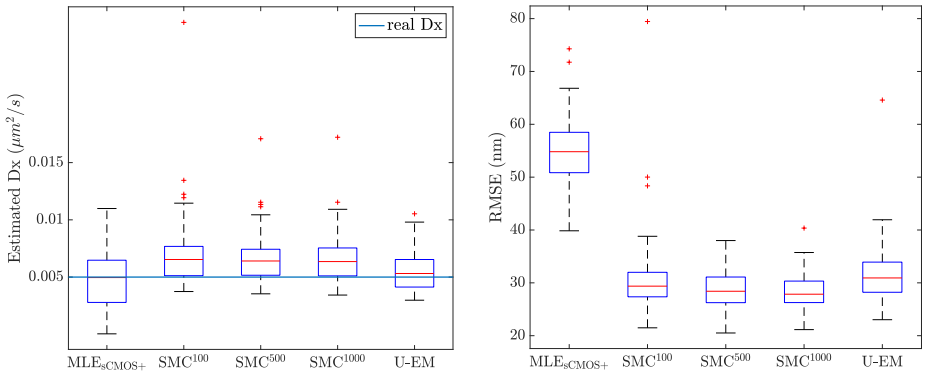


Figure 4·16: Estimation performance of different SPT methods on sCMOS camera model at $G = 10$, $N_{bgd} = 10$.

To further highlight the localization performance differences between the algorithms, in Fig. 4·17 we show the results from a typical run. While all methods track the true trajectory, $\text{MLE}_{\text{sCMOS}}$ produces more outliers while the EM-based methods stay closer to the trajectory throughout the run.

Table 4.7: Algorithm performance at SBR = 1 on sCMOS data.

Approach	Est.D_x (μm^2/s)	Est.D_y (μm^2/s)	RMSE$_x$(nm)	RMSE$_y$ (nm)
MLE$_{\text{sCMOS+}}$	0.00484 ± 0.00247	0.00999 ± 0.00391	54.59 ± 6.18	54.76 ± 6.45
MLE$^{*}_{\text{sCMOS+}}$	0.00484 ± 0.00247	0.00999 ± 0.00391	54.22 ± 5.65	54.55 ± 6.12
SMC-EM^{100}	0.00699 ± 0.00287	0.0114 ± 0.00312	30.77 ± 6.65	36.50 ± 9.65
SMC-EM^{100,*}	0.00661 ± 0.00177	0.0112 ± 0.00286	29.89 ± 3.61	35.23 ± 4.30
SMC-EM^{500}	0.00668 ± 0.00206	0.0110 ± 0.00311	28.71 ± 3.47	33.92 ± 4.49
SMC-EM^{500,*}	0.00642 ± 0.00159	0.0109 ± 0.00294	28.71 ± 3.47	33.33 ± 3.31
SMC-EM^{1000}	0.00662 ± 0.00207	0.0109 ± 0.00307	28.46 ± 3.46	33.44 ± 4.04
SMC-EM^{1000,*}	0.00646 ± 0.00171	0.0108 ± 0.00291	28.34 ± 3.26	33.02 ± 3.26
U-EM	0.00556 ± 0.00165	0.00931 ± 0.00293	31.58 ± 5.26	44.00 ± 17.24
U-EM*	0.00551 ± 0.00158	0.00891 ± 0.00239	31.25 ± 4.08	38.56 ± 6.52

$*$ excluding outliers.

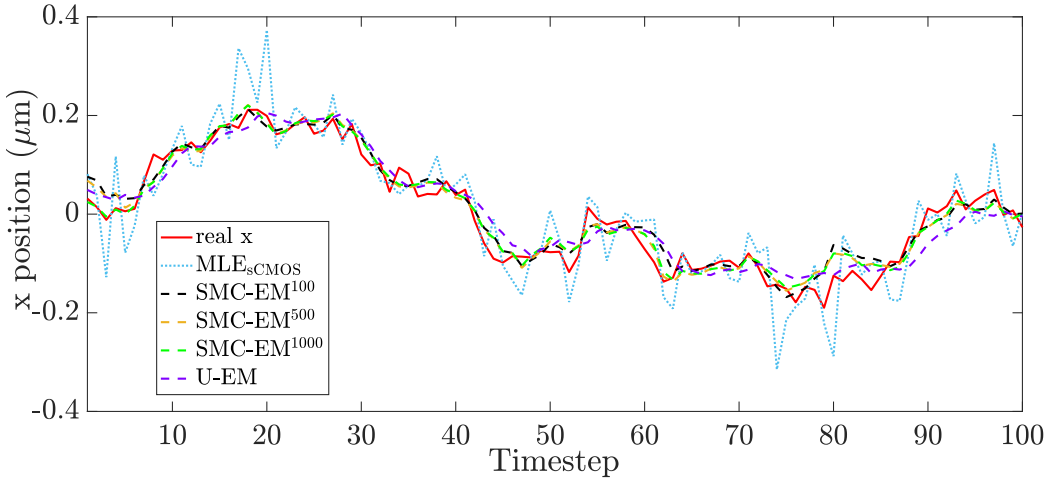


Figure 4·17: Typical localization results at $G = N_{bgd} = 10$.

4.3.2 Estimation with Ornstein Uhlenbeck flow

In this subsection, we combine the sCMOS observation model with an Ornstein-Uhlenbeck motion model andstudy the relative performance of SMC-EM and U-EM under this scenario. In addition, since the quality of the final estimates depends both on the chosen algorithm and the amount of available data, we also consider the impact of the number of camera frames available for analysis.

The investigation work start with data simulations with Table 4.8 listing the parameter settings particular to the O-U process simulation. When using U-EM, the tuning parameters for the UKF were set to $(\alpha,\ \kappa,\ \beta) = (1,\ 0,\ 2)$.

Table 4.8: Parameter settings for simulating O-U process

Symbol	Parameter	Values
$D_{x,y}$	Diffusion coefficient	0.01 μm^2/s
A	Stiffness coefficient	$1.0s^{-1}$
δt	Shutter period	10 ms
G	Peak intensity gain (signal)	100
N_{bgd}	Background noise	10

First, we focus on a typical case with a fixed image length of $N = 100$. We simulate 100 sample paths and corresponding image sequences and analyze them using SMC-EM and U-EM. 10 EM iterations were run under each EM based method. Fig. 4·18 showing the evolution of the parameter estimates as a function of the EM iteration.

The overall mean and standard deviation of the estimated parameters are summarized in Table 4.9. These results indicate that both U-EM and SMC-EM have good performance. As the number of particles used in SMC-EM grows, the RMSE, parameter estimation variance, and parameter estimation bias all decrease. However, this comes at a cost in computation time.

Correspondingly, the overall mean and standard deviation of the estimated posi-

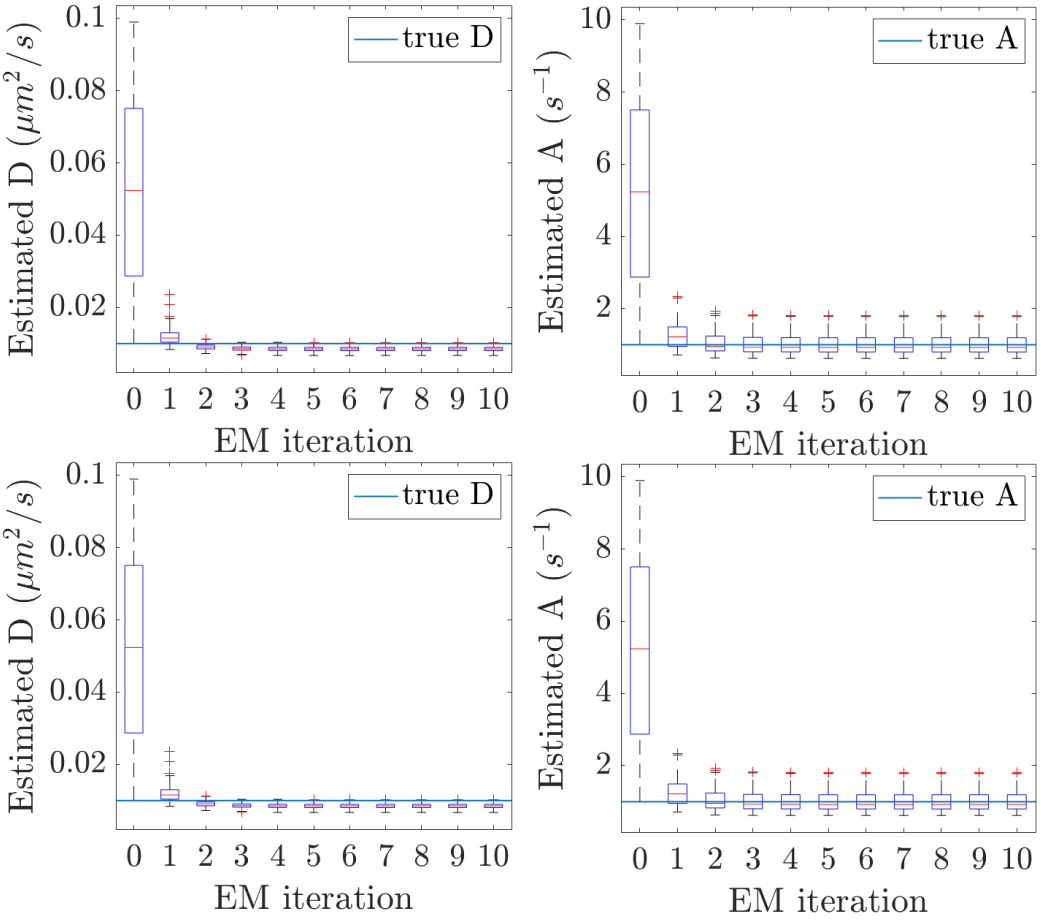


Figure 4·18: Boxplot of estimates as a function of EM iteration for (top) U-EM and (bottom) SMC-EM^{100}.

Table 4.9: Parameter estimation with 100 images

Method	D ($\mu m^2/s$)	A (s^{-1})
U-EM	0.008514 ± 0.00072991	1.01 ± 0.28134
SMC-EM^{50}	0.0080737 ± 0.00096272	0.99224 ± 0.2702
SMC-EM^{100}	0.0086592 ± 0.00087748	1.0164 ± 0.28005
SMC-EM^{500}	0.0092505 ± 0.00091714	1.0466 ± 0.29636

tion are summarized in Table 4.10 where performance is determined using the RMSE between the true particle position and the mean of the smoothed distribution $p(x_t|Y_N)$ across an entire trajectory. In the table, SMC-EM^M denotes an SMC-EM scheme using M sampled particles.

As expected, these results show that the performance of SMC-EM depends strongly on the number of particles used. All schemes, however, show very good performance with a resolution far below the diffraction limit. These same results are shown as boxplots in Fig. 4·19.

Table 4.10: Localization performance with 100 images

Method	RMSEx (nm)	RMSEy (nm)
U-EM	8.6558 ± 0.9979	8.9193 ± 1.1069
SMC-EM^{50}	13.6751 ± 2.0489	13.4994 ± 1.8195
SMC-EM^{100}	10.6292 ± 1.3278	10.7201 ± 1.3692
SMC-EM^{500}	7.5029 ± 0.7495	7.6169 ± 0.7535

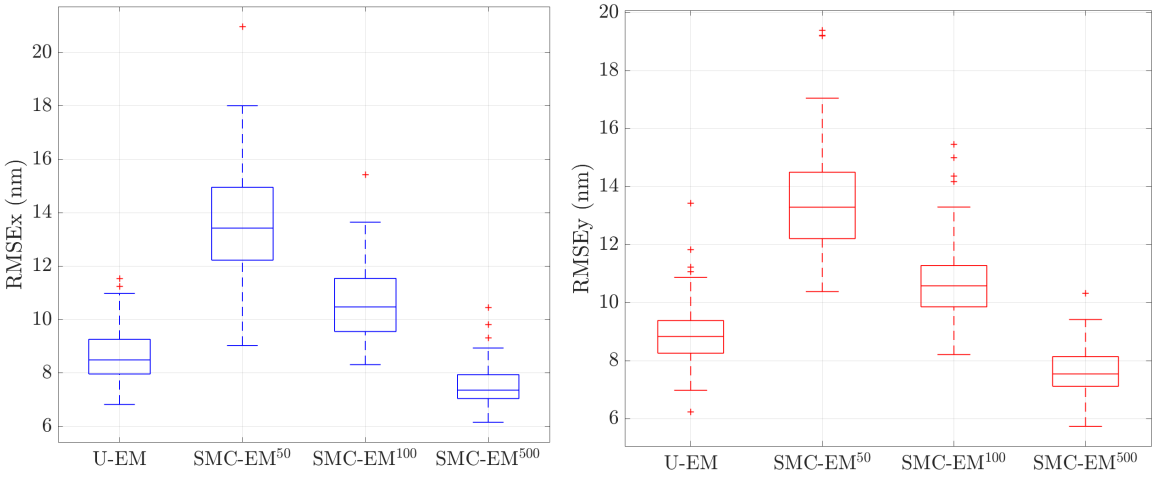


Figure 4·19: Boxplot of RMSE by U-EM and SMC-EM of (left, blue) x and (right, red) y.

A typical example of a trajectory estimation result by SMC-EM and U-EM shown in Fig. 4·20. For space reasons, only results in x are shown; results in y are similar.

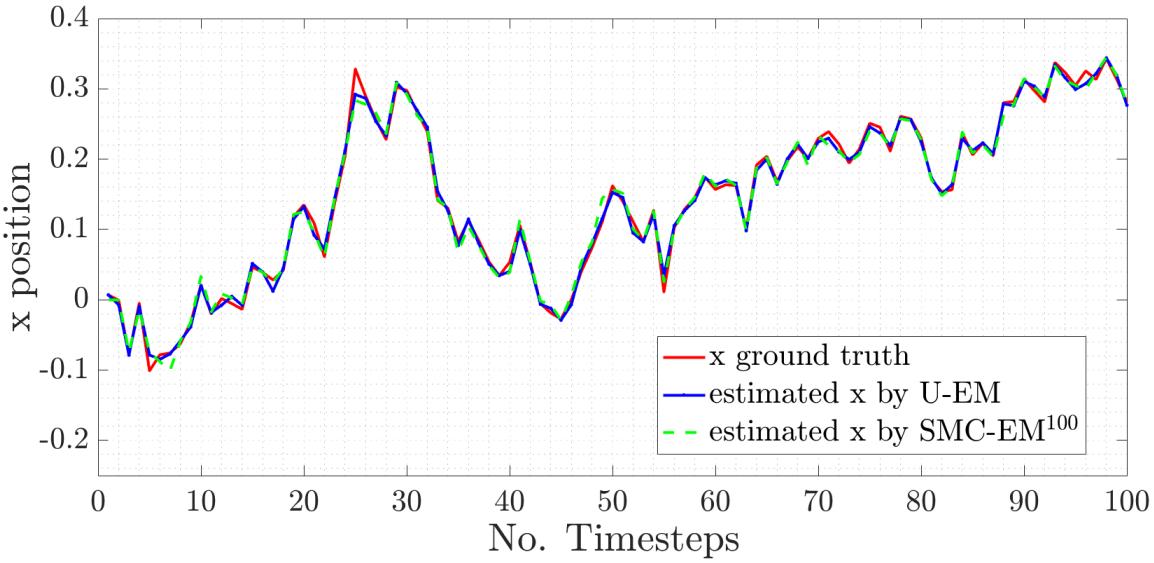


Figure 4·20: Typical trajectory estimation result.

As part of this work, we explore the bottlenecks in computation and find that the main limitation comes from the calculation of the double integrals in the observation model (2.19). Therefore, we replace the direct execution with a table lookup approach which guarantees an error $< 10^{-3}$ for computing $\lambda_{p,t}$. To improve the SMC performance, we take advantage of parallel processing in Matlab. The calculations were carried out on a 2.3 GHz Intel Core i5 running Mac OS 10.14.4. Fig. 4·21 shows the corresponding improvement in the runtime of the two SPT methods. Note that it takes only two seconds for U-EM to complete an analysis run with 100 images.

In the SPT setting, the number of image frames available depends on a variety of factors including the frame rate, the intensity of the excitation, and the type of fluorescent label used. Data sets can range from the 10's to 1000's of frames. In this work, we explored different data lengths, from $N = 10$ to 1000 on log spacing. For each N, 100 datasets were simulated with parameter settings as Table 4.8. In this work, 10 EM iterations are enough for estimation convergence. The parameter estimation and localization performance by U-EM, SMC-EM^{100}, and SMC-EM^{500} are

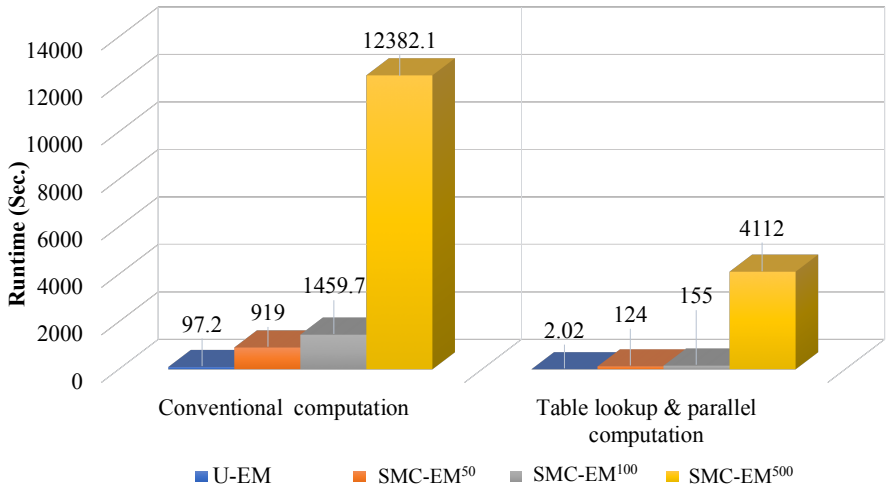


Figure 4·21: Runtime of different EM-based methods on single dataset at $N_{bgd} = 10, G = 100$ with image length of 100.

shown in Fig. 4·22.

Due to space limitations, only the results of RMSEx are presented here; results of RMSEy are similar. As expected, as the number of images increases, the final estimates have lower variance and bias for the parameters and lower RMSE. For SMC-EM, more images mainly contributes to a reduced variance, while the larger number of sampled particles mainly contributes to a closer median estimate.

In this subsection, we described the application of two EM-based methods, SMC-EM and U-EM, to SPT data analysis, focusing on Ornstein-Uhlenbeck motion and sCMOS cameras. Our results indicate that U-EM has a significant advantage over SMC-EM in terms of computation time but that, with increasing number of particles in the Monte Carlo methods, SMC-EM provides more accurate estimation. We also explored the impact of data length on estimation performance with results showing that increasing the amount of data reduces both bias and variance.

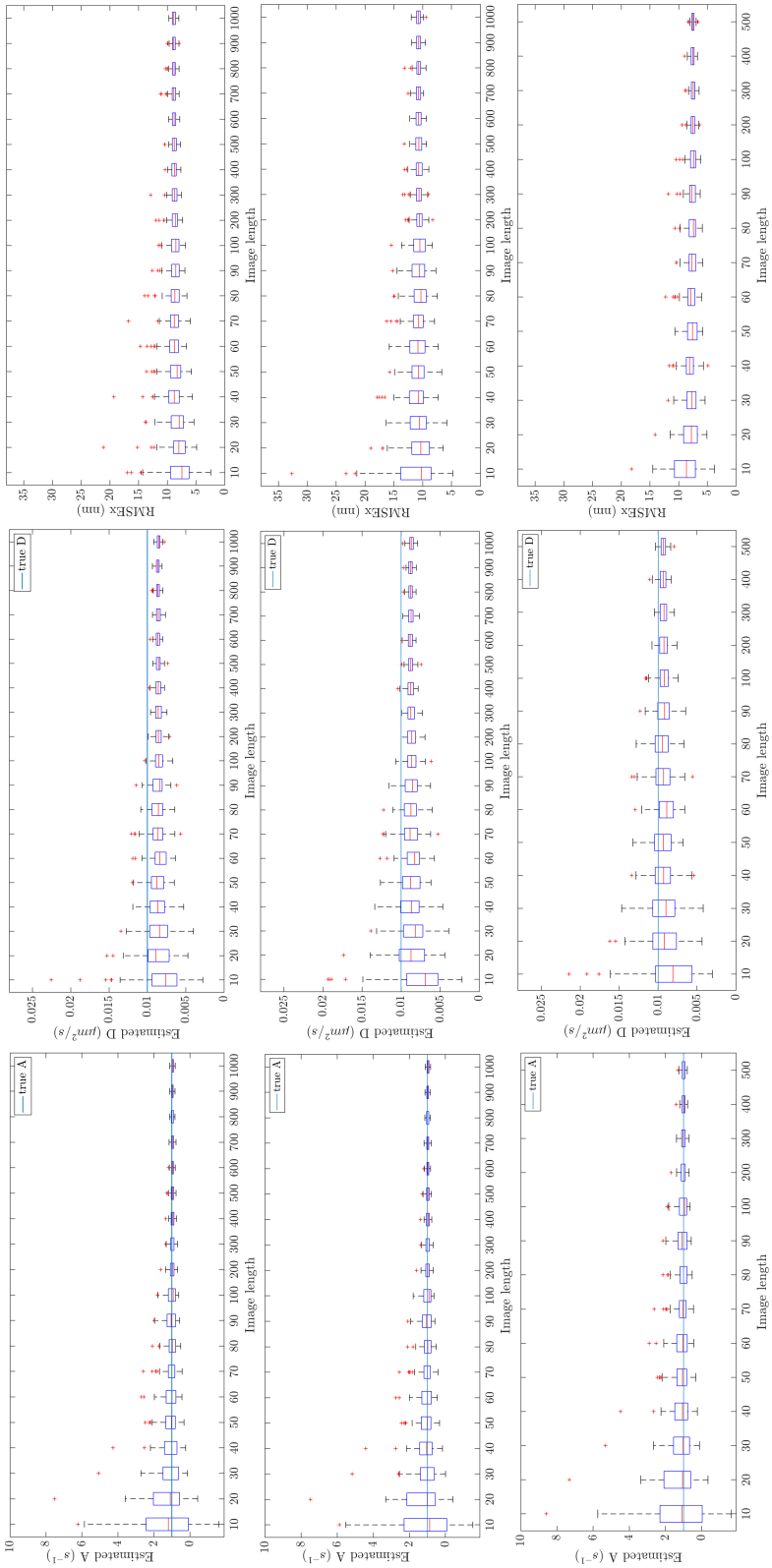


Figure 4·22: Boxplot of final estimation by (top row) U-EM, (middle row) SMC-EM^{100}, and (bottom row) SMC-EM^{500}.

4.4 Time-varying Diffusion

Finally, in this section we demonstrate how to apply U-EM to time-varying motion models, developing an estimation algorithm based on local likelihood estimation. We once again focus on the diffusion motion model with nonlinear observation model. The idea of local likelihood is very old (for some historical background see e.g. (Loader, 1999)). Local likelihood is a natural development of the sliding window approach (Loader, 1999). However, a thorough theoretical understanding of the method was not developed until the 1990s in the statistics literature under the name *local least squares* or *local polynomial modelling* (Loader, 1999; Fan and Gijbels, 1996). Despite these developments, the theory has not diffused widely outside the statistics literature and so a number of basic insights are still ignored in other application domains.

There are, of course, alternatives to a local modelling approach. For example, there is significant literature on how to model data using a global polynomial fit (kernel approach). However, this method potentially needs a large number of components to yield a reasonable low bias (Fan and Gijbels, 1996), and, as a consequence, may introduce over-parametrization, which has an effect on the variance of the estimate. The local approximation approach, in general, only requires a small number of parameters.

4.4.1 Time-varying Likelihood

The idea in our modification to allow for time-varying parameters is to pose a time-varying likelihood which is time-invariant in a window of a nominated length. The likelihood is then optimized in this window, and the process is carried out by leaving one sample out and taking a new one in order to keep the same window size. The process finishes when we have included the last available sample.

The local likelihood is defined as:

$$l_t(\theta_t) = \sum_{k=1}^{N} K_{k,t} l(y_k|\theta_t), \tag{4.15}$$

where $K_{k,t} := K(\frac{k-t}{h})$ is usually called *kernel*, h is the window length[1], t is a point within the window, usually the middle point, and k indicates any point within the window. Different values for h can be used, as well as different kernels (or weights), and the most appropriate selection of them is a nontrivial issue, see e.g. (Fan and Gijbels, 1996). One common window is the so called Epanechnikov, see e.g (Fan and Gijbels, 1996), which is used throughout this work.

Due to the nonlinear measurements of SPT, obtaining the likelihood function is an intractable problem, and thus we use the auxiliary $\mathcal{Q}$ function of the EM algorithm as an approximation of the likelihood, extending it to the time-varying setting through the local approach. We then consider the following equation:

$$\mathcal{Q}_t(\theta_t, \hat{\theta}_t^{(i)}) = \sum_{k=1}^{N} K_{k,t} \mathbb{E}\{p_{\theta_t}(x_k, y_k)|Y_N, \hat{\theta}_t^{(i)}\}. \tag{4.16}$$

From (3.8), we can write

$$\mathcal{Q}_t(\theta_t, \hat{\theta}_t^{(i)}) = I_1(\theta_t, \hat{\theta}_t^{(i)}) + I_2(\theta_t, \hat{\theta}_t^{(i)}) + I_3(\theta_t, \hat{\theta}_t^{(i)}), \tag{4.17}$$

where now each I_i, $i = 1, 2, 3$ is a function of the time-varying parameter θ_t. As noted above, in the SPT setting, the parameter θ_t only appears in I_2 hence, we can write the time-varying E-step as

$$\mathcal{Q}_t(\theta_t, \hat{\theta}_t^{(i)}) = \sum_{k=1}^{N} \mathbb{E}\{K_{t,k} \log p(x_k|x_{k-1})|Y_N, \hat{\theta}_t^{(i)}\}. \tag{4.18}$$

To optimize this time-varying function, we use a similar procedure as in the time-

[1] or *bandwidth* in the statistics literature

invariant case, that is, we set the derivative of Q_t with respect to θ_t to zero and solve. The resulting estimates optimize the (windowed) likelihood within window h. The estimation algorithm continues when the time points are shifted by one unit and finishes when the last data point is included in the window. Of course, the shifting can be done by more than one unit to reduce computation, see e.g. (Fan and Gijbels, 1996).

Notice that the difference between $\mathcal{Q}_t$ and $\mathcal{Q}$ (found in the time-invariant EM) is the inclusion of $K(v)$, and the time dependancy of the parameters of interest. We add the sub-index t in this function to denote that it depends on the time-varying parameter θ_t.

To summarize, the local likelihood approach introduces a time-varying E-step, denoted by $\mathcal{Q}_t$, that will be optimized within a window h. This, in turn, will optimize the log-likelihood function within the same window, to create the local likelihood approach. For our SPT problem, and discarding the constant terms and the parameters for the initial conditions, we can obtain the following auxiliary function

$$\begin{aligned}\mathcal{Q}_t(\theta_t, \hat{\theta}_t^{(i)}) = &- \log|Q_t^{-1}| \sum_{k=1}^{N} K_{k,t} + \\ &\operatorname{tr}\{Q_t^{-1}[\tilde{S}_{11} - 2\tilde{S}_{01}^T + \tilde{S}_{00}]\}\end{aligned} \tag{4.19}$$

with

$$\tilde{S}_{11} = \sum_{k=1}^{N} K_{k,t}[\hat{x}_{k|h}\hat{x}_{k|h}^T + P_{k|h}], \tag{4.20}$$

$$\tilde{S}_{01} = \sum_{k=1}^{N} K_{k,t}[\hat{x}_{k|h}\hat{x}_{k-1|h}^T + P_{k,k-1|h}]^T, \tag{4.21}$$

$$\tilde{S}_{00} = \sum_{k=1}^{N} K_{k,t}[\hat{x}_{k-1|h}\hat{x}_{k-1|h}^T + P_{k-1|h}]. \tag{4.22}$$

The values for $\hat{x}_{k|h}$, $P_{k|h}$, and $P_{k,k-1|h}$ are the smoothed state, variance, and covariance,

which are found using the UKF and URTSS, for details see (Lin and Andersson, 2019).

The M-step is completed by taking derivative of $\mathcal{Q}_t$ with respect to Q_t^{-1} and setting it to zero, obtaining as a result:

$$\hat{Q}_t^{(i+1)} = \frac{1}{\sum_{k=1}^{h} K_{k,t}}[\tilde{S}_{11} - 2\tilde{S}_{01}^T + \tilde{S}_{00}]. \tag{4.23}$$

For the purpose of showing the benefits of our time-varying approach, we use the naïve version of EM. However, we note that there is a robust version for the M-step, see e.g. (Gibson and Ninness, 2005), based on the Cholesky factorization, which improves the robustness of the implementation.

It is well-known that the only requirement needed in the EM algorithm to converge to a stationary point of the likelihood function (not necessarily the global maximum) is that $\mathcal{Q}(\hat{\theta}^{(i+1)}, \hat{\theta}^{(i)}) \geq \mathcal{Q}(\hat{\theta}^{(i)}, \hat{\theta}^{(i)})$. In fact, this is assured to occur when another auxiliary function found in the EM algorithm, known as $\mathcal{H}(\theta, \hat{\theta}^{(i)})$, is proven to be non-increasing; for details see e.g. (McLachlan and Krishnan, 2008).

Since our algorithm is based on EM, it inherits these same convergence properties. Of course, this also implies we cannot guarantee convergence to the true value because the EM algorithm itself only yields the global maximum under certain special cases (McLachlan and Krishnan, 2008, ch.3). In our scheme, the proposed $l_t(\theta_t)$ defined in (4.15) follows the property:

$$l_t(\hat{\theta}_t^{(i+1)}) \geq l_t(\hat{\theta}_t^{(i)}).$$

where the proof can be found at (Godoy et al., 2020). This well reflects that local likelihood does increase at each step.

4.4.2 Demonstration of the Effect of Kernel

We first demonstrate the effect of the inclusion of a Epanechnikov kernel function $K(v)$, for two different peak intensities $G = \{10, 50\}$ with 10 being a relatively weak

signal and 30 a relatively strong one. Fig. 4·23 shows five lines corresponding to two different window sizes each with and without a kernel is used or not, and the real value of D_x. As expected, the kernel smoothing effect is stronger with shorter windows and with lower signal level as in Fig. 4·23(Top) compared to Fig. 4·23(Bottom). These results thus indicate that a kernel is an important element of the estimation process.

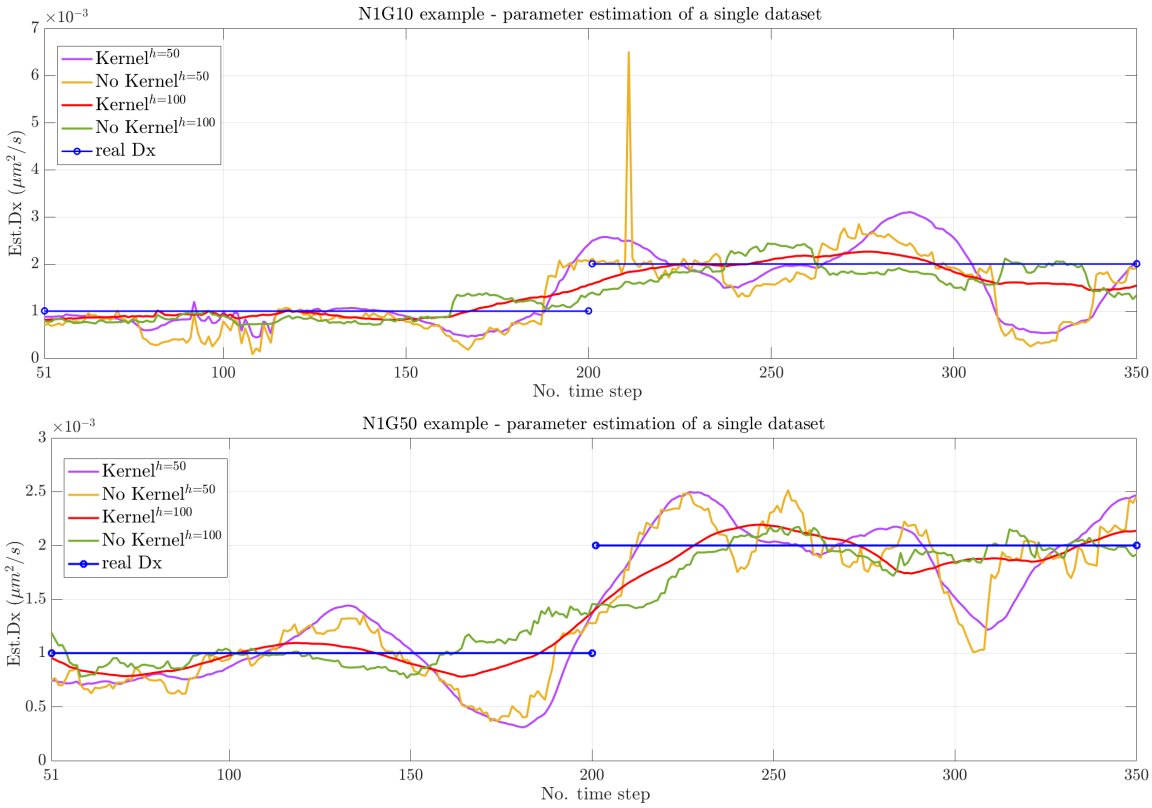


Figure 4·23: Effect of the kernel with window sizes of 50 (purple, with kernel, and yellow, without kernel) and 100 (red, with kernel, and green, without kernel). (Top) Low SNR with $N_{gbd} = 1$ and $G = 10$. (Bottom) High SNR with $N_{gbd} = 1$ and $G = 50$.

4.4.3 Demonstration of the Window Size

We now fix the kernel to the Epanechnikov form and look more deeply at the effect of the different window sizes. The results for $h = 50, 100$, at two different SBR

levels, running 50 different simulation trials, are shown in Fig. 4·24 with the median indicated by the center lines and the middle two quantiles by the shaded area. Color corresponding to window size with 50 in blue and 100 in red. As expected, the figure shows that the smaller window size leads to a quicker response but larger variance. Comparing the results in Fig.4·24, we note an increased variance in the estimate at lower SBR levels.

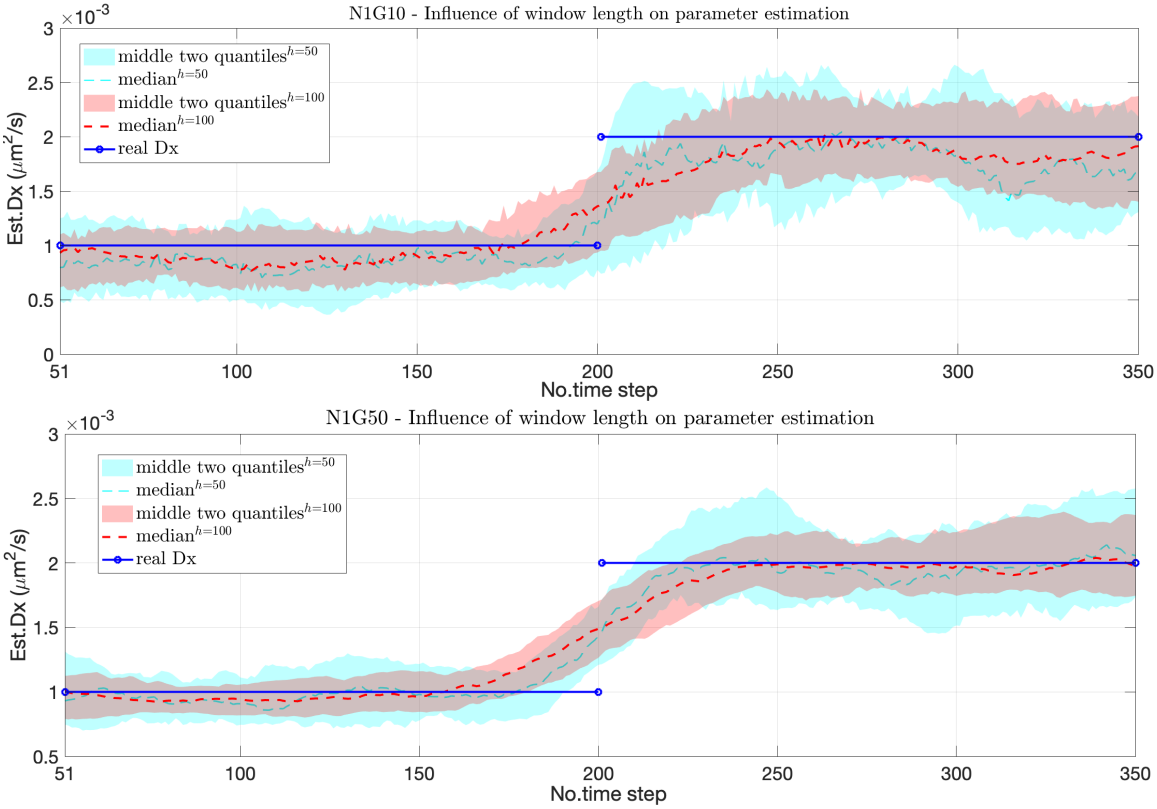


Figure 4·24: Effect of window length h=50 (light blue) and h=100 (red) with median across 50 trials indicated by dashed lines and the center two quantiles by the shaded region. (Top) Low SBR with $N_{bgd} = 1$, $G = 10$. (Bottom) High SBR with $N_{bgd} = 1$, $G = 50$.

Table 4.11 shows the RMSE (over 50 runs) for different window lengths and signal intensities. The table indicates improvement in terms of accuracy for longer window lengths but with diminishing returns as the window increases. Note that while here

Table 4.11: Root mean square error (RMSE) for different window lengths and signal intensity (G)

window h	(RMSE ± std) $\times 10^{-3}$ G=50	(RMSE ± std) $\times 10^{-3}$ G=10
50	0.8006 ± 0.1408	0.9197 ± 0.2752
75	0.7616 ± 0.1406	0.8179 ± 0.2626
100	0.7375 ± 0.1386	0.7622 ± 0.2547
125	0.7144 ± 0.1347	0.7227 ± 0.2515

the window length is chosen empirically, there are data-driven methods available, e.g. the Steins Unbiased Risk Estimator (SURE) (Long et al., 2005), to select good window sizes.

4.4.4 Demonstration of the Slide Length

The proposed scheme involves non-trivial calculations as, in each window, the EM needs to be run across the entire set of data in that window. While these computations can be performed off-line, implying that the computational complexity is not a major concern, it is often a matter of convenience to a user for estimates to be performed as quick as possible. One simple approach for reducing the overall computation time is to simply shift the window by more than one data point each time. In Fig.4·25, we show the results of running the time-varying approach, stepping forward by 1, 5, 20, and 50 steps.

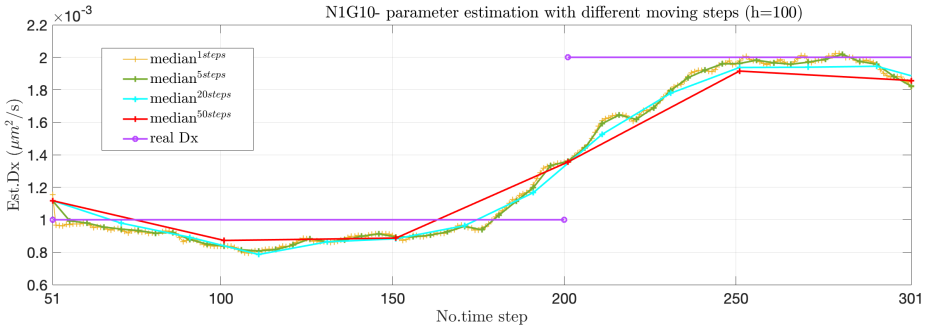


Figure 4·25: Comparison of different shift sizes with shifts of 1 (yellow), 5 (green), 20 (blue), and 50 (red). The true parameter is shown in purple.

Somewhat surprisingly, there is little loss in accuracy, though of course there is loss of sensitivity to multiple, rapid changes in parameter values with larger shifts.

So far we have introduced an algorithm for time-varying parameter estimation in the context of single particle tracking where we have a complex, nonlinear measurement model. The proposed algorithm considers the use of previously developed tools, namely the EM algorithm combined with a UKF and the URTSS, in a local approach. Using physically accurate simulations, we explored the effect of using a kernel and the window lengths, all at two different SBRs. We also considered the effect of increasing shifts in the window as a means of reducing the computational load of the analysis technique.

Chapter 5

Computationally Efficient Application of Sequential Monte Carlo - Expectation Maximization

5.1 Variant of Sequential Monte Carlo Method

Under the EM framework, the selection of filtering and smoothing algorithms is very flexible so long as they produce the necessary distributions. In earlier chapters, we have made use of the basic version of SMC-EM where the SMC components is comprised of an SIR filter and a FFBS. In this chapter, we turn instead to an alternative Gaussian Particle Filter (aGPF) (Kotecha and Djuric, 2003) and a Backward Simulation Particle Smoother (BSPS) (Godsill et al., 2004) to produce a more computationally efficient version of SMC-EM. While the improved efficiency does theoretically come at the cost of accuracy in the representation of the distributions, our results indicate that, at least for the models considered, this impact is small in practice. To distinguish these two versions of SMC-EM, we denote SMC_1-EM as the old version and SMC_2-EM as the newer, more computationally efficient version.

The Gaussian Particle Filter (GPF) (Kotecha and Djuric, 2003) approximates the filtered distribution as a Gaussian (though extensions to more general distributions through propagating higher moments are straightforward). As with all such filters, the GPF consists of a *measurement update step* and a *time update step*. In the measurement update step, particles are drawn from an importance sampling function,

$\pi(x_t|Y_t)$. The choice of this function is informed by the specific problem (see (Handschin and Mayne, 1969; Doucet et al., 2000; Tanizaki, 2003) for details) but often this is selected to be the *prediction distribution* (a normal distribution with a mean and covariance determined in the time update step). These samples are weighted based on the measurement distribution and their sample mean and covariance calculated to produce the *filtered distribution*. In the time update step, samples are drawn from this filtered distribution, propagated through the motion model, and the sample mean and covariance determined to produce the prediction distribution. In a simplification known as the alternate GPF (aGPF), the weighted samples from the filtered distribution are reused in the time update, avoiding the need to resample particles.

To further reduce the computational load, we selet all the particles at the time update step to be at the mean of the filtered distribution. These are then propagated through the motion model as usual. We also follow a suggestion of (Kotecha and Djuric, 2003) and use these propagated particles in the measurement update rather then drawing from the importance function. With these choices, we avoid the need for resampling from any distribution, simplify the implementation of the propagation through the motion model, and avoid the need to calculate explicitly the covariance matrix of either the filtered of the predicted distribution. We refer to this algorithm as the simplified aGPF (sGPF); it is summarized in Alg. 1.

To initialize the filter at time t_0, the first samples are drawn from an importance function defined as a uniform density over a cubic volume centered at $(x_o, y_o, 0)$, where (x_o, y_o) define the center of the first image, with a side length of twice the pixel size Δx. This is a somewhat arbitrary choice and can be easily modified if there is additional information available or if a larger uncertainty is needed at the initial time.

The BSPS, shown in Algorithm 2, takes the filtered weights of the sGPF $\{\tilde{w}^j_{norm,t}, x^j_t : t = 1, ..., N; j = 1, ..., M\}$ and calculates the smoothed weights. The computational

Algorithm 1 simplified GPF

/* Initialize */
    Set t = 0
    Sample particles $\tilde{x}_0^i$, $i = 1, \ldots, M$, from the initial importance function

/* Measurement update */
    /* Compute particle weights */
    $\tilde{w}_t^i = p(y_t|\tilde{x}_t^i)$, $i = 1, \ldots, M$.
    /* Normalize the weights */
    $\tilde{w}_{norm,t}^i = \dfrac{\tilde{w}_t^i}{\sum_{j=1}^M \tilde{w}_t^j}$, $i = 1, ..., M$.
    /* Calculate the mean of the filtered distribution */
    $\mu_t = \sum_{j=1}^M \tilde{w}_{norm,t}^j \tilde{x}_t^j$.

/* Time update */
    /* Initialize M particles */
    $\tilde{x}_t^i = \mu_t$, $i = 1, \ldots, M$.
    /* Propagate through the motion model */
    $\tilde{x}_{t+1}^i = f_t(\tilde{x}_t^i, w_t, \theta)$, $i = 1, 2, \ldots, M$.

complexity of this scheme is $O(NMK)$. We note that there are other variants of BSPS with even better computational complexity, such as BSPS with rejection sampling (Douc et al., 2011). Such approaches, however, must compute upper bounds on the state transition density and we find the cost of doing so for the confined model considered here makes such methods more costly than simple BSPS. One of the benefits of BSPS, however, is that the simulated trajectories are independent to each other. Our implementation takes advantage of this to simplify the computational load by parallelizing computations when possible.

5.2 Three Dimensional Diffusion with Confined Mode

5.2.1 Model Simplification

The complete one step distribution for the confined diffusion motion model in (2.8) involves an infinite sum. Our approximation is simply an informed choice of the

Algorithm 2 Backward Simulation Particle Smoother

/* Simulate K smoothed trajectories from time N */
for $k = 1$ to K **do**
  /* Draw index ℓ according to weights $\{\tilde{w}^j_{norm,N}\}_{j=1}^M$ */
  set $\tilde{x}^k_{N|N} = \tilde{x}^\ell_N$.
end for
/* Propagate the remaining simulated trajectories */
for $t = N-1$ to 1 **do**
  for $k = 1$ to K **do**
    /* Compute new weights */
    $\tilde{w}^{j,k}_{t|N} \propto w^j_t p(\tilde{x}^k_{t+1|N}|\tilde{x}^j_t)$ for $j = 1, ..., M$.
    /* Normalize the smoothing weights */
    $\tilde{w}^{j,k}_{t|N} = \frac{\tilde{w}^{j,k}_{t|N}}{\sum_{j=1}^M \tilde{w}^{j,k}_{t|N}}$, $j = 1, ..., M$.
    /* Draw index ℓ according to $\{\tilde{w}^{j,k}_{t|N}\}_{j=1}^M$ */
    set $\tilde{x}^k_{t|N} = \tilde{x}^\ell_t$.
  end for
end for

number of terms needed to maintain good accuracy without undue computation. For simplicity of notation we define

$$\beta \triangleq \frac{D\Delta t\pi^2}{L^2}.$$

Writing the probability distribution in (2.8) out to N_p terms we get

$$p_{N_p}(x_{t+1}|x_t) = \frac{1}{L} + \frac{2}{L} \times \sum_{n=1}^{N_p} \exp^{-\beta n^2} \cos\left[\frac{n\pi}{L}\left(x_{t+1} + \frac{L}{2}\right)\right] \cos\left[\frac{n\pi}{L}\left(x_t + \frac{L}{2}\right)\right]. \tag{5.1}$$

Note that while this is no longer a true distribution (as it does not integrate to one), the goal in what follows is to choose N_p so that this error is small enough to ignore.

To that end, define the error

$$\begin{aligned} e &\triangleq p(x_{t+1}|x_t) - p_{N_p}(x_{t+1}|x_t) \\ &= \frac{2}{L}\sum_{n=N_p+1}^{\infty} \exp^{-\beta n^2} cos\left[\frac{n\pi}{L}\left(x_{t+1}+\frac{L}{2}\right)\right]\cos\left[\frac{n\pi}{L}\left(x_t+\frac{L}{2}\right)\right]. \end{aligned} \tag{5.2}$$

We then have

$$|e| \leq \frac{2}{L}\sum_{n=N_p+1}^{\infty} \exp^{-\beta n^2} \approx \frac{2}{L}\sqrt{\frac{\pi}{4\beta}}\text{erfc}(\sqrt{\beta}N_p), \tag{5.3}$$

where erfc is the complementary error function, given by

$$\text{erfc}(x) = \frac{2}{\sqrt{\pi}}\int_x^{\infty} \exp^{-t^2}\, dt. \tag{5.4}$$

Let ϵ denote a desired threshold on the error. Then, from (5.3), the needed number of terms is

$$N_p = \frac{1}{\sqrt{\beta}}\text{erfc}^{-1}\left(\frac{\epsilon L}{2}\sqrt{\frac{4\beta}{\pi}}\right). \tag{5.5}$$

This expression depends on the parameters of the confined diffusion model, L and D (through β), and is illustrated for the case of $L = 0.5$ in Fig. 5·1. While these values are not known a priori, in practice it is reasonable to assume that approximate bounds on their values are known, guided by the physical problem and prior experience.

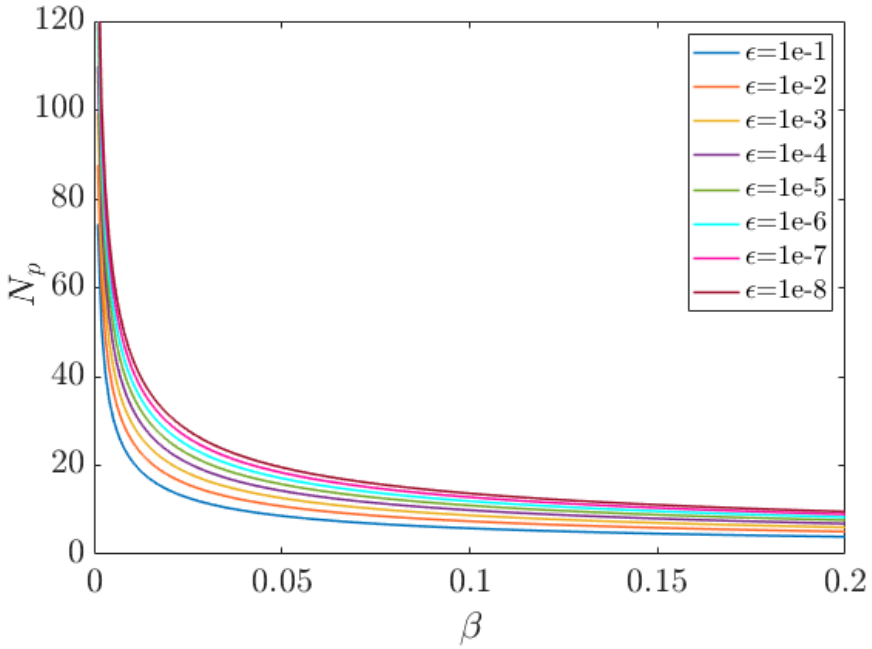


Figure 5·1: Relationship between N_p and β.

In addition, as Fig. 5·1 illustrates, high accuracy is achieved over a wide range of parameters using a very small number of terms.

We turn our attention now to the measurement model in (2.18). The zeroth-order Bessel function of the first kind is

$$J_0(r) = \sum_{n=0}^{\infty} \frac{(-r^2/4)^n}{n!^2}. \tag{5.6}$$

To reduce the computational cost of evaluating (5.6), we rely on a method introduced in (Kadri, 2019). This technique approximates the near-field (small r) and far field (large r) using simple cosine functions,

$$J_{0,\text{near}} = \cos(\mathcal{M}r)\,,\; J_{0,\text{far}} = \sqrt{\frac{2}{\pi b_1 r}}\cos(b_2 r - b_3)\,, \tag{5.7}$$

where $\mathcal{M}$ and the b_i $(i = 1, 2, 3.)$ are constants chosen to match the true Bessel function at select locations. More details can be found at (Kadri, 2019). To use this

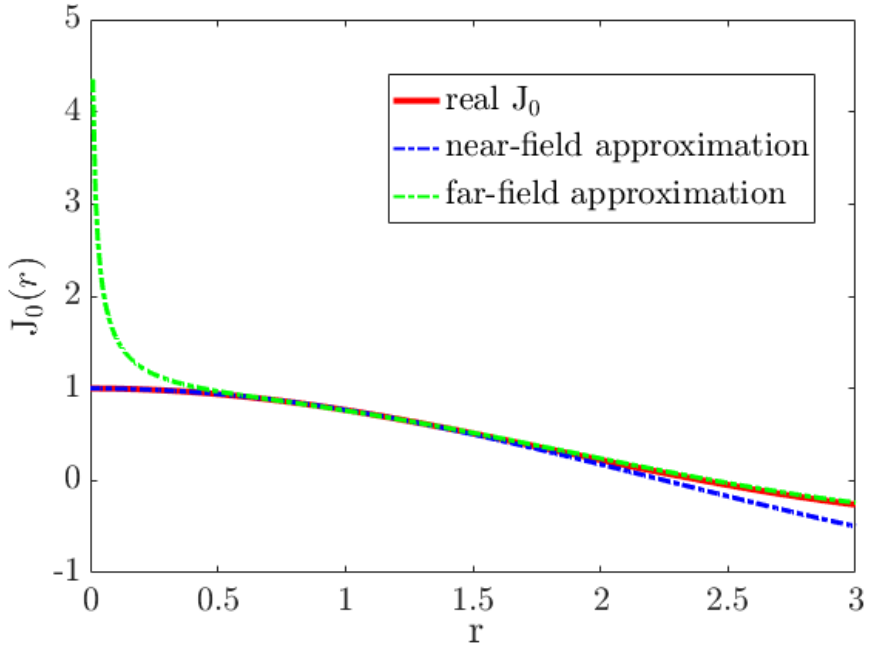


Figure 5·2: Bessel approximation methods.

in our observation model, we set r to be

$$r \triangleq \kappa \sin\theta \sqrt{x^2 + y^2} \tag{5.8}$$

where κ is a constant physical value determined by

$$\kappa \triangleq 2\pi n/\lambda \tag{5.9}$$

where n is the refraction index and λ is the emission wavelength.

These two approximations, together with the true function, are shown in Fig. 5·2. The near-field approximation is quite accurate for small r. As we are assuming our image data has been segmented, we know that the true particle location lies within the $\sqrt{P} \times \sqrt{P}$ pixels and thus r is guaranteed to be small, allowing us to use only the near field approximation in the remainder of this paper.

To demonstrate the impact of these two modifications, we show the computation time of the original formulation (which involved 10,000 terms in the confined model

(2.8) and a numerical evaluation of the Bessel function) and of the modified versions (using 20 terms in (2.8) and the near-field approximation of J_0). The calculations were carried out in Python 3.8 on a 2.3 GHz Intel Core i5 running MacOS 10.15. Since these models are evaluated at each time step, these reductions have a significant impact on the overall time of our estimation algorithm.

Table 5.1: Runtime comparison: original vs. simplified

Runtime (sec)	Before modification	After modification
Motion model (2.8)	5.645862	0.000924
Measurement model (2.19)	48.551610	0.063256

Because the single particle is smaller than the diffraction limit of light, the image on the camera is described by the PSF of the instrument. In 2-D (and in the focal plane of the objective lens), the PSF is well approximated by

$$PSF(x, y; x_o, y_o) = G \cdot \exp\left(-\frac{(x-x_o)^2}{2\sigma_x^2} - \frac{(y-y_o)^2}{2\sigma_y^2}\right), \quad \sigma_x = \sigma_y = \frac{\sqrt{2}\lambda}{2\pi \mathrm{NA}}, \quad (5.10)$$

where (x_o, y_o) is the location of the particle, G is the peak intensity of the fluorescence, λ is the wavelength of the emitted light and NA is the numerical aperture of the objective lens being used (Zhang et al., 2007).

5.2.2 Observation by CCD/EMCCD

In this part, we demonstrate the performance of SMC_2-EM, comparing it to SMC_1-EM and then take advantage of the computational efficiency of SMC_2-EM to investigate the effect of data length on estimation performance.

In this work, the motion model and measurement model follow (2.8) and (2.10) respectively. We assume the images were formed with Born-Wolf PSF in (2.18). The parameter values chosen for the simulations are typical values in the biophysical

setting (see, e.g., (Chenouard et al., 2014; Ashley and Andersson, 2015)) and are listed in Table 5.2. In practice, images are captured at a given frame rate (every Δt units of time). Photons are collected during the shutter period δt (with $\delta t \leq \Delta t$. The particle being tracked of course is in motion while the shutter is open, leading to an effect known as motion blur. While imaging parameters are typically chosen to minimize this effect, the blur is not zero. To account for this, each of our images is generated as the sum of N_{sub} sub-images collected over the shutter period δ_t. For each setting considered, 25 datasets were simulated to produce statistics on the results. All simulation work was carried out in the Python 3.8 environment.

Table 5.2: Imaging parameters for 3D simulation with confined motion.

Symbol	Parameter	Value
D_x, D_y, D_z	real diffusion coefficient in 3D	0.01 μm^2/s
L_x, L_y, L_z	length of confined channel in 3D	500 nm
G	peak intensity	100 counts
N_{bgd}	background intensity rate	10 counts
N	number of image frames/dataset	100
n	Refraction index	1.33
N_{sub}	subsamples per image	100
α	maximum semiangle of objective lens	$\sin^{-1}(\frac{NA}{n})$
δt	shutter period	10 ms
Δt	imaging period	100 ms
κ	wave number of the emitted light	$2\pi n/\lambda$

For the motion model approximation, $Np = 40$ was selected in the expansion (giving an error of $\epsilon < 1 \times 10^{-5}$). When implementing the two versions of SMC-EM, both parallel processing (on four cores) and a table-lookup (for the double integrals in (2.19)) were used. The motion model parameters to be estimated were the confinement lengths in three dimensions ($L_{x,y,z}$) and the diffusion coefficients in three dimensions ($D_{x,y,z}$). The estimation of the confinement length in the x direction at

the $(e+1)^{th}$ EM iteration is

$$\hat{L}_{x,e+1} = \max_t 2|\hat{x}_{t|N,e}| \tag{5.11}$$

where $\hat{x}_{t|N,e}$ is the smoothed estimate of the particle position at time t for the e^{th} EM iteration. Estimates in y and z are similar. The diffusion coefficients were estimated by numerically finding the root of the derivative of the (particle-based approximation to the) $\mathcal{Q}$ function.

Performance is measured through both the RMSE of the localization of the tracked particle across the trajectory and the accuracy of the parameter estimates. Note that because the measurement model (2.18) is symmetric above and below the focal plane (defined as $z = 0$), we cannot distinguish between a particle being above or below that plane. Thus, when considering the RMSE we use $|z|$.

As shown in Fig. 5·3, SMC_2-EM is significantly faster than SMC_1-EM. Even with 1000 Monte Carlo samples, SMC_2-EM remains much faster than our original approach.

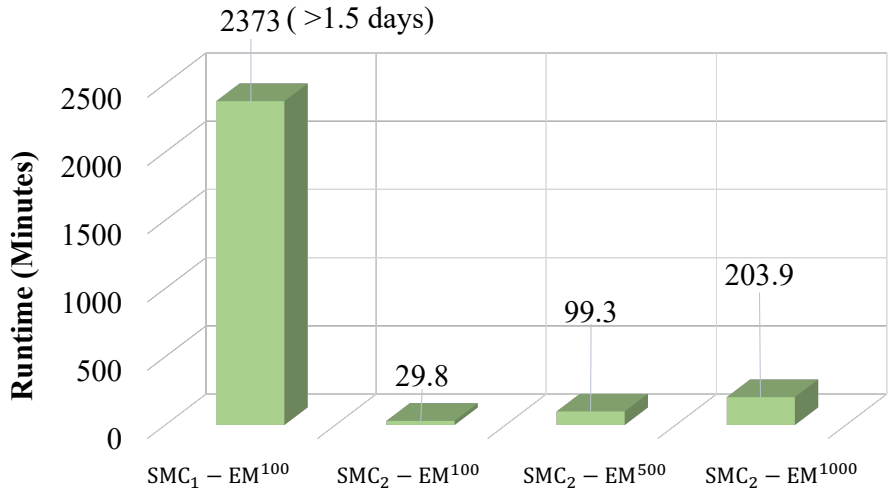


Figure 5·3: Runtime of different versions of SMC-EM for 10 EM iterations on a single dataset with image length $N = 100$.

Then we compared the performance of localization and parameter estimation under the two versions of SMC-EM using 100 image frames (a typical value in practice) and 100 Monte Carlo samples in each algorithm. The RMSE of the localization for the two methods is shown as boxplots in Fig. 5·4, indicating that the simplifications of SMC_2-EM have little effect on localization accuracy.

Taking advantage of the computational efficiency of SMC_2-EM, we increased the Monte Carlo samples from 100 to 500 and to 1000. The localization performance for all cases are shown in Table 5.3.

Table 5.3: Mean and standard deviation of localization performance with $N = 100$ images.

Method	RMSE_x (nm)	RMSE_y (nm)	$\text{RMSE}_{\lvert z\rvert}$ (nm)
SMC_1-EM^{100}	15.3045 ± 1.4592	15.0352 ± 1.3360	47.2614 ± 5.7451
SMC_2-EM^{100}	15.6719 ± 1.2892	15.6274 ± 1.4743	49.8244 ± 5.6984
SMC_2-EM^{500}	13.1121 ± 1.1169	13.0864 ± 1.1250	37.3751 ± 4.8774
SMC_2-EM^{1000}	12.9188 ± 1.1550	12.8046 ± 1.1451	35.0089 ± 3.2183

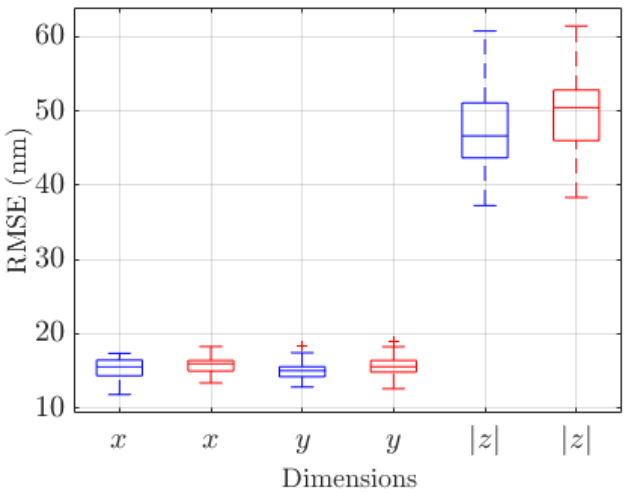
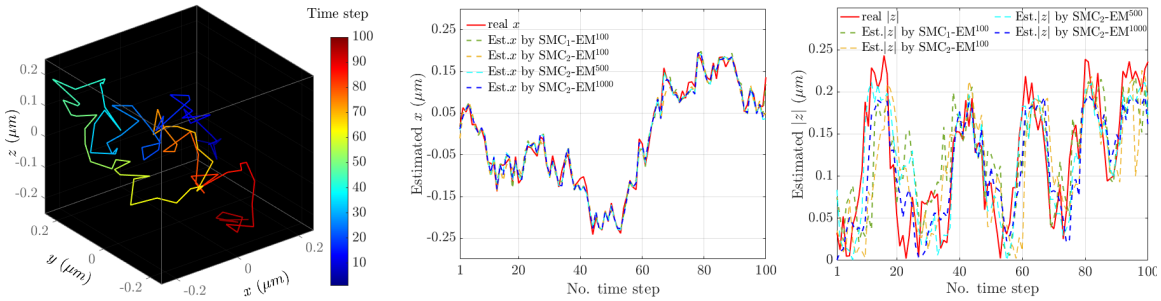


Figure 5·4: Boxplot of root mean square error (RMSE) by SMC_1-EM^{100} (blue) and SMC_2-EM^{100} (red) respectively. Note that the solid line inside the box denotes the median, the edges of the box represent the first and third quartiles, the vertical dashed line indicates the bounds for data within 1.5 times the interquartile range, and the red + symbols are outliers.

As expected, the localization accuracy improves as more Monte Carlo samples are used. A typical run and resulting estimated trajectories can be seen in Fig. 5·5.



(a) (b) (c)

Figure 5·5: Typical results for trajectory estimation using SMC_1-EM and SMC_2-EM. (a) The ground truth trajectory with color indicating time. (b) x-axis results. (Results in y are similar and are omitted for space reasons.) (c) z-axis results. There's little difference in localization between SMC_1-EM and SMC_2-EM, however, there's a big difference in terms of computational efficiency, see Fig. 5·3.

In terms of parameter estimation, the results of the estimated confinement lengths and diffusion coefficients are summerized in Table 5.4. As with RMSE, SMC_2-EM still shows parameter estimation performance on par with SMC_1-EM despite the much faster computation time. Increasing the number of Monte Carlo samples shows some benefit in estimating the diffusion coefficients but has very little effect on determining the confinement length. This can be explained by considering the typical trajectory in Fig. 5·5. With the selected diffusion coefficients and simulation time, the fluorescent particle rarely explores the full confinement. Since the ML estimator for this parameter is simply the range of motion in each axis, this leads to a negative bias in that estimate.

Table 5.4: Parameter estimation performance with $N = 100$ images.

Method	L_x (μm)	L_y (μm)	L_z (μm)	D_x (μm^2/s)	D_y (μm^2/s)	D_z (μm^2/s)
SMC_1-EM^{100}	0.45619 ± 0.021631	0.45409 ± 0.017009	0.42645 ± 0.031515	0.0087497 ± 0.0013610	0.0088405 ± 0.0016812	0.011935 ± 0.0036097
SMC_2-EM^{100}	0.45471 ± 0.022704	0.45371 ± 0.017289	0.42542 ± 0.037334	0.0085595 ± 0.0014248	0.0088739 ± 0.0017232	0.010455 ± 0.0039255
SMC_2-EM^{500}	0.45834 ± 0.017476	0.45373 ± 0.016257	0.39403 ± 0.034646	0.0092509 ± 0.0014024	0.0091535 ± 0.0014212	0.009164 ± 0.0025437
SMC_2-EM^{1000}	0.45701 ± 0.017557	0.45511 ± 0.017103	0.38553 ± 0.029079	0.0093116 ± 0.0013983	0.0093294 ± 0.0014023	0.010374 ± 0.0042114

One of the advantages of SMC_2-EM is its ability to handle larger data sets due to its lower complexity. Since tracking a particle for a longer time allows it to more effectively explore its entire confined region, one expects that the estimation of the confinement parameter will get improved with longer data runs. To explore this, we ran simulations with the same parameter settings as before but now using 50, 100, 500, and 1000 images. The localization estimates are summarized in Table 5.5 and the parameter estimates in Table 5.6.

As expected, there is improvement in all metrics as the data length is increased with the largest impact being on the estimation of the confinement length.

Table 5.5: Influence of data length on localization using SMC_2-EM^{100}.

Data length N	$RMSE_x$ (nm)	$RMSE_y$ (nm)	$RMSE_{\|z\|}$ (nm)
50	16.8254 ± 3.2570	15.6224 ± 2.5032	48.3973 ± 10.4022
100	15.6719 ± 1.2892	15.6274 ± 1.4743	49.8244 ± 5.6984
500	14.8394 ± 0.7646	15.3108 ± 3.0564	47.8715 ± 3.5421
1000	14.7521 ± 0.4867	14.7728 ± 0.4596	47.2465 ± 1.8068

Table 5.6: Influence of data length on parameter estimation using SMC_2-EM^{100}.

Data length N	L_x (μm)	L_y (μm)	L_z (μm)	D_x (μm^2/s)	D_y (μm^2/s)	D_z (μm^2/s)
50	0.40068 ± 0.075374	0.40825 ± 0.061829	0.38755 ± 0.046502	0.0082067 ± 0.0025068	0.0078197 ± 0.0021422	0.0097096 ± 0.006772
100	0.45471 ± 0.022704	0.45371 ± 0.017289	0.42542 ± 0.037334	0.0085595 ± 0.0014248	0.0088739 ± 0.0017232	0.010455 ± 0.0039255
500	0.48164 ± 0.007607	0.48342 ± 0.006331	0.49621 ± 0.020851	0.0088356 ± 0.0008336	0.0088379 ± 0.0007134	0.010643 ± 0.0021146
1000	0.49247 ± 0.003490	0.49301 ± 0.004967	0.52136 ± 0.020327	0.0087642 ± 0.0005017	0.008880 ± 0.00050147	0.01 0547 ± 0.0011662

5.2.3 Observation by sCMOS

In this part, we use simulations to demonstrate SMC_2-EM and compare its performance against algorithms based on the standard localize-then-estimate paradigm. Without specific illustration, the SMC-EM for all the remaining contents refers to the SMC_2-EM. We assume segmentation of the raw images to a sequence of images containing the DH-PSF of the single particle has been performed. This is, of course,

a non-trivial task, especially at low signal levels. Our interest here, however, is on the relative performance of the estimators and by beginning with the segmented image sequences we can avoid confounding effects of segmentation.

For the comparison, we use two approaches for stand-alone localization. The first is based on a GF to each lobe of the DH-PSF in the image to determine their centers; the particle location is then determined from (2.22) and (2.23). The second is $\text{MLE}_{\text{sCMOS}}$, developed in (Huang et al., 2013) for estimating two-dimensional particle locations from sCMOS camera data, and modified here to include the DH-PSF for 3D localization. Once the particle trajectory has been determined, we estimate model parameters using either the MSD or a fast, MLE-like scheme. For the MSD, we perform a nonlinear fit to the MSD based on the confined diffusion model in (Kusumi et al., 1993). For the MLE-like scheme, we use the true MLE for the confinement length; as shown in (Ashley and Andersson, 2015) this estimator is simply given by the range in the estimated trajectory in each axis.

However, the nonlinear nature of the motion model precludes a simple solution of the MLE of the diffusion coefficient. Rather than utilizing a numerical solver, we apply a computationally efficient approximation to MLE for the diffusion coefficient assuming a free diffusion model as developed in (Berglund, 2010; Michalet and Berglund, 2012). We denote these combinations of algorithms using *localization-estimation* so that, for example, GF-MLE refers to localization using a Gaussian fit and parameter estimation using the MLE-like scheme, while $\text{MLE}_{\text{sCMOS}}$-MSD refers to localization using the $\text{MLE}_{\text{sCMOS}}$ algorithm and parameter estimation using a fit to the MSD.

Simulations were made of a particle diffusing in 3D, confined to a cube centered at the origin with dimensions of 500 nm. Each simulation consisted of $N = 100$ frames at an imaging rate of 10 frames/s (for a period of $\Delta t = 100$ ms). In practice, cameras

accumulate photons over an integration period and the resulting motion blur arising from particle motion during integration is an important factor in performance.

To replicate this effect, trajectories of length $N \times N_{sub}$ were generated where N_{sub} represents a sub-sampling factor. An image is generated at each of the subsample steps in the shutter period δ_t. These are then averaged together to produce the final image for that imaging period to incorporate the effect of motion blur. The ground truth position for that image is taken to be the average of the particle position over the shutter period. The parameter settings for all simulations are summarized in Table 5.7. To generate statistics on algorithm performance, 50 datasets were simulated for each experimental setting.

Table 5.7: Simulation Parameter Settings

Symbol	Parameter	Value
$D_{x,y,z}$	diffusion coefficient in $\{x, y, z\}$	0.01 μm^2/s
$L_{x,y,z}$	confinement length in $\{x, y, z\}$	500 nm
N_{bgd}	background intensity rate	10 counts
G	peak intensity	$10 - 50$ counts
N	number of image frames/dataset	100
N_{sub}	subsamples per image	100
NA	numerical aperture	1.2
λ	emission wavelength	540 nm
r	distance between DH-PSF lobes	300 nm
k	gain of z to θ in DH-PSF	$-0.1\pi/180$ rad
σ_{xy}	width of each lobe of DH-PSF	234 nm
P	number of pixels	225 pixels
$\Delta x, \Delta y$	effective pixel width	100 nm
δt	shutter period	10 ms
Δt	imaging period	100 ms
QE	camera quantum efficiency	1.0
ε	peak expected value of readout noise	10 counts

To investigate the behavior of the algorithms at low signal levels, we varied the peak intensity G across the range of 10-50 counts, holding the background rate fixed.

The SNR of the images can be defined as (Long et al., 2012; Beier and Ibey, 2014)

$$SNR = \frac{G \times QE}{\sqrt{(G + N_{bgd}) \times QE \times F_n^2 + \varepsilon^2}} \tag{5.12}$$

where F_n is the excess noise factor ($F_n = 1$ for sCMOS cameras) and the other terms are defined in Table 5.7). Because we directly set the background and peak intensity level, we report results as a function of Signal-to-Background Ratio (SBR), defined by

$$SBR = \frac{G}{N_{bgd}}. \tag{5.13}$$

From (5.12) and (5.13), the corresponding SNR and SBR levels considered in the simulations are shown in Table 5.8.

Table 5.8: Relationship between SNR and SBR

SBR	1	2	3	4	5
SNR	0.913	1.75	2.53	3.27	3.95

Typical images generated by the DH-PSF of a single particle under these SBR conditions are shown in Fig. 5·6.

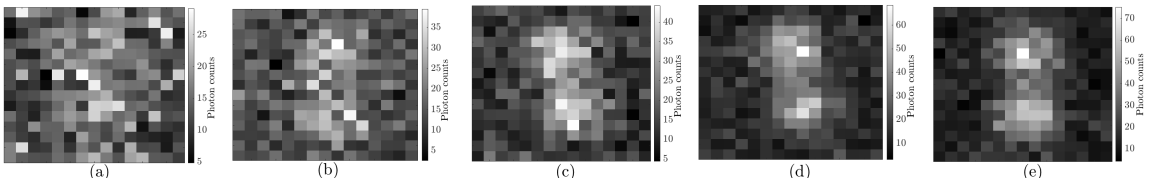


Figure 5·6: Typical images at (a) SBR=1, (b) SBR=2, (c) SBR=3, (d) SBR=4, and (e) SBR=5.

Performance across different signal-to-background ratio (SBR)

We measure localization performance using the Root Mean Square Error (RMSE) over the entire trajectory as compared to the ground truth. Due to the motion blur effect, the notion of "ground truth" must be defined; here we use the average of the

positions of the true trajectory over the integration period. The results over these simulations are shown in Fig. 5·7 as a function of SBR with GF shown in black/gray, $\text{MLE}_{\text{sCMOS}}$ in red, SMC^{100}-EM in green, and SMC^{1000}-EM in purple. The mean for each algorithm is shown as a dashed line, the median as a dotted line and the shadowed region shows the 10%-90% quantiles of the estimates.

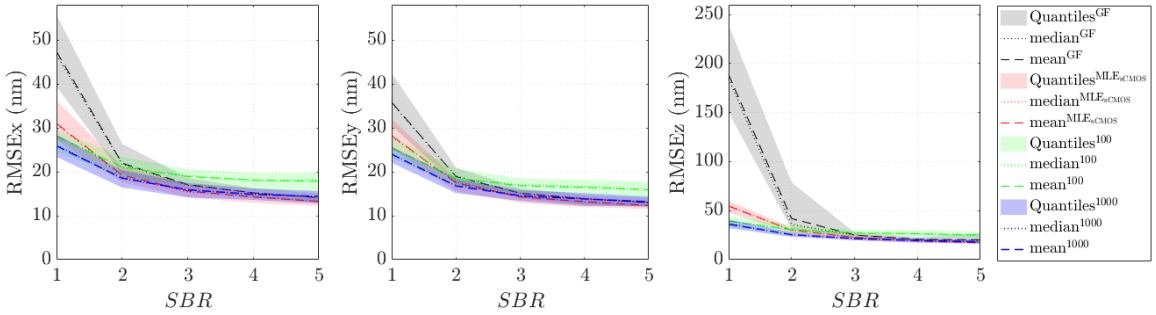


Figure 5·7: Localization performance evaluated by RMSE among different methods as a function of SBR. The shadowed area denotes the 10% quantile - 90% quantile of all estimates for each estimator.

The results indicate several interesting features. First, as expected, increasing the number of Monte Carlo samples used in SMC-EM improves localization performance. Second, the algorithms differ primarily at the lowest SBR levels, performing similarly after an SBR of 4. At lower signal levels, the GF shows the lowest accuracy and, in the limiting case of SBR=1 is essentially equivalent to taking the center of the segmented image as the particle location. Both $\text{MLE}_{\text{sCMOS}}$ and SMC-EM, however, yield good localization even at the lowest SBR levels, with SMC-EM showing the best results.

An estimation result on a typical trajectory (with SBR=1) is shown in Fig. 5·8. All of the algorithms follow the general trend of the true trajectory. GF, however, yields multiple large outliers, particularly in the z-direction, driving its larger RMSE. The $\text{MLE}_{\text{sCMOS}}$ is more variable than the SMC-EM methods which use data from the entire sequence of images and the motion model when inferring the particle location.

Because $\text{MLE}_{\text{sCMOS}}$ consistently outperforms the GF approach, we do not consider the GF method further.

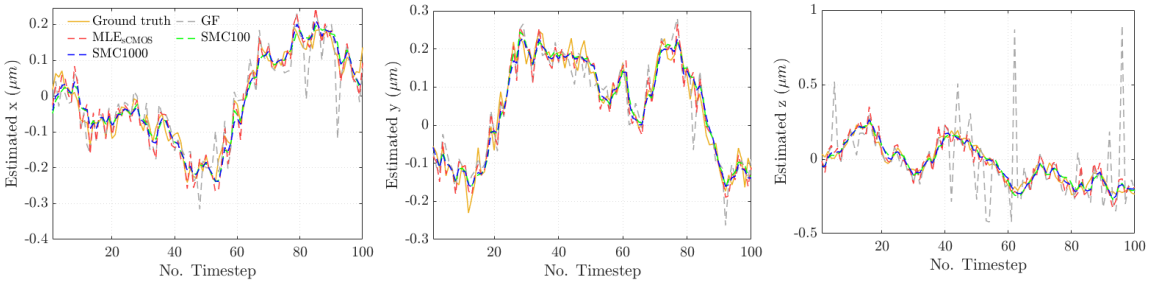


Figure 5·8: Trajectory estimation comparison by different SPT methods at SBR=1.

The results on parameter estimation using the different schemes are shown in Fig. 5·9 for the diffusion coefficient and in Fig. 5·10 for the confinement length. The results from $\text{MLE}_{\text{sCMOS}}$-MSD are shown in light blue, those from $\text{MLE}_{\text{sCMOS}}$-MLE in red, from SMC^{100}-EM in green, and from SMC^{1000} in purple. As before the mean for each algorithm is shown as a dashed line, the median as a dotted line, and the shadowed region shows the 10%-90% quantiles of the parameter estimate.

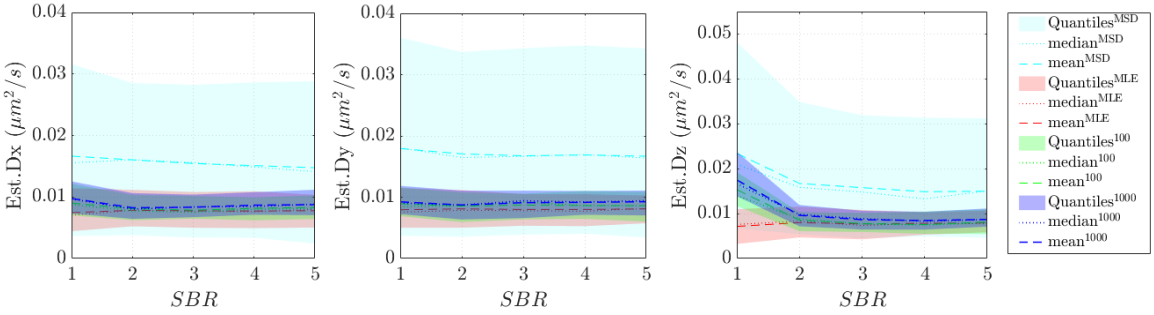


Figure 5·9: Diffusion coefficient estimation performance as a function of SBR. The true diffusion coefficients were $D_x = D_y = D_z = 0.01\ \mu\text{m}^2/s$. The shadowed area shows the 10% quantile - 90% quantile of all estimates.

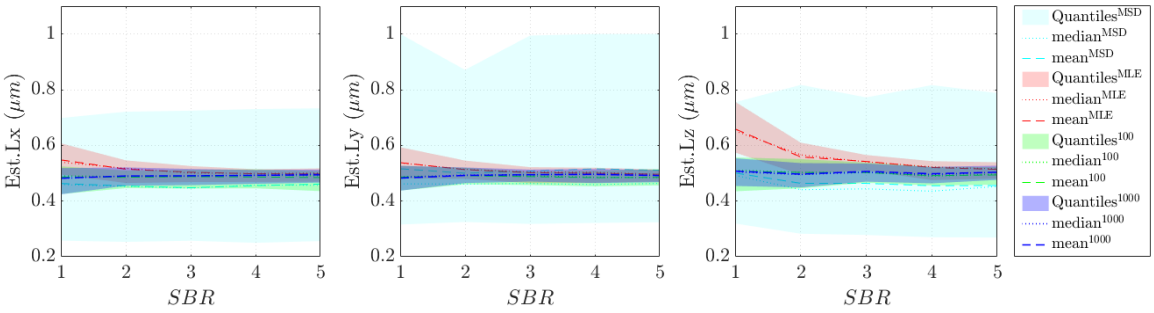


Figure 5·10: Confinement length estimation performance as a function of SBR. The true confinement lengths were $L_x = L_y = L_z = 0.5\ \mu m$. The shadowed area shows the 10% quantile - 90% quantile of all estimates.

These results clearly show that a fit to the MSD yields the worst performance in estimating the model parameters with both a larger bias and a much larger range of estimates. In general, the SMC-EM methods and $\text{MLE}_{\text{sCMOS}}$-MLE have very similar performance for estimating the diffusion coefficient, though the SMC-EM methods have a smaller spread and bias. The exception is in the z axis where the SMC-EM methods exhibit a larger positive bias than $\text{MLE}_{\text{sCMOS}}$-MLE at an SBR of one but otherwise shows the same trend. At SBRs of two and above, this bias remains consistent and, while small, is non-zero. There is some evidence in our prior work that the primary driver of this bias is the relatively short length of the trajectories; as the datalength is increased, this bias decreases (Lin and Andersson, 2021).

The confinement length results show that at SBRs of 4 or above, the SMC-EM and $\text{MLE}_{\text{sCMOS}}$-MLE approaches are quite similar. As the SBR decreases, however, the $\text{MLE}_{\text{sCMOS}}$-MLE overestimates the length while the EM approaches remain accurate, though with an increased spread in the estimates at lower SBR.

Performance across different confinement lengths

It is reasonable to expect that estimation performance will depend on the con-

finement length. If L is very large, the particle will not interact with the boundaries very often and you may need very long runs to get an accurate estimate of the confinement. However, because the motion essentially becomes free diffusion, estimating the diffusion parameter is likely to be easier. Conversely, with small values of L, the nonlinearities in the motion model due to the confinement will be dominant. Because the particle is very likely to interact with the boundaries, even in short trajectories, estimation of L may be easier while the estimation of D may suffer since the motion no longer looks like diffusion. This is particularly likely in the $\text{MLE}_{\text{sCMOS}}$-MLE algorithm since the diffusion estimation assumes a free diffusion model.

To explore this effect, we performed simulations for confinement lengths ranging from 0.1 μm - 0.5 μm. Because $\text{MLE}_{\text{sCMOS}}$ outperforms GF and MLE outperforms MSD, we compared SMC-EM only to $\text{MLE}_{\text{sCMOS}}$-MLE. Based on the results in Chapter 5.2.3, we expect the algorithms to differ at the lowest SBR but to perform similarly at higher SBR. We therefore show results here for SBR=1 and SBR = 5. We also found that confinement length had little effect on localization and focus here on parameter estimation.

The estimation results for SBR=1 are shown in Fig. 5·11. In this figure, results from $\text{MLE}_{\text{sCMOS}}$-MLE are shown in red, from SMC^{100}-EM in green, and from SMC^{1000}-EM in purple. The mean for each algorithm is shown as a dashed line, the median as a dotted line, and the shadowed region shows the 10%-90% quantiles. The true parameter values are shown as a solid black line.

The results indicate that SMC-EM accurately captures the confinement length across the entire range of L considered. $\text{MLE}_{\text{sCMOS}}$-MLE has the correct trend but shows a bias that decreases as the confinement length increases. Because the MLE for the confinement length is given by the range of the estimated trajectory, this likely reflects a larger propensity for occasional outliers in localization. Because the

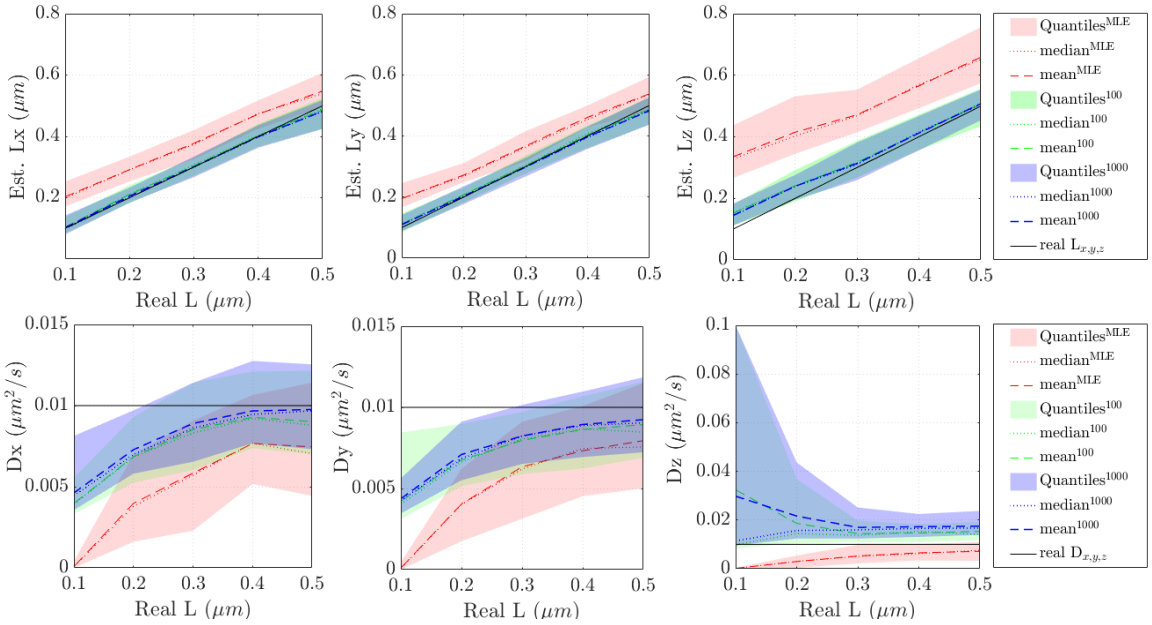


Figure 5·11: Parameter estimation performance as a function of confinement length at SBR=1. The true diffusion coefficients were $D_x = D_y = D_z = 0.01\ \mu\text{m}^2/s$. The shadowed area shows the 10% quantile - 90% quantile of all estimates.

SMC methods utilize the motion model in localization as well as in parameter estimation, such outliers are much less likely. Similarly, the SMC-EM methods are more reliable for diffusion coefficient estimation at smaller confinement lengths. At the smallest confinement lengths, the particle spends most of its time interacting with the boundary of the domain and has limited mobility. As a result the diffusion coefficient is significantly underreported. The estimate improves with increasing L, with the SMC-EM methods consistently outperforming $\text{MLE}_{\text{sCMOS}}$-MLE.

Estimation results for SBR=5 are shown in Fig. 5·12. Because localization is more accurate at the higher SBR, the estimation of confinement length using $\text{MLE}_{\text{sCMOS}}$-MLE is more accurate, matching the performance of SMC-EM. At the smallest confinement lengths, however, $\text{MLE}_{\text{sCMOS}}$-MLE still underestimates the diffusion parameter, performing substantially worse than the SMC-EM approach.

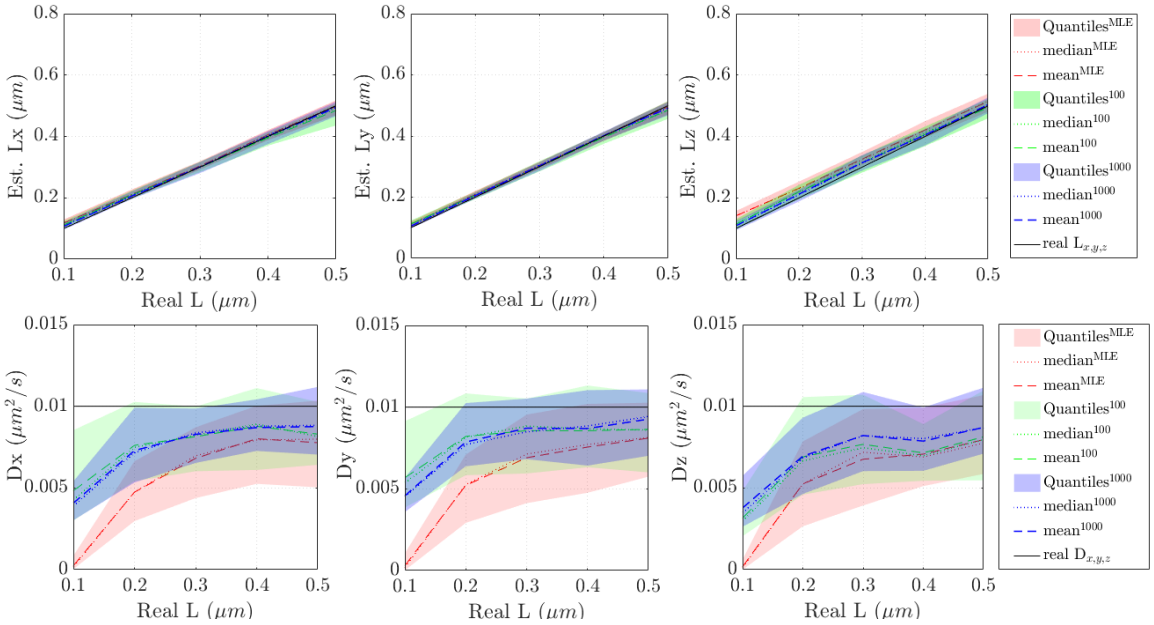


Figure 5·12: Parameter estimation performance as a function of confinement length at SBR=5. The true diffusion coefficients were $D_x = D_y = D_z = 0.01\ \mu\text{m}^2/s$. The shadowed area shows the 10% quantile - 90% quantile of all estimates.

An alternative metric of performance in parameter estimation is the *success rate* defined as the percentage of trials with parameter estimates within 25% of the true value (Michalet and Berglund, 2012). While this is a coarse metric, it provides a simple, concise means of comparing performance across a wide range of conditions. Heat maps showing the success rate of the algorithms across all values of SBR and confinement length we considered are shown in Fig. 5·13 for the confinement length and in Fig. 5·14 for the diffusion coefficient.

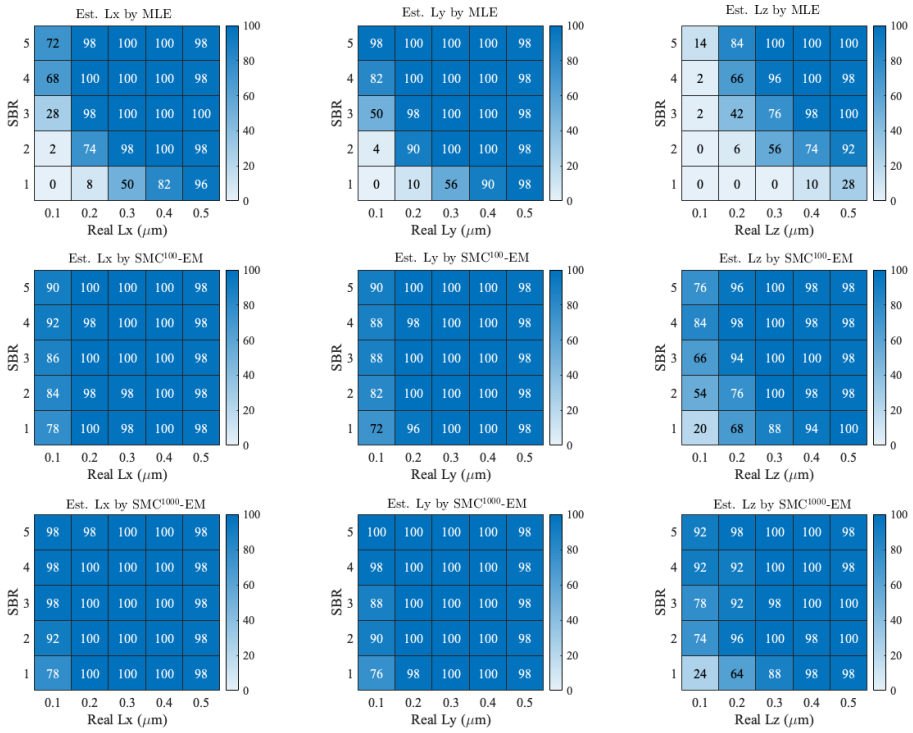


Figure 5·13: Heat map of success percentage in confinement length estimates. (top row) $\text{MLE}_{\text{sCMOS}}$-MLE. (middle row) SMC^{100}-EM. (bottom row) SMC^{1000}-EM.

In each image, the columns show results in the $\{x, y, z\}$ directions, the first row is for $\text{MLE}_{\text{sCMOS}}$-MLE, the second for SMC^{100}-EM, and the third row for SMC^{1000}-EM. The general trend for the confinement length is that both $\text{MLE}_{\text{sCMOS}}$-MLE and SMC-EM accurately estimate this parameter at higher SBR and larger L. As either SBR or confinement length is reduced, the SMC-EM methods outperform $\text{MLE}_{\text{sCMOS}}$-MLE.

This general trend is repeated in the diffusion coefficient estimation shown as Fig. 5·14, with two notable differences. First, the heat maps indicate that at the smallest confinement length and lowest SBR, the SMC-EM methods show a large increase in success rate for the axial diffusion coefficient. Referring back to Fig. 5·11 shows that this is driven by a very large increase in variability of the SMC-EM estimates under

these conditions rather than a true improvement in performance. Similarly, at the smallest confinement lengths, SMC^{100}-EM has a higher success rate than SMC^{1000}-EM. This again is due to the increased variability in the diffusion coefficient estimates, driven now by the smaller number of Monte Carlo samples in the filters.

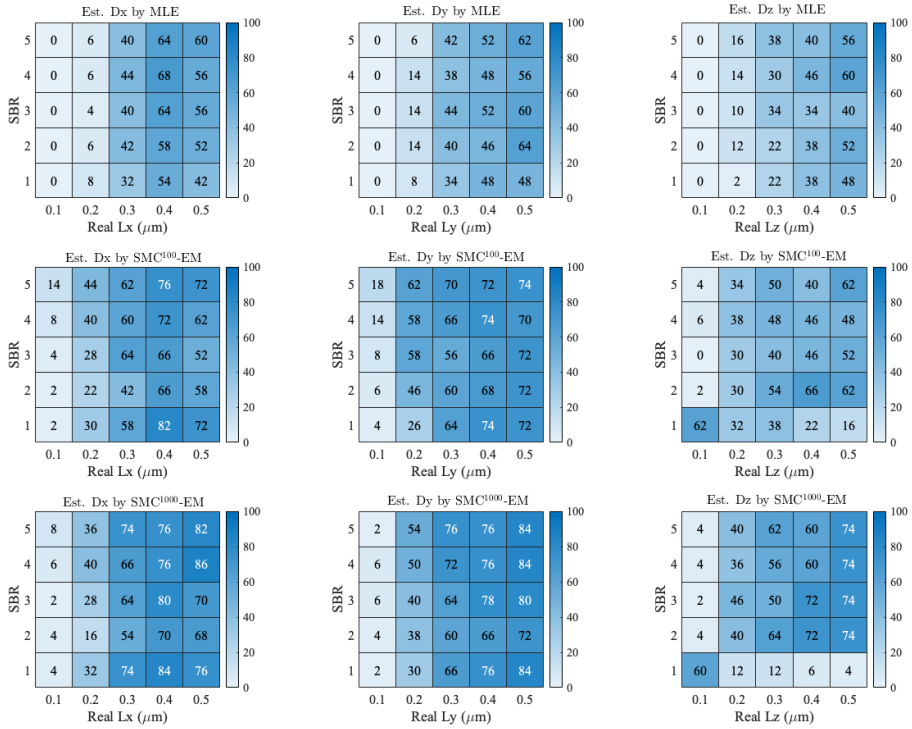


Figure 5·14: Heat map of success percentage in diffusion coefficient estimates. (top row) MLE$_{\text{sCMOS}}$-MLE. (middle row) SMC^{100}-EM. (bottom row) SMC^{1000}-EM.

Additionally, as expected the larger number of Monte Carlo samples in SMC^{1000}-EM improves performance, at the cost of additional computation time. The SMC-EM algorithms were implemented in Python 3.8 on a 2.3 GHz Intel Core i5 running MacOS 11.2. This work utilized significantly faster filtering schemes than the early implementation of SMC-EM found in (Ashley and Andersson, 2015) (see main paper for details).

To compare, we re-implemented the original algorithm in the Python environment, taking advantage of two other computational improvements we included in the newer version, namely parallel processing (on four cores) for the Monte Carlo filters, and a table lookup (to avoid computing the integrals defining the photon rate in each pixel of the camera).

As shown in Fig. 5·15, the updated algorithm (labeled as SMC_2-EM) runs substantially faster. This speed improvement allows us to use a much larger number of Monte Carlo samples, improving the overall performance of the algorithm.

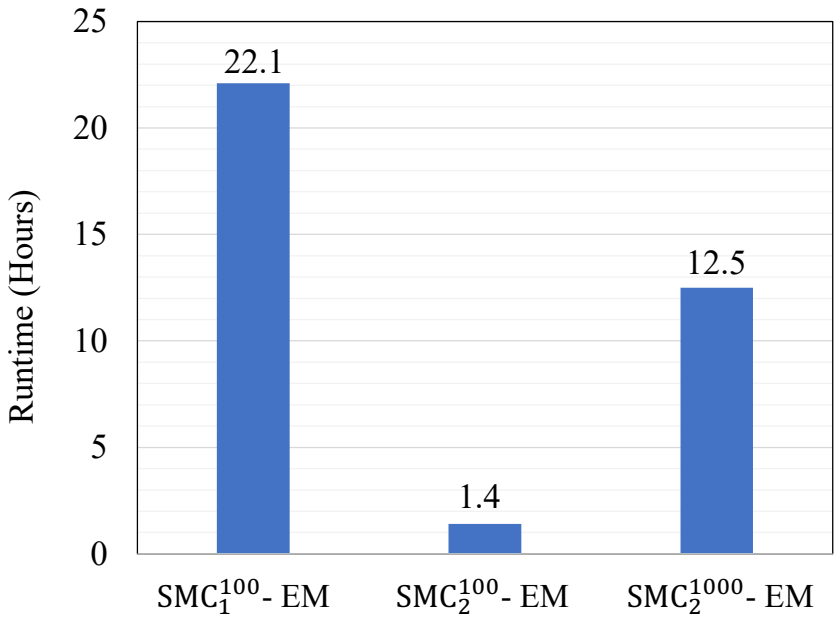


Figure 5·15: Runtime of different versions of SMC-EM for 10 EM iterations on a single dataset with image length $N = 100$.

Fig. 5·16 - 5·20 present the results of the simulation study of localization performance as a function of Signal-to-Background Ratio (SBR) and confinement length. As with the results in the main paper, the results indicate that SMC-EM outperforms the other approaches at the lowest signal level. In addition, our joint localization and estimation scheme is especially beneficial when the particle is tightly confined. In all

the figures, red corresponds to $\text{MLE}_{\text{sCMOS}}$, green to SMC^{100}-EM (using 100 Monte Carlo samples), and purple to SMC^{1000}-EM. The median of the results is shown as a dotted line and the mean as a dashed line. The shadowed region shows the 10%-90% quantiles. The results are organized from smallest confinment ($L = 0.1$ μm) to largest ($L = 0.5$ μm).

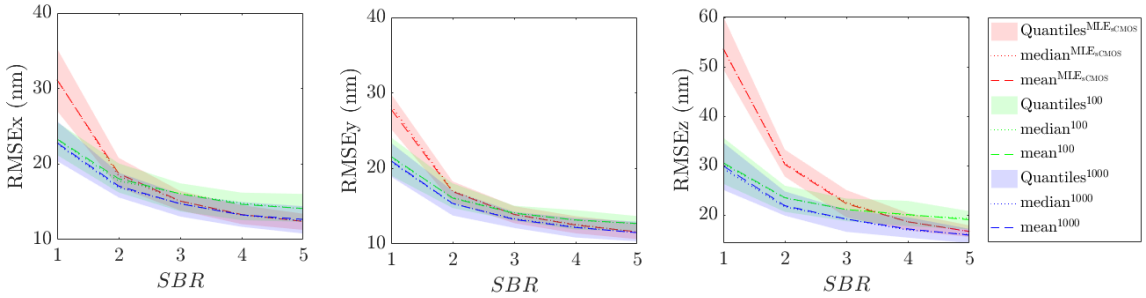


Figure 5·16: True L=0.1 μm: RMSE as a function of SBR. The shadowed area indicates the 10% - 90% quantiles, same to the remaining Fig. in this chapter.

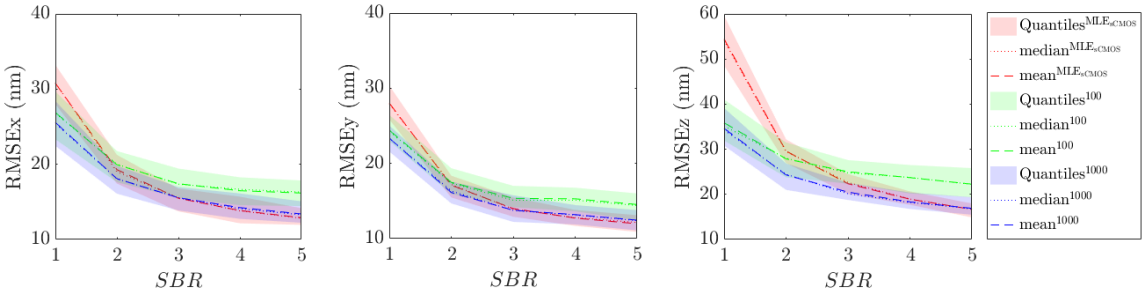


Figure 5·17: True L=0.2 μm: RMSE as a function of SBR.

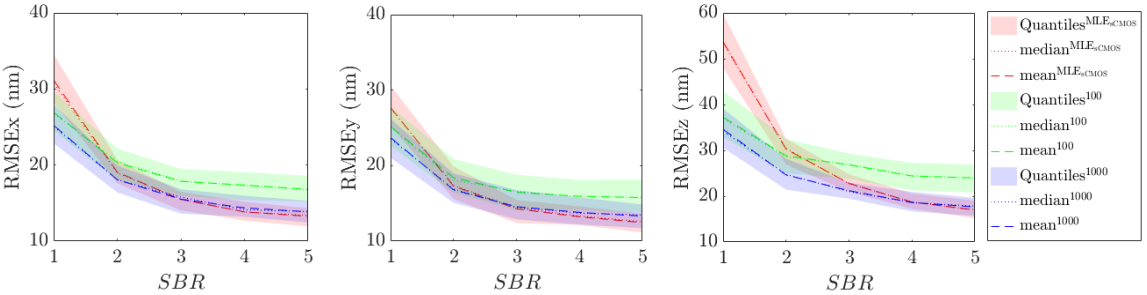


Figure 5·18: True L=0.3 μm: RMSE as a function of SBR.

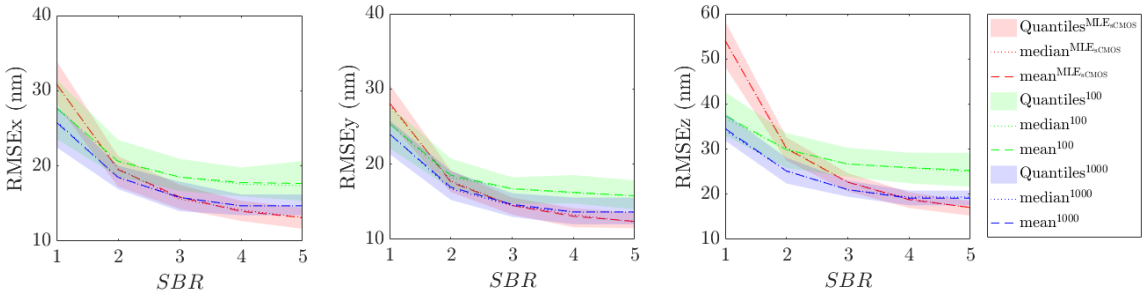


Figure 5·19: True L=0.4 μm: RMSE as a function of SBR.

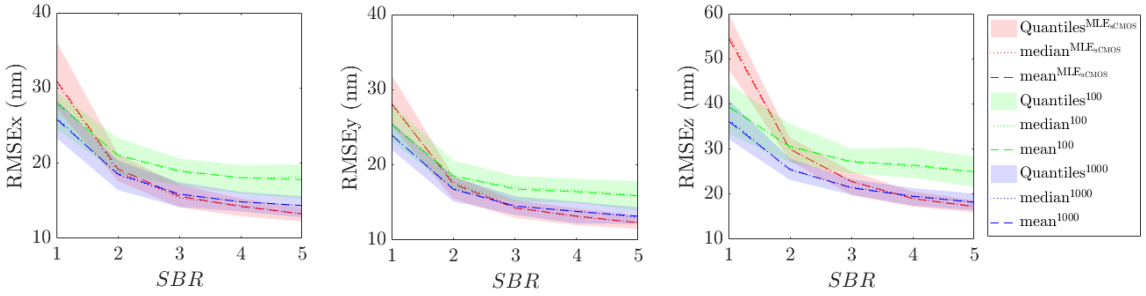


Figure 5·20: True L=0.5 μm: RMSE as a function of SBR.

Fig. 5·21 - 5·23 show the results of the simulation study of parameter estimation as a function of confinement length and Signal-to-Background Ratio (SBR) ranging from 2 - 4. These Fig. together with Fig. 5·11 - 5·12 reveal that, SMC-EM outperforms the other algorithms at the lowest signal level and when the particle is tightly confined. The median of the results is shown as a dotted line and the mean as a dashed line. The shadowed region shows the 10%-90% quantiles. The results are organized from smallest confinment ($L = 0.1$ μm) to largest ($L = 0.5$ μm).

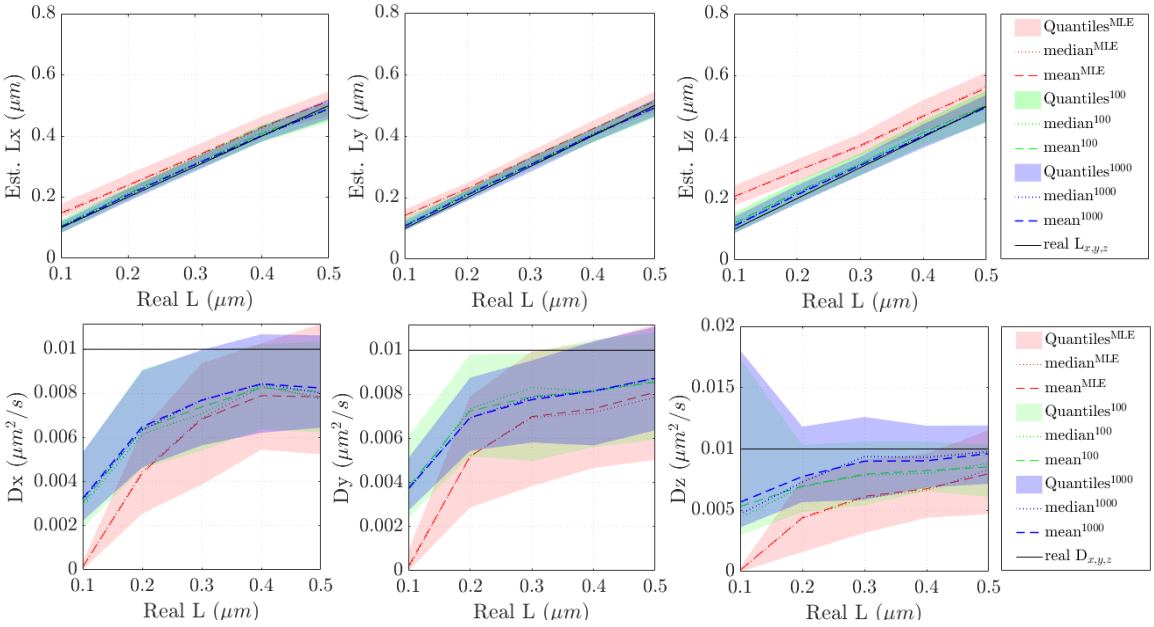


Figure 5·21: SBR=2: parameter estimation as a function of confinement length (true $D_x = D_y = D_z = 0.01$ μm^2/s).

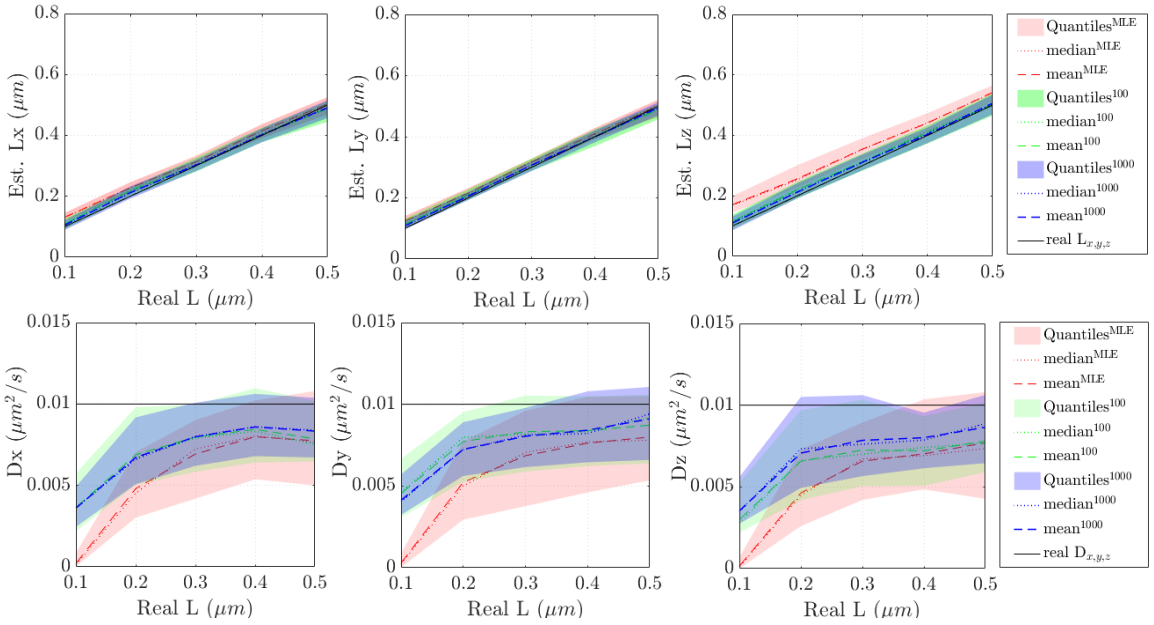


Figure 5·22: SBR=3: parameter estimation as a function of confinement length (true $D_x = D_y = D_z = 0.01$ μm^2/s).

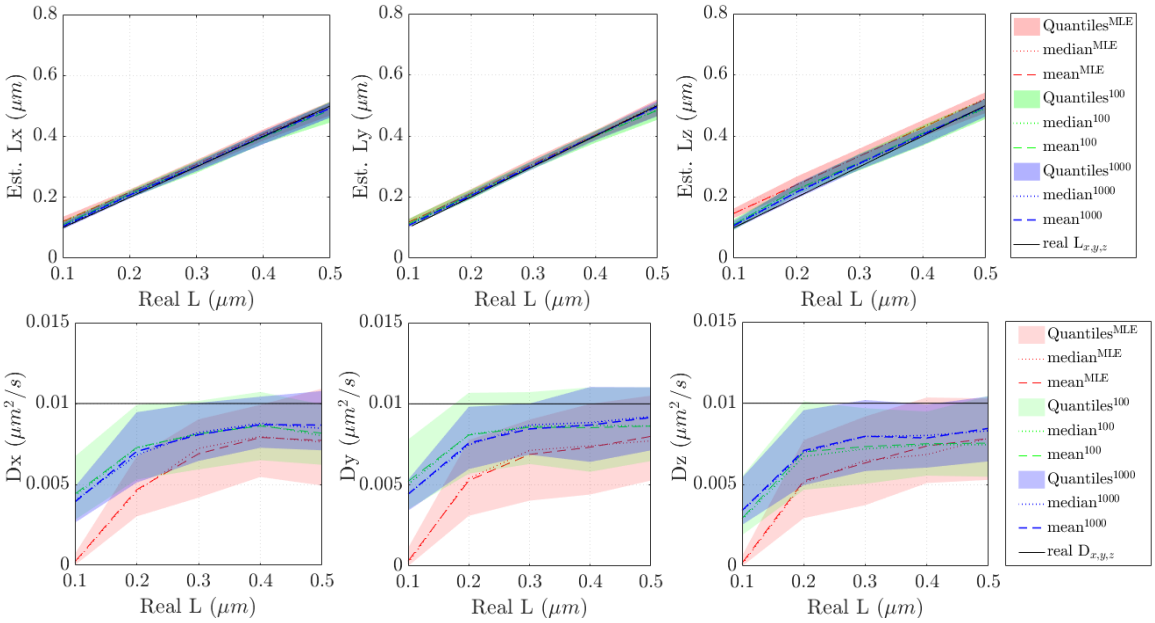


Figure 5·23: SBR=4: parameter estimation as a function of confinement length (true $D_x = D_y = D_z = 0.01$ μm^2/s).

Chapter 6

Particle Identification Network for Particle Detection

In real SPT experiments, the acquired image typically contains multiple particles of interest. Our analysis techniques, however, require a stream of images for a single particle. Our goal in this chapter is to detect those particles in a single image and to extract for each a small sub-image that is suitable for downstream processing, including particle localization and linking across multiple frames into a trajectory. As this is a pattern recognition and classification problem, we apply a machine learning-based approach.

In this work, we first design and create three different Particle Identification Networks (PINs): PIN_{CNN} based on a standard CNN structure, $\text{PIN}_{\text{ResNet}}$ based on a ResNet-50, and PIN_{FPN} based on a FPN. Next all PINs are trained using the same collection of simulated data created with a range of SBRs, emitter density, and different types of PSFs: Born-Wolf PSF, Double-Helix PSF, and Astigmatic PSF. Then a quantitative comparison is conducted among three PINs by testing the same collection of data. In addition, the performance of all PINs is validated using experimental images of labeled AMPA receptors in rat hippocampal neurons at both low and high light conditions.

The primary contributions of this work: (1) the design, training, and testing of deep networks focused on particle detection and image cropping, (2) the comparison of three different ML architectures across a range of signal and background levels,

showing that the complexity of $\text{PIN}_{\text{ResNet}}$ relative to PIN_{CNN} yields clear benefits at very low SBRs while, at least for the PSFs considered, the additional layers of PIN_{FPN} do not improve the performance relative to $\text{PIN}_{\text{ResNet}}$, and (3) validation of the approach on experimental data.

6.1 Problem Definition

In this section, we describe the particle detection problem and the expected output before introducing the three PIN variants. Finally, we discuss the loss function used for training. Note that to keep the description simple, throughout this work we assume the input to the networks to be grayscale images of size 512×512 pixels. The PINs, however, can handle images of other sizes without modification so long as the kernel size is kept constant. We approach the detection task as a classification problem in machine learning; this has a rich and well-developed foundation (see, e.g., (Ketkar and Santana, 2017; Gupta and Sehgal, 2021)).

The input to our networks is a raw, 512×512 pixel image containing fluorescent, sub diffraction-limit sized particles (see Fig. 6·5), though training could also be done using data with non-fluorescent labels such as gold nanoparticles (see, e.g., (Liu et al., 2020) for labeling strategies in SPT) or label-free techniques such as interferometric Scattering Microscopy (iSCAT) data (Taylor and Sandoghdar, 2019). We consider images with three different PSFs (Gaussian, astigmatic, and double helix). Given the typical size of these PSFs in an image, we define the output to be a 64×64 image where the value in each pixel defines the probability that the corresponding 8×8 pixel region in the original image contains a particle. We refer to this output as a heatmap.

Note that at this point, we assume there is at most one particle in any 8×8 region. This is a common assumption in SPT and is typically enforced by proper experimental design. However, overlap can occasionally occur and thus later in this chapter we test

our approach against images with dense collections of particles .

As is common in a classification problem, this output heatmap is converted to a decision as to whether or not there is a particle in the corresponding region by selecting a threshold on the probability; an output below the threshold is a negative (there is no particle in the corresponding region) while a value above the threshold is classified as a positive (there is a particle in the corresponding region). The quality of a threshold is evaluated through two standard metrics, Precision (Pr) and Recall (Re), defined by

$$\mathrm{Pr} \triangleq \mathrm{TP}/(\mathrm{TP}+\mathrm{FP}), \tag{6.1a}$$

$$\mathrm{Re} \triangleq \mathrm{TP}/(\mathrm{TP}+\mathrm{FN}), \tag{6.1b}$$

where TP represents the true positive classifications, FP the false positives, and FN the false negatives. Pr measures the ability of the model to identify only true objects in the class (here, the presence of a particle) as a fraction of all classified elements, correct and incorrect. Re evaluates the ability of the model to find the objects in the data. These two metrics both range from zero to one, representing competing goals; improving performance in one degrades performance in the other. This tradeoff is typically visualized using a precision-recall curve.

To select an appropriate threshold for a network we use the F_1 score, given by

$$\mathrm{F}_1(p) \triangleq \frac{2}{\frac{1}{\mathrm{Re}(p)} + \frac{1}{\mathrm{Pr}(p)}}, \tag{6.2}$$

where p is a particular choice of threshold. Note that F_1 is bounded between zero and one, reaching the lower bound if either the recall or precision goes to zero, and

the upper bound only if both go to one. The threshold, p_{thresh} is then given by

$$p_{\text{thresh}} = \arg\max_{p} \mathrm{F}_1(p). \tag{6.3}$$

It is useful to compare the performance of identification networks independent of any particular choice of threshold. Given a list of N monotonically increasing thresholds, this can be done using the Average Precision (AP), defined as

$$\mathrm{AP} \triangleq \sum_{n=2}^{N} (\mathrm{Re}_n - \mathrm{Re}_{n-1})\mathrm{Pr}_n \tag{6.4}$$

where Re_n and Pr_n are the precision and recall at the n^{th} threshold. (6.4) shows that AP is a weighted sum of the precision of the classifier at each threshold with a weight defined by the corresponding change in the recall achieved by the change in threshold.

After applying the threshold, each 8×8 pixelated region is labeled as either containing a particle or not. However, the particle may be located anywhere within that 8×8 area. To ensure the image of the entire PSF is included for later processing, a 16×16 pixel area is extracted, with the center corresponding to the center of the 8×8 region.

We highlight that unlike previous chapters, we are considering here only a single image; our problem is purely one of detection without any dynamic model information. As one looks across images, however, the same particle may move from one 8×8 region to another; as described further in Ch. 7 a linking step must be performed in order to generate the image sequence needed by our EM-based framework.

6.2 Architecture of Particle Identification Networks

6.2.1 PIN_{CNN}: CNN-based architecture

Inspired by (Newby et al., 2018; Helgadottir et al., 2019) where convolutional neural networks were applied to localize sub-diffraction limit-sized particles in images, our first PIN uses a similar CNN architecture for emitter detection. Different from (Helgadottir et al., 2019) that returns localization only and directly, the PINs return a sequence of cropped images required for EM based framework. Our structure, shown in Fig. 6·1, uses three convolutional layers (denoted as CONV 1/2/3), each with 0 padding of size 1 and a kernel size of 3. After each layer, a Rectified Linear Unit (ReLU) is used as the activation function and max pooling of size two applied to down-sample the input representation and produce the subsequent feature map. After the third convolution layer, a 1×1 convolution is applied to linearly transform the channel size of preceding layer from 16 to 1, then the output is passed through a sigmoid function to return the output with the range 0 to 1. Given a selection of a threshold p_{thresh}, the heat map is converted to a detection map which can be used to crop regions from the original image that contain a particle (this step is labeled as "0/1" in Fig. 6·1). Note that this same thresholding and cropping procedure is used in all three of our PIN architectures.

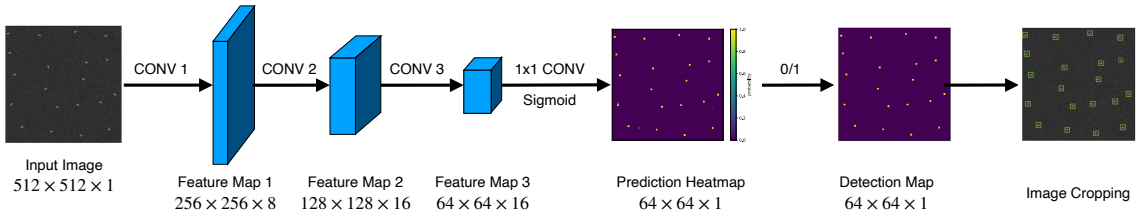


Figure 6·1: Architecture of PIN_{CNN}.

6.2.2 $\text{PIN}_{\text{ResNet}}$: ResNet-50 based Architecture

Inspired by ResNet-50 (He et al., 2016), we constructed $\text{PIN}_{\text{ResNet}}$, a much deeper neural network than PIN_{CNN}. An outline of the architecture is shown in Fig. 6·2. The output of an initial convolutional layer (CONV 1) passes into a *Stride-Reduced ResNet-50* structure made up of four sequential groups of operations highlighted in blue, green, yellow and red with each box indicating an intermediate feature map. (Note that this name follows from the fact that our implementation used a reduced stride step (from two to one) in groups 1, 3, and 4, as compared to the original structure in (He et al., 2016).) The final output of the Stride-Reduced ResNet-50 block is then passed through a 1×1 convolutional layer followed by a sigmoid function to produce the desired heatmap.

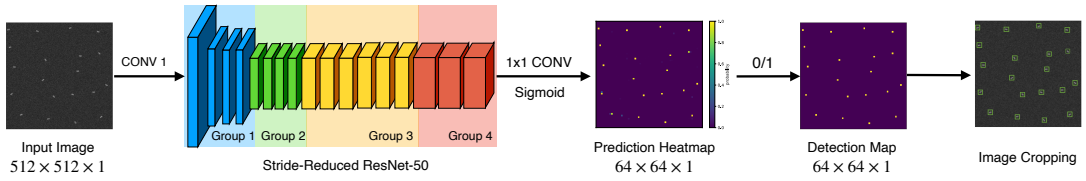


Figure 6·2: Architecture of $\text{PIN}_{\text{ResNet}}$.

6.2.3 PIN_{FPN}: Feature Pyramid Network based Architecture

PIN_{FPN}, first proposed in (Lin et al., 2017a), is a state-of-the-art for detecting objects with features across a large range of length scales. While more complicated than the ResNet architecture, PIN_{FPN} may perform better as PSFs exhibit both fine and coarse features. The basic structure, shown in Fig. 6·3 is built from the Stride-Reduced ResNet-50 block but with additional lateral and vertical connections. The vertical connections up-sample the features to help predict higher resolution features with the lateral connections help to enhance existing features by merging features maps of the

same spatial size. The multiple heatmaps are aggregated into the final heatmap by taking their mean.

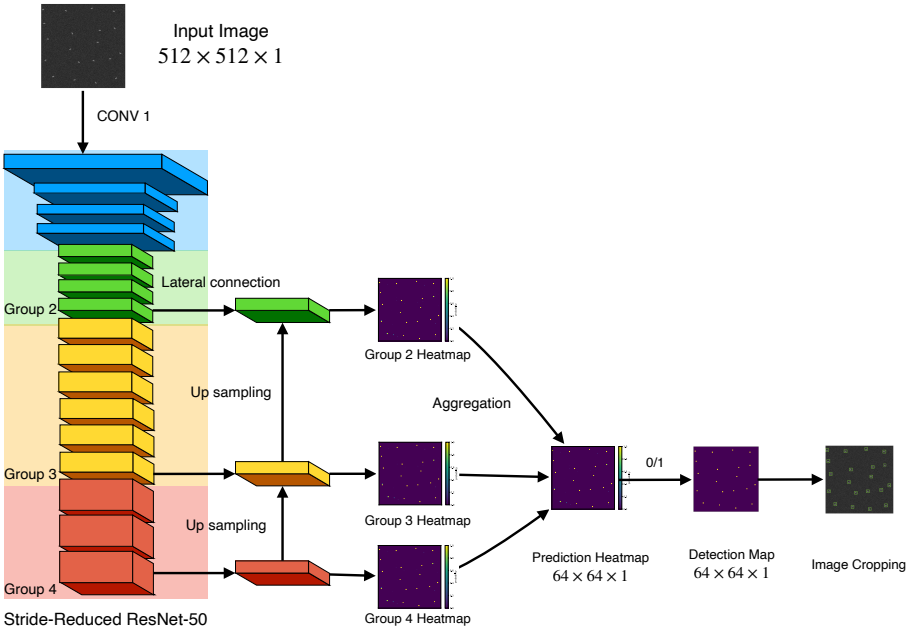


Figure 6·3: Architecture of PIN_{FPN}

From Fig. 6·2 - Fig. 6·3, we see both $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} share the same structure of Stride-Reduced ResNet-50 of which details is shown as Fig. 6·4. The Group 1, 2, 3, and 4 in Fig. 6·2 - Fig. 6·3 correspond to blue, green, yellow and red diagrams in Fig. 6·4 respectively. Note that the color boxes within Stride-Reduced ResNet-50 denote the feature maps, while the color rectangles in Fig. 6·4 denote the convolution layers. The second group is operated with stride of 2, so a downsampling layer is needed to match the dimension. Therefore, the addition of a 1×1 convolution layer with stride 2 is added to group 2, which is denoted by a green dashed and curved arrow. The other three groups are constructed with a stride of 1 and thus do not

need the addition of this 1×1 convolution layer. There are, however, multiple skip connections without a 1×1 convolution layer included as shown by the solid curved arrows between layers. In essence, each group uses a stack of three layers (1×1, 3×3, and 1×1 convolutions respectively). Finally, the output of this sequence of stacks is passed to a 1×1 convolution followed by a Sigmoid function to produce the desired prediction heatmaps. For both $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN}, batch normalization is used prior to the activation function so as to stabilize and accelerate the training process.

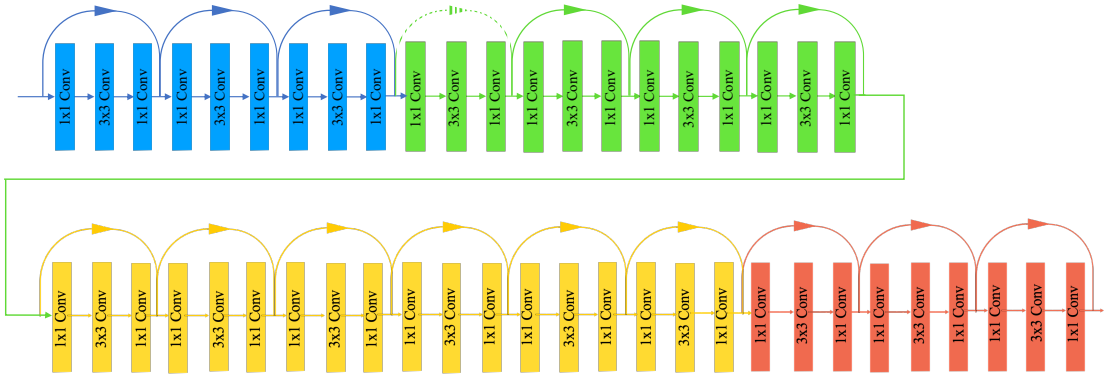


Figure 6·4: Details of the four groups that define the Stride-Reduced ResNet-50. The dashed curved arrow denotes a skip connection with an addition of 1×1 convolution layer with stride 2 while solid curved arrows denote skip connections without any addition of convolution layers.

6.2.4 Loss Function

The proposed PINs were trained with supervised learning using a binary cross entropy loss between the prediction heatmap(s) and the ground truth heatmap on a pixel-by-pixel basis. That is, the loss function was given by

$$\mathcal{L}(\hat{y}, y) = \frac{1}{P} \sum_{p=1}^{P} \left[-y_p \cdot \log \hat{y}_p - (1 - y_p) \cdot \log(1 - \hat{y}_p) \right], \tag{6.5}$$

where P denotes the total number of pixels, $\hat{y}$ the prediction heatmap (with $\hat{y}_p$ the value in the p^{th} pixel), and y the ground truth heatmap (with y_p the value in the p^{th} pixel).

Note that in most fluorescence images, the relative occurrence between positive labels (that is, existence of an emitter) and negative labels (no emitter) are not balanced. To handle such an imbalance, one typically uses a average focal loss.

To take the possible class imbalance problem into consideration, the proposed PINs are trained with the average focal loss (Lin et al., 2017b). While the weighted binary cross entropy is defined as:

$$\mathcal{L}(\hat{y}, y) = \frac{1}{P} \sum_{p=1}^{N} -\alpha \ \log \hat{y}_p - (1-\alpha) \ (1-y_p) \ \log(1-\hat{y}_p), \tag{6.6}$$

where P denotes the total number of pixels, $\hat{y}$ denotes the predicted heatmap, y denotes the ground truth heatmap, $\hat{y}_p$ denotes the predicted result at the p-th pixel, while the y_p denotes the ground truth at the p-th pixel, α is a hyperparameter that balances between positive and negative examples by weighing prediction errors up and down (this will change the precision and recall accordingly). Noting that the weighted binary cross entropy here is slightly different than the widely used representation, where α is the weight on positive examples.

The average focal loss between the predicted heatmap(s) and the ground truth heatmap per pixel is:

$$\mathcal{L}(\hat{y}, y) = \frac{1}{P} \sum_{p=1}^{N} -\alpha \ y_p \ (1-\hat{y}_p)^{\gamma} \ \log \hat{y}_p - (1-\alpha) \ (1-y_p) \ \hat{y}_p^{\gamma} \ \log(1-\hat{y}_p), \tag{6.7}$$

where $\gamma \geq 0$ is a tunable focusing parameter that adjusts the rate where easy-examples are down-weighted. However, this introduces additional hyperparameters that need to be carefully tuned. Both binary cross entropy and average focal loss return similar results, for the ease of parameter tuning, all results for the remaining part are returned

with binary cross entropy.

6.2.5 Implementation Details

The proposed PIN_{CNN}, $\text{PIN}_{\text{ResNet}}$, and PIN_{FPN} are initialized with Gaussian distribution and trained distributedly on 2 NVIDIA Tesla V100 GPUs with an Adam (Kingma and Ba, 2014) optimizer with a learning rate of 0.001 for 10 epochs on the simulated datasets. Due to the different sizes of the proposed PINs, we use batch sizes of 128, 32, and 16 images per GPU respectively for each PIN.

Each image contains 20 emitters (except for data generated to test performance as a function of the number of emitters), all with the same peak intensity. Images were generated at SBRs ranging from 10 to 100, in steps of 10, by increasing the value of G while leaving the background rate fixed at 10. Of all the simulated datasets, 65000 images (65%) were fed to the PIN for training, 15000 images (15%) were used for validation, and 20000 images (20%) were used for evaluation.

6.3 Simulations

To train each of our PINs, and to compare their performance in controlled settings, we used physical simulations. As described below, these simulations include background and camera readout noise with data generated across a range of SBR, including for very weak emitters. Further, we separately trained and tested the networks using data from three different PSFs, namely a standard PSF using the Born-Wolf PSF (BW-PSF), a Double-Helix PSF (DH-PSF) modeled as a pair of Gaussian spots that rotated with the $z-$position of the emitter, and an astigmatic PSF (A-PSF). In this section we describe our image generation process.

We assume the image acquired by the camera is arranged into a 512×512 square array of pixels. The pixel size is Δx by Δy with the actual dimensions determined both by the physical size of the camera elements on the camera and by the magnifica-

tion of the optical system. The intensity generated by the emitters is well described as a Poisson process with the expected photon intensity in the p^{th} pixel arising from an emitter i at position (x_i, y_i) given by

$$\lambda_{p,i} = G \int_{x_p^{min}}^{x_p^{max}} \int_{y_p^{min}}^{y_p^{max}} PSF\left(x_i - \xi, y_i - \xi'\right) \, d\xi d\xi', \tag{6.8}$$

where G denotes the peak signal intensity level, and the integration limits are over the boundaries of the p^{th} pixel. The total expected intensity in a pixel is then just the sum over all contributing emitters. The total expected intensity in this pixel is then just the sum over all emitters,

$$\lambda_p = \sum_i \lambda_{p,i}. \tag{6.9}$$

The background signal arising from out-of-focus fluorescence and sample autofluorescence is also modeled as a Poisson process. For simplicity, we assume a constant expected rate of N_{bgd} photons/pixel over the entire image. Modifying this to vary over different regions in the image is straightforward but we found good performance in experiment even with this simple model. The measured intensity in pixel p is then given by (2.10). To meet the assumption that the PSFs do not overlap, we randomly sampled the initialized emitter positions while ensuring there were at least 10 pixels apart.

6.3.1 Simulation Setup

Recalling the measurement model summarized in (6.8) and (6.9), the point spread function determines the shape of images. In this work, three different PSFs are taken into consideration: a standard PSF using the BW-PSF, a DH-PSF modeled as a pair of Gaussian spots that rotated with the $z-$position of the emitter, and an A-PSF.

Table 6.1 shows the fixed parameter settings that are used for three different point spread functions. The imaging period Δt denotes the time interval between any two continuous images, the image sequence length N denotes the number of images per dataset, each image is of 512×512 pixels, each pixel covers the region of $\Delta x \times \Delta y$ nano-meters. All images are simulated with a constant background noise N_{bgd}. The peak signal intensity G determines the SBR in this work, and it varies from 10 to 100, both NA and λ are physical parameters used for microscopy setup.

Details on specific parameter values, including peak intensity G, background rates, and optical parameters, can be found in Table 6.1.

Table 6.1: Fixed parameter settings for all simulations

Symbol	Parameter	Value
Δt	Imaging period	100 ms
N	Image sequence length	100
Δx, Δy	Effective pixel length	100 nm
N_{bgd}	background noise	10
G	peak signal intensity	10 - 100

In this simulation setup, we consider three different PSF modes (Born-Wolf, Double-Helix, Astigmatic) following (2.19), (2.21), and (2.24) respectively. Typical images formed using the three different PSFs are shown in Fig. 6·5.

6.3.2 Simulation Results

For each PSF considered, 100,000 images were generated. Each image contained 20 emitters, randomly placed in 512×512 pixels. Peak intensities in each image were the same for every emitter, and images were generated with peak intensities ranging from $G = 10$ to $G = 100$ counts with a fixed background rate of 10 counts/pixel.

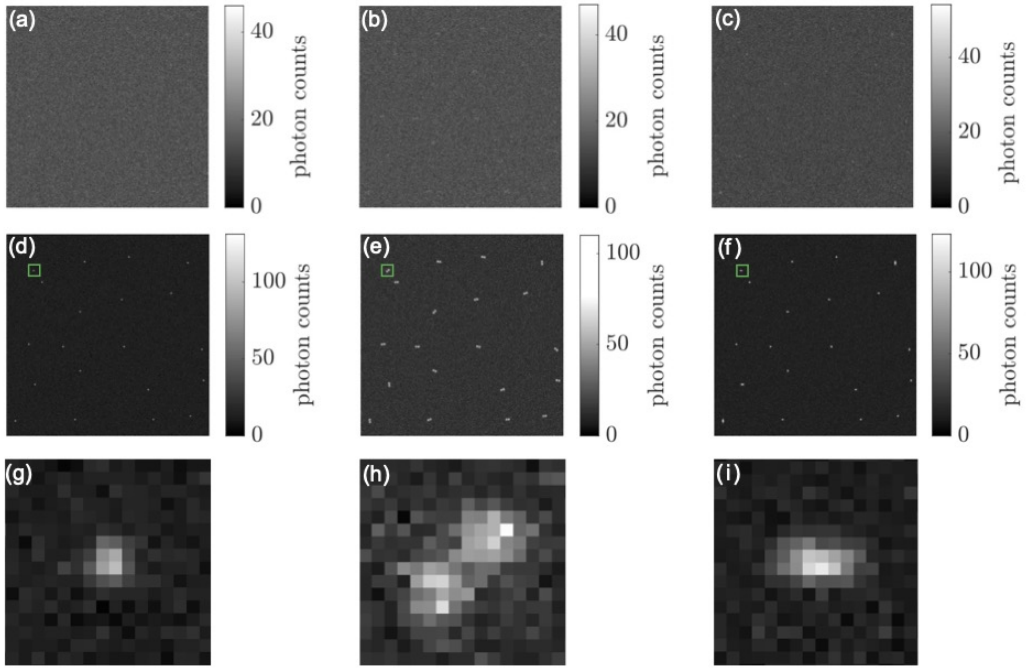


Figure 6·5: Typical simulated images using (left column, (a,d,g)) BW-PSF, (center column (b,e,f)) DH-PSF, and (right column (c,f,i)) A-PSF. and formed at (first row, (a-c) low SBR (SBR =1) and (second row, (d-f)) high SBR (SBR=10); (g-i) cropped regions of the individual fluorescent emitters highlighted with green boxes in the second row.

For the first comparison, we focus on the precision-recall curves of the PINs based on the evaluation datasets. In terms of the detection performance, both $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} outperform PIN_{CNN}, and this difference is especially obvious when SBR$\leq$3, the precision-recall curves support that both PIN_{FPN} and $\text{PIN}_{\text{ResNet}}$ outperform PIN_{CNN} at low SBRs, but all PINs perform well at relative high SBRs. Though $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} return similar results, but $\text{PIN}_{\text{ResNet}}$ have simpler computational complexity. Therefore, of all PINs concerned, $\text{PIN}_{\text{ResNet}}$ is state-of-the-art across a wide range of SBRs. However, for the relatively high SBR levels, we suggested PIN_{CNN} as the priority approach due to its outstanding computational simplicity. It is also clear that performance is higher on the DH-PSF and A-PSF, likely due to the richer structure of these engineered PSFs.

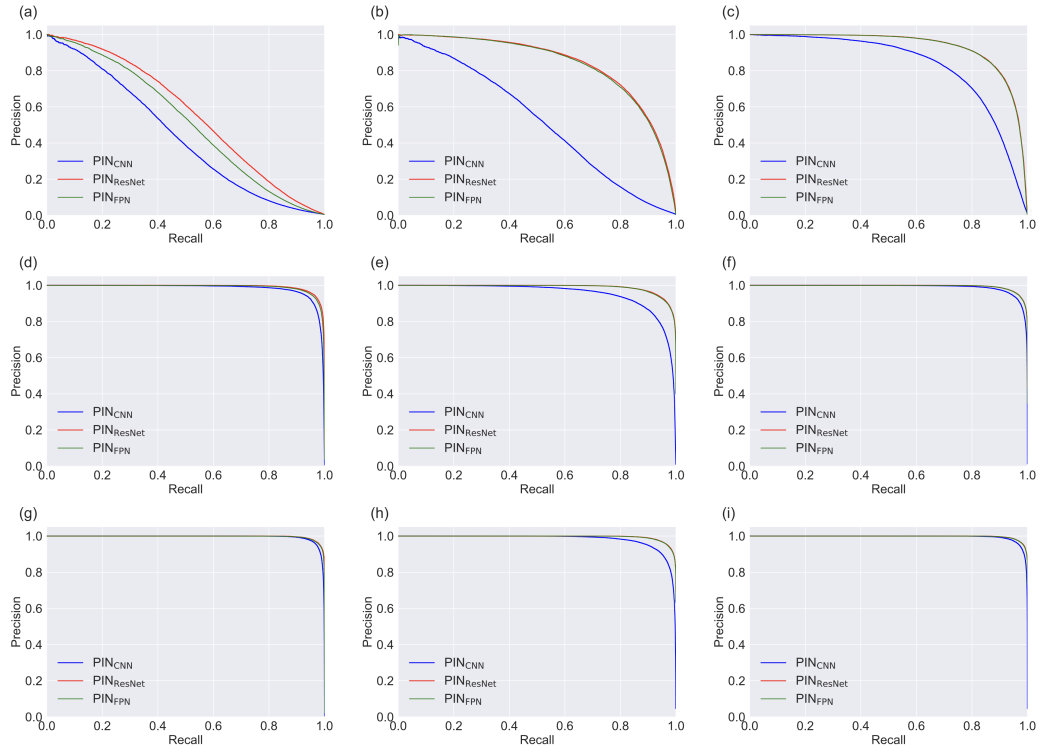


Figure 6·6: Precision - recall curves of PINs on datasets with varied PSFs where SBR=1 for subfigurs (a) - (c), SBR=2 for subfigurs (d) - (f), SBR=3 for subfigurs (g) - (i). The columns from left to right correspond to BW-PSF, DH-PSF, and A-PSF respectively.

For each setting of {PSF type, SBR, PIN method}, we calculated the probability threshold by following (6.2) - (6.3) such that we obtained the largest F_1 score among all potential probabilities. And the probability threshold helps determine whether a sub-region contains a fluorescent emitter or not. With the binary cross entropy, the calculated F_1 scores under different scenarios are listed in Table 6.2.

The detailed average precision scores using the evaluation datasets across all SBRs and for all three PSFs and PINs are listed in Table 6.3, and recapitulated in Fig. 6·7 for the lowest SBRs. Similar to the Precision-Recall curves, $\text{PIN}_{\text{ResNet}}$ outperforms all other methods under all conditions, though the difference from PIN_{FPN} is small. At SBRs larger than 3, all three networks perform extremely well.

Table 6.2: The calculated probability threshold using weights trained with binary cross entropy.

PSF	SBR	1	2	3	4	5	6	7	8	9	10
Born-Wolf	PIN_{CNN}	0.03	0.40	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60
	$\text{PIN}_{\text{ResNet}}$	0.30	0.40	0.40	0.50	0.60	0.50	0.50	0.50	0.50	0.50
	PIN_{FPN}	0.30	0.50	0.60	0.50	0.50	0.60	0.50	0.50	0.50	0.50
Double-Helix	PIN_{CNN}	0.04	0.40	0.50	0.60	0.60	0.60	0.60	0.60	0.50	0.60
	$\text{PIN}_{\text{ResNet}}$	0.40	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
	PIN_{FPN}	0.40	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
Astigmatic	PIN_{CNN}	0.09	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.70	0.70
	$\text{PIN}_{\text{ResNet}}$	0.40	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.40
	PIN_{FPN}	0.50	0.50	0.50	0.50	0.60	0.50	0.50	0.60	0.50	0.50

Table 6.3: The AP scores using weights trained with binary cross entropy.

PSF	SBR	1	2	3	4	5	6	7	8	9	10
Born-Wolf	PIN_{CNN}	0.43574	0.98210	0.99406	0.99605	0.99720	0.99765	0.99780	0.99814	0.99825	0.99829
	$\text{PIN}_{\text{ResNet}}$	0.56356	0.99253	0.99716	0.99830	0.99884	0.99908	0.99926	0.99945	0.99953	0.99955
	PIN_{FPN}	0.51832	0.99006	0.99646	0.99769	0.99850	0.99875	0.99900	0.99930	0.99935	0.99941
Double-Helix	PIN_{CNN}	0.52068	0.95240	0.98112	0.98883	0.99163	0.99318	0.99357	0.99427	0.99424	0.99432
	$\text{PIN}_{\text{ResNet}}$	0.83945	0.98884	0.99514	0.997036	0.99798	0.99835	0.99874	0.99891	0.99912	0.99924
	PIN_{FPN}	0.83404	0.98823	0.99487	0.99689	0.99788	0.99831	0.99867	0.99888	0.99907	0.99921
Astigmatic	PIN_{CNN}	0.88282	0.98906	0.99461	0.99624	0.99702	0.99730	0.99763	0.99770	0.99742	0.99743
	$\text{PIN}_{\text{ResNet}}$	0.93106	0.99443	0.99741	0.99837	0.99882	0.99125	0.99932	0.99939	0.99947	0.99952
	PIN_{FPN}	0.93008	0.99433	0.99733	0.99828	0.99875	0.99904	0.99927	0.999335	0.99942	0.99947

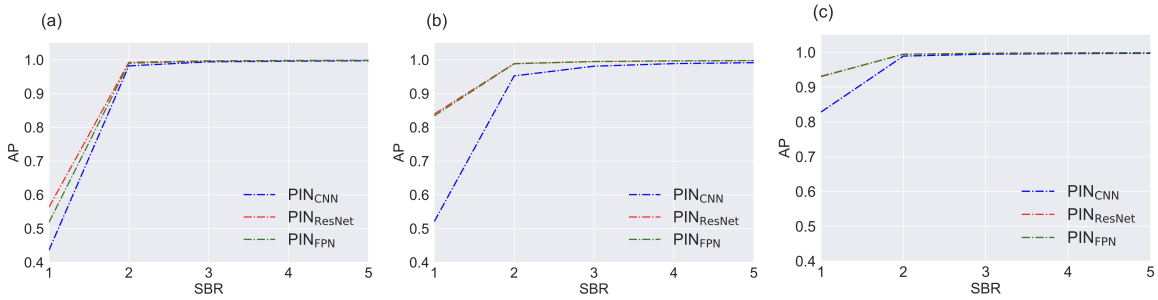


Figure 6·7: Comparison of average precision scores across the lowest SBRs for (a) BW-PSF; (b) DH-PSF; (c) A-PSF.

From these results, we see that the additional complexity of PIN_{FPN} does not bring a concomitant increase in performance relative to $\text{PIN}_{\text{ResNet}}$, at least for the PSFs considered here. Of the three networks considered, $\text{PIN}_{\text{ResNet}}$ is the best choice, particularly at low SBRs. At higher SBRs, PIN_{CNN} performs nearly as well; due to the relative simplicity of its network structure, it is likely a better choice.

Take the images formed by Born-Wolf PSF at SBR = 1 as an example, we dive into the details about detection results shown in Fig. 6·8. At such a low SBR level, PIN_{CNN} returns the smallest number of true positives, both $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} return more True Positives and less False Negatives, though they bring more False Positives as well. For relative high SBR levels (SBR> 3), almost all detected boxes return by PINs are true positives.

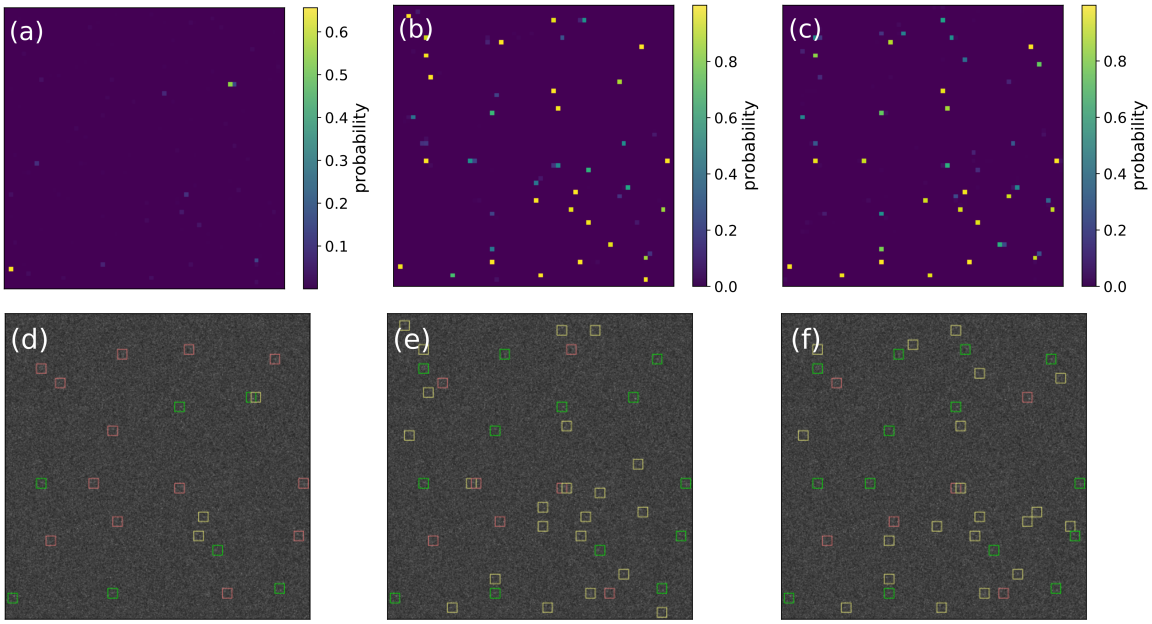


Figure 6·8: Comparison of detection results at SBR=1 where (a) - (c) are 64×64 probability heatmap returned by PIN_{CNN}, $\text{PIN}_{\text{ResNet}}$, and PIN_{FPN} respectively; (d) - (f) are the 512×512 detection results returned by PIN_{CNN}, $\text{PIN}_{\text{ResNet}}$, and PIN_{FPN} respectively; the green, yellow and red boxes denote True Positive, False Positive, and False Negative respectively. All bounding boxes (i.e, extraction area) are of 16×16 pixels.

While all training and evaluation was done using 20 emitters/image, real-world data comes with a range of densities. We therefore tested our trained networks on simulated data with densities ranging from 10 to 50 emitters per image. Since the above simulation results indicate that all three PINs perform similarly at higher SBRs, we focused on an SBR of one where the differences were most clear. For each

PSF and each emitter density, 1000 images were generated and processed using the three PINs. Performance was evaluated using the AP score; results are shown in Fig. 6·9. These results indicate that both $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} outperform the standard CNN network structure across all emitter densities. Interestingly, above 30 emitters/image with the DH-PSF and A-PSF, PIN_{FPN} has a slightly better performance than $\text{PIN}_{\text{ResNet}}$. However, the larger size of the DH-PSF and A-PSF relative to the BW-PSF means that at these higher densities, the engineered PSFs begin to overlap. While the networks considered here are not trained to handle overlapping emitters, it is important to note that deep learning approaches can handle this situation, particularly when working with engineered PSFs that show additional structure (Nehme et al., 2018) and thus it is reasonable to expect that the PINs can also be extended to handle this situation by design.

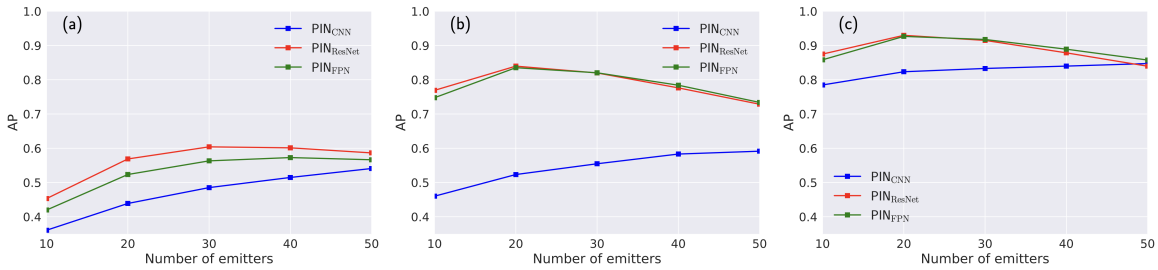


Figure 6·9: Average precision score as a function of emitter densities at SBR = 1 with (a) Born-Wolf PSF; (b) Double-Helix PSF; (c) Astigmatic PSF.

To better understand the performance across emitter densities, we show in Fig. 6·10 - 6·12 the average counts of true positive, false positive, and false negative events using PIN_{CNN}, $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} under each PSF. If these metrics were independent of the number of emitters in the frame, we would expect to to see each grow in a 1:1 ratio with the number of emitters. This is approximately true in the true positives and false negatives at the lowest number of emitters. However, when reducing from 20 to 10 emitters, there is a marked increase in the false positives, indicating that

the network over-identifies particles when the density is lower than what the network was trained at. As the density is increased past 20, the false positive counts stay approximately constant (or even improves for the BW-PSF) while the true positive rate falls off. These results support that, as expected, the network performs best with a number of emitters close to that of the training data. This could likely be improved by training using images with varying numbers of emitters in the dataset.

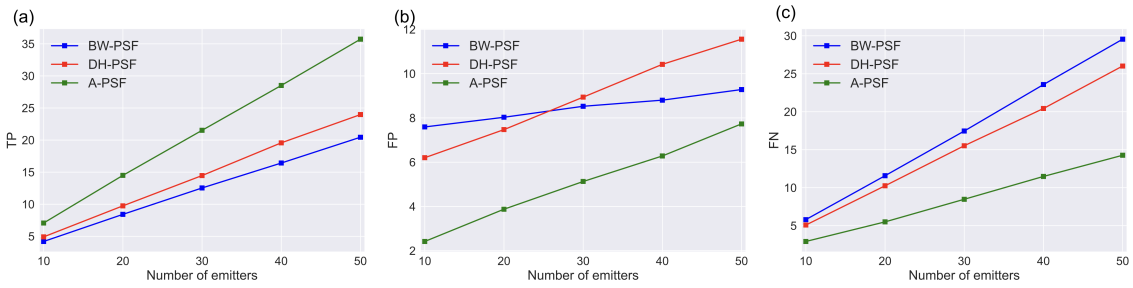


Figure 6·10: Average counts of (a) TP, (b) FP, and (c) FN under three PSFs and across emitter densities using PIN_{CNN}.

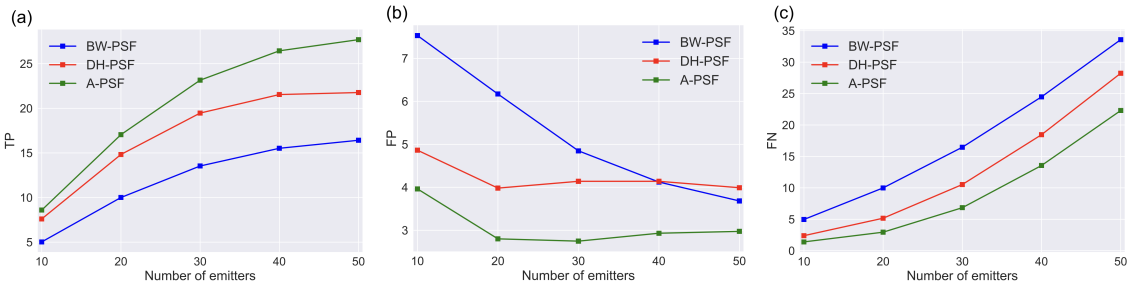


Figure 6·11: Average counts of (a) TP, (b) FP, and (c) FN under three PSFs and across emitter densities using $\text{PIN}_{\text{ResNet}}$.

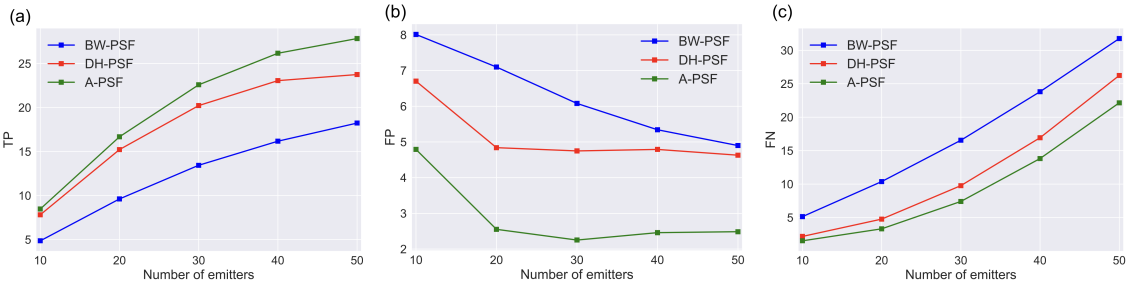


Figure 6·12: Average counts of (a) TP, (b) FP, and (c) FN under three PSFs and across emitter densities using PIN_{FPN}.

6.4 Experiments

6.4.1 Experimental Setups

To validate our approach, we applied our trained PINs to images of fluorescently labeled AMPA recepters in live primary rat hippocampal neurons. Samples were imaged using a 100x, 1.4 N.A. oil immersion objective (Zeiss Plan-Apochromat) and data was acquired using a scientific CMOS (sCMOS) camera (Prime 95B, Teledyne-Photometrics) under different levels of excitation intensity to produce data sets at low (SBR ≤ 3), medium ($3 <$ SBR ≤ 6), and high (SBR > 6) light conditions.

Images were acquired using a Zeiss Axiovert 200 inverted fluorescence microscope. This was equiped with fluorescent filter cubes from Chroma, and a scientific CMOS camera from Teledyne-Photometrics model Prime 95B. Contextual images were taken using Zernike phase contrast imaging with illumination light in the red spectrum to avoid exciting the fluorescent labels.

Neurons were prepared from embryonic day 18 rat embryos as previously described (Hou et al., 2008). Dissociated hippocampal neurons were seeded onto Poly-L-Lysine coated coverslips in 60mm dishes. Neurons were maintained in Neurobasal medium (Thermo Fisher Scientific) and supplemented with 2% Neurocult SM1 Neurobasal medium (Thermo Fisher Scientific), 1% Horse Serum (Atlanta Biologicals), L-glutamine (Corning), and 1% penicillin/streptomycin (Corning). One week after seeding, 10μM 5'-fluoro-2'-deoxyuridine (Sigma-Aldrich) was added to the neuron media to suppress glial cell growth. On day in vitro (DIV) 15, neurons were incubated with 1:200 GluA1 N-terminal antibody (Neuromab) and 1:200 655nm emission quantum dots conjugated with F(ab')2 anti-mouse IgG (Invitrogen) for 10 min at 37^{o}C. The incubation medium was then removed, and neurons were rinsed twice untreated feeding media prior to being maintained in artificial cerebrospinal fluid (ACSF) for imaging. The typical images at each setting are shown in Fig. 6·13.

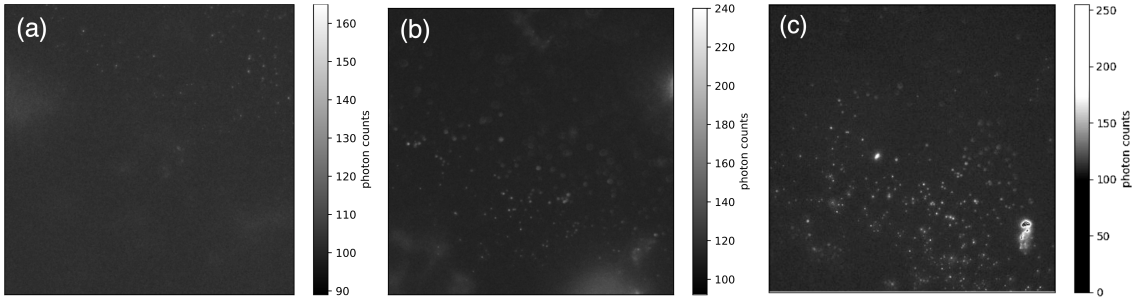


Figure 6·13: Examples of experimental images where (a) - (c) are at low, medium and high SBR respectively. The brightness contrast have been adjusted for easier visualization.

6.4.2 Experimental Results

In this section, we describe typical results on the experimental data under the different conditions. Results on an image at low, medium, and high SBR are shown in Fig. 6·14 - 6·16 respectively. While there is no ground truth to allow for quantitative evaluation, by visual inspection PIN_{CNN} clearly misses many particles that both $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} detect, though these latter two also have a higher false positive rate. In the image shown as Fig. 6·14, PIN_{CNN} detected 2 particles, $\text{PIN}_{\text{ResNet}}$ detected 86 particles, and PIN_{FPN} detected 90 particles.

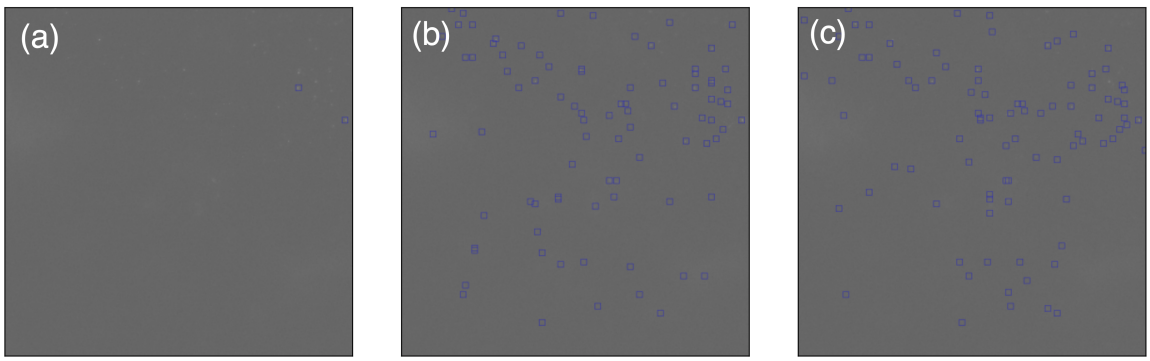


Figure 6·14: Typical experimental results at low SBR using (a) PIN_{CNN}, which detected two particles, (b) $\text{PIN}_{\text{ResNet}}$, which detected 86 particles, and (c) PIN_{FPN}, which detected 90 particles.

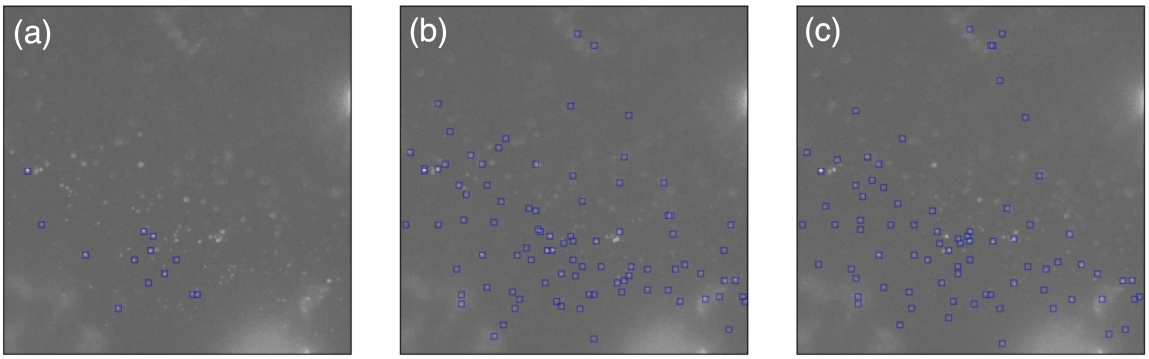


Figure 6·15: Example results of the medium SBR experiment by PINs where (a) detection map by PIN_{CNN}; (b) $\text{PIN}_{\text{ResNet}}$; (c) PIN_{FPN}.

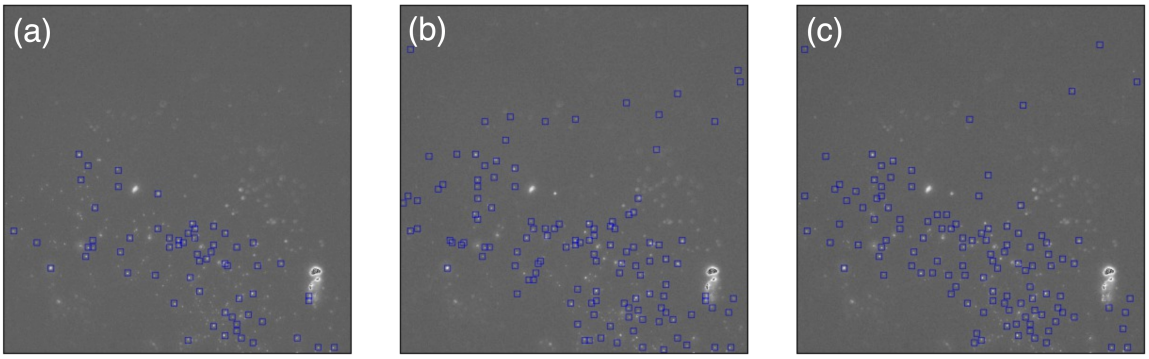


Figure 6·16: Example results of the high SBR experiment by PINs where (a) detection map by PIN_{CNN}; (b) $\text{PIN}_{\text{ResNet}}$; (c) PIN_{FPN}.

In terms of detection performance, both $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} return more detected emitters than PIN_{CNN}. A visual inspection of these images supports the statement that $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} outperform PIN_{CNN}, though without ground truth this is a qualitative statement. Since we use the estimated parameters trained from PINs on simulated datasets, we can only detect fluorescent emitters that are described by the specific type of PSF used in training. Take Fig. 6·16 as an example, though some bright spots have very clear appearances, but they are not formed by Born-Wolf PSF, therefore the PINs cannot detect them.

Fig. 6·15 shows the results on an image with particles at a medium SBR. Once again, both $\text{PIN}_{\text{ResNet}}$ and PIN_{FPN} detect many more particles than PIN_{CNN} with the

standard CNN architecture finding 13 particles, the ResNet 96, and the FPN 79. Fig. 6·16 shows the results in the high SBR setting, and the results are similar to those medium SBR images with PIN_{CNN} detecting 63 particles, $\text{PIN}_{\text{ResNet}}$ 105, and PIN_{FPN} 117. Notice that in both Figs. 6·15 and 6·16, many of the clearly visible bright spots are ignore by all of the PINs. This is driven primarily by the fact that their shapes do not match that of the corresponding PSF. The deep networks are thus both robust against noise in the image but also brittle against differences between the training data and the data in actual application.

6.5 Summary

In this work we considered three different deep neural network architectures for detecting particles in fluorescence microscopy images. These architectures were trained, validated, and tested using simulated data for three different PSFs and across a range of SBRs. The results indicate that all architectures perform well at high SBRs but when the SBR is very low, the additional complexity of the residual network and feature pyramid network architectures provides a benefit. These results indicate that for most settings, $\text{PIN}_{\text{ResNet}}$ provides a good balance of complexity and performance, at least for the PSFs considered. The networks were then tested against data with varying numbers of particles. Finally, $\text{PIN}_{\text{ResNet}}$ was demonstrated on experimental data and connected to an existing localization and parameter estimation algorithm to verify the method produces results that are suitable for downstream analysis.

Chapter 7

Expectation Maximization Combined with Particle Identification Networks for Particle Detection, Extraction, and State Estimation

7.1 Generic Framework

The goal of this chapter is to combine the PINs for particle detection and extraction with our EM-based estimation framework to create an end-to-end scheme to produce trajectories and parameter estimates for multiple particles in an image sequence. The entire process is shown as Fig. 7·1 where a blue background is used to indicate our core algorithms and a gray background is used to highlight outputs.

Throughout this chapter we apply the method across a range of SBRs and over different PSFs. Based on the results of Chapter 6, we focus on $\text{PIN}_{\text{ResNet}}$ as our image detection and extraction tool. Based on the results of Chapter 5, we use the computationally efficient version of SMC-EM as the system identification method. Throughout we use three dimensional, isotropic Brownian motion with $D_x = D_y = D_z = 0.01$ $\mu\text{m}^2/\text{s}$ as the underlying particle dynamics.

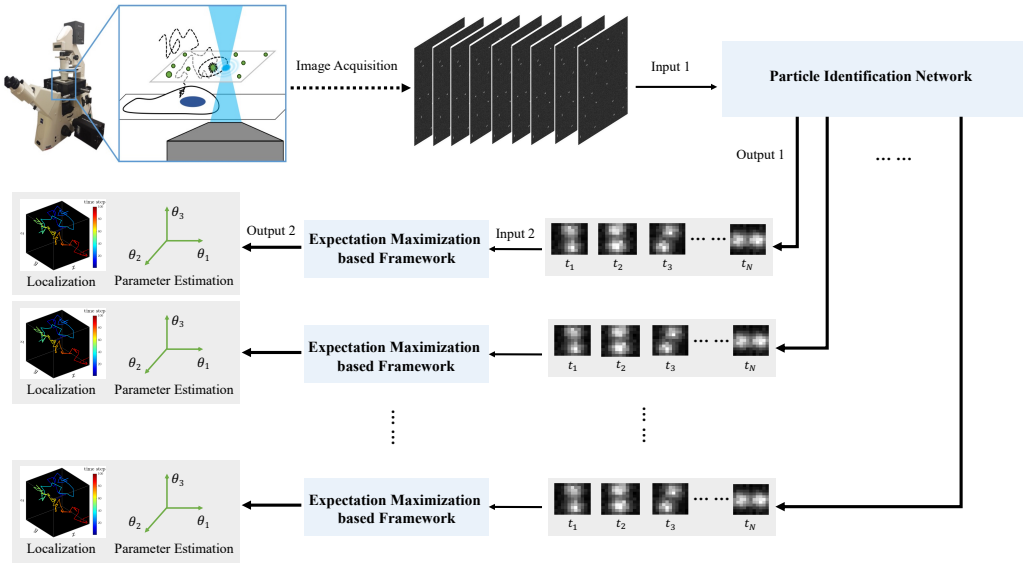


Figure 7·1: Combined PIN and EM-based estimation. The cropped images produced by the PIN are linked using nearest neighbor association into a set of image sequences, one for each particle in the original data. These sequences are then fed into our EM-based estimation approach for trajectory and parameter estimation.

One new challenge in the general setting is how to initialize the particle filters. If those initial locations are too far away from the particle, the SMC filters and smoothers will fail. In this work, when using the BW-PSF and A-PSF, we initialize our particles at the center of the pixel showing peak counts. For the DH-PSF, we found the pixel of peak intensity for each of the two lobes, and initialized at the midpoint. We have found through simulations that this is sufficient, though one could certainly combine our method with a coarse localization using, for example, a Gaussian fit or even an MLE-based method.

7.2 Simulations

In the application demonstrations, we validate the performance of the combination method based on simulations with three different PSF types (BW-PSF, DH-PSF,

A-PSF) at low (SBR=3) and high (SBR=10) light conditions respectively.

We first fed raw images of 512-by-512 pixels to $\text{PIN}_{\text{ResNet}}$, using the training weights from Chapter 6, and used nearest-neighbor linking to obtain multiple sequences of extracted images of 16-by-16 pixels. Note that in this work, we assume the emitters are sparse in each image to ensure that their corresponding PSFs do not overlap. As a result of this assumption, simple nearest-neighbor linking is sufficient to build trajectories unambiguously. To keep our results simple, we randomly picked 10 of these sequences to pass along to SMC-EM for localization and parameter estimation.

7.2.1 Image with Born-Wolf PSF

Because the BW-PSF is symmetric about the optical plane (see (2.19)), we focus here only on estimation in the $x-y$ plane. The mean and standard deviation of estimates by SMC^{100}-EM are listed in Table 7.1. Fig.7·2 and Fig.7·3 show the estimated diffusion coefficients as a function of EM iterations at low and high SBR respectively. As expected, we find good performance in both settings with a larger dispersion of estimate values in the low SBR setting relative to the high SBR condition.

The localization performance report in Table 7.1 shows excellent localization under both high and low SBR settings. Both are well below the Rayleigh criterion, (4.9), for the resolution in a optical image.

Table 7.1: Mean and standard deviation of estimates for images with BW-PSF by SMC^{100}-EM.

SBR	Dx ($\mu m^2/s$)	Dy ($\mu m^2/s$)	RMSE_x (nm)	RMSE_y (nm)
low	0.009535 ± 0.001578	0.009445 ± 0.001505	21.099798 ± 5.212201	20.419784 ± 2.420950
high	0.009610 ± 0.001237	0.008913 ± 0.001361	12.267687 ± 2.325187	12.778948 ± 1.789141

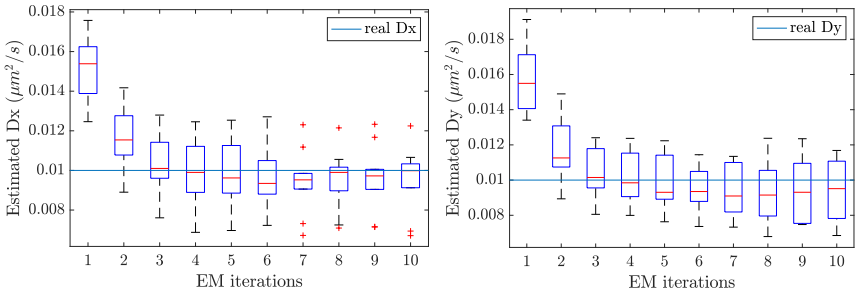


Figure 7·2: Low SBR for BW-PSF: Diffusion coefficient estimates by SMC-EM^{100}.

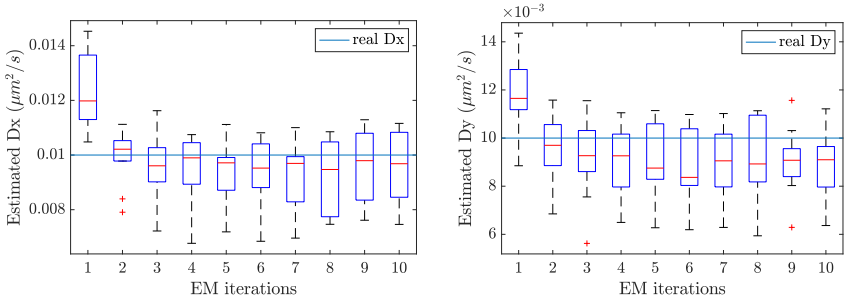


Figure 7·3: High SBR for BW-PSF: Diffusion coefficient estimates by SMC-EM^{100}.

The typical trajectory estimation results at both low and high SBRs are shown in Fig. 7·4. The estimated trajectory closely follows the true trajectory under both low and high light conditions.

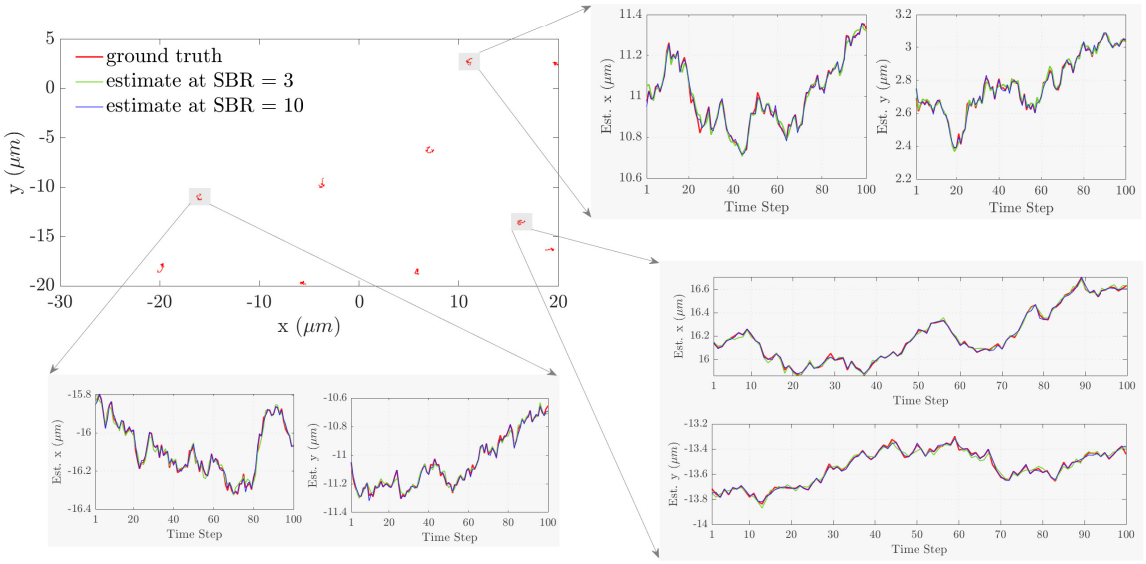


Figure 7·4: Localization performance on simulation dataset by SMC-EM^{100} where the top left figure shows the global trajectories of 10 sequences of particles in a same image, while the shadowed figures presents details of the trajectory estimates under both low and high SBR levels. The time interval between two continuous time steps is 0.1 seconds.

7.2.2 Image with Double-Helix PSF

Because the DH-PSF encodes information about all three axes, we report performance on localization and estimation in x, y, and z. The mean and standard deviation of estimates using SMC^{100}-EM are listed in Table 7.2. As expected, estimation performance is good, though all diffusion coefficients show a small negative bias. Localization precision is again well under the Rayleigh limit.

Table 7.2: Mean and standard deviation of estimates for images with DH-PSF by SMC^{100}-EM.

SBR	Dx ($\mu m^2/s$)	Dy ($\mu m^2/s$)	Dz ($\mu m^2/s$)	RMSE$_x$ (nm)	RMSE$_y$ (nm)	RMSE$_z$ (nm)
low	0.008667 ± 0.001994	0.008001 ± 0.001648	0.008001 ± 0.008131	33.974816 ± 5.768505	29.786391 ± 12.645372	50.525915 ± 19.526161
high	0.008985 ± 0.001631	0.008245 ± 0.001509	0.008245 ± 0.001349	18.866758 ± 2.752270	17.136534 ± 5.457885	31.785005 ± 7.145085

Both Fig.7·5 and Fig.7·6 show the estimated diffusion coefficients as a function of EM iterations at low and high SBR respectively, revealing that, with 100 Monte Carlo particles, the diffusion coefficient estimates at either low or high SBR are steadily

converging to real value. From the results in Chapter 4 on the quantitative analysis of SMC-EM with different number of Monte Carlo particles, we expect to see a smaller bias for estimates when using a larger number of Monte Carlo particles in SMC-EM, at a corresponding cost in computation time.

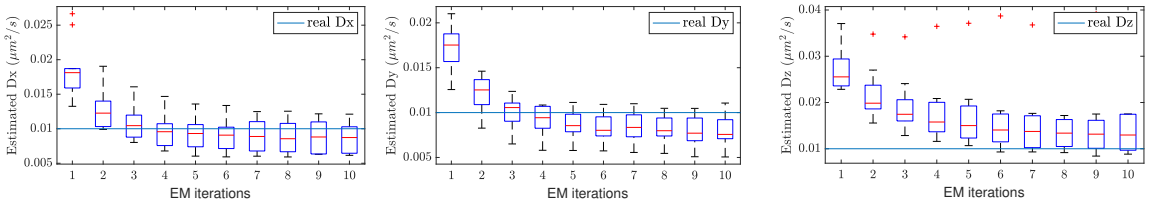


Figure 7·5: Low SBR for DH-PSF: Diffusion coefficient estimates by SMC-EM^{100}.

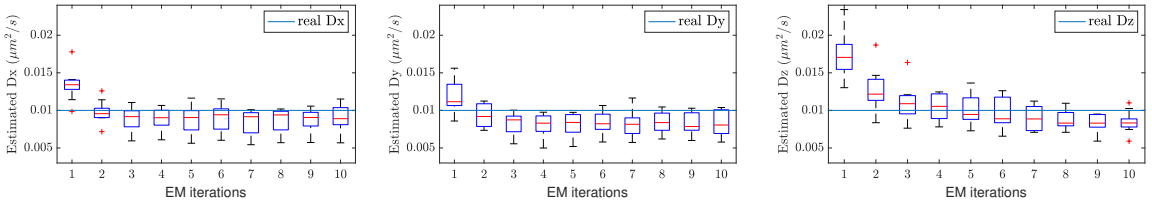


Figure 7·6: High SBR for DH-PSF: Diffusion coefficient estimates by SMC-EM^{100}.

7.2.3 Image with Astigmatic PSF

The A-PSF also encodes information about the axial position of the particle and we therefore again report performance in all three directions. The mean and standard deviation of estimates by SMC^{100}-EM are listed in Table 7.3. Interestingly, performance is slightly worse in terms of localization error than with the DH-PSF (and strikingly poor in the z-direction at low SBR). This is likely due to the fact that the A-PSF does not have as much change in its output signal as a function of z as the DH-PSF does. For more details about the relative performance of DH-PSF and A-PSF, see (Badieirostami et al., 2010). Interestingly, the diffusion coefficient estimation remains good despite the poor performance in localization. The diffusion

coefficient estimation as a function of EM iterations are presented in Fig. 7·7 and Fig. 7·8.

Table 7.3: Mean and standard deviation of estimates for images with A-PSF by SMC^{100}-EM.

SBR	Dx ($\mu m^2/s$)	Dy ($\mu m^2/s$)	Dz ($\mu m^2/s$)	RMSE$_x$ (nm)	RMSE$_y$ (nm)	RMSE$_z$ (nm)
low	0.011147 ± 0.001965	0.008960 ± 0.001372	0.008960 ± 0.041199	25.663304 ± 12.269741	28.339670 ± 17.970637	277.096174 ± 151.606747
high	0.009613 ± 0.001590	0.008419 ± 0.001018	0.008419 ± 0.003914	13.732770 ± 4.258777	16.426717 ± 5.830601	61.235463 ± 13.842484

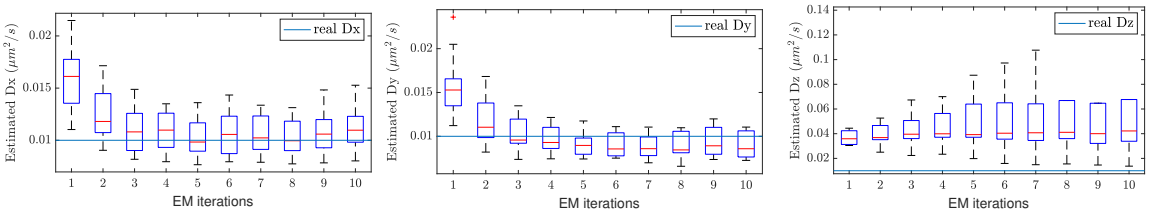


Figure 7·7: Low SBR for A-PSF: Diffusion coefficient estimates by SMC-EM^{100}.

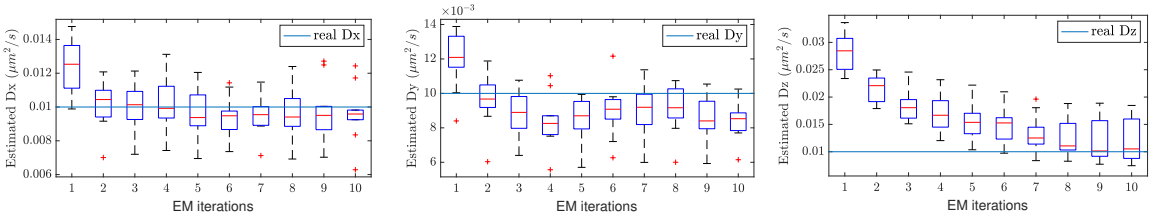


Figure 7·8: High SBR for A-PSF: Diffusion coefficient estimates by SMC-EM^{100}.

7.3 Experiments

Since real experimental data lacks known ground truth, we cannot report on quantitative performance. Moreover, each particle in an image is moving under a different motion model, with different parameters and thus cannot be used to generate statistics on the estimates. Finally, the emission intensities of each particle differ and thus we expect different localization performance across particles. Nevertheless, it is important to demonstrate that our techniques work on real experimental data. To that end, we present a qualitative demonstration using the experimental data of labeled

AMPA receptors in primary rat hippocampal neurons to demonstrate the effectiveness of our method on real biological data. Details on the experimental setup can be found in Chapter 6, together with typical experimental images (Fig. 6·13).

7.3.1 Results

While $\text{PIN}_{\text{ResNet}}$ identifies multiple particles in each image, to keep things simple we selected two of the sequences at low SBR and two at high SBR. Fig. 7·9 and Fig. 7·10 give the example of image extraction from experimental datasets under two two different light conditions. These clearly show the difference in image quality arising from the change in signal level. Note that in these images, the brightness and contrast were adjusted for visualization; these were left unmanipulated in the data used for estimation. In the blow-out images a constant offset was removed to shift the average background rate (from regions with no particle) to an average of 10 counts.

Given the experimental setup, the PSFs are well-described using the Born-Wolf model. Further, we know the AMPA receptors are moving in/near the synaptic cleft between two neurons, which is a small domain on the order of 100 nm in linear dimension. We therefore select a confined diffusion motion model with each axis confined to a domain $[-L/2, L/2]$. For simplicity we take L to have the same value for each axis. The one-step transition density in a single axis is given by (2.8). Note that the estimation takes place in a local coordinate frame.

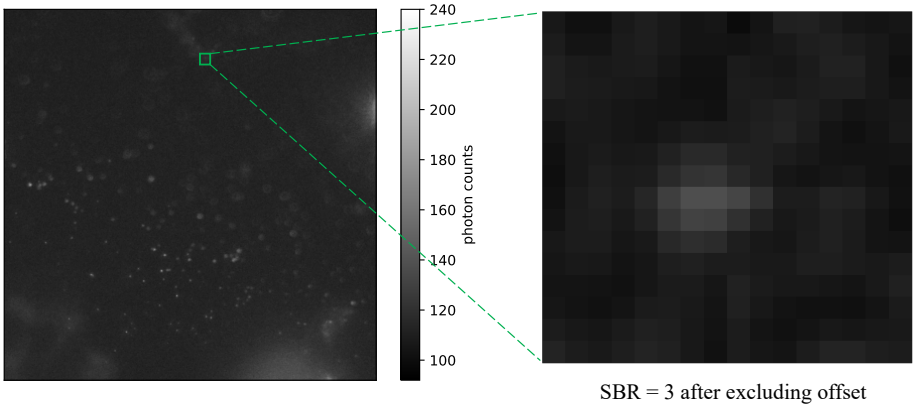


Figure 7·9: Example of image extraction at low SBR from an experimental dataset. The image on the left is the experimental image of which brightness and contrast have been adjusted for easier visualization; the image on the right is the extracted image after excluding offsets.

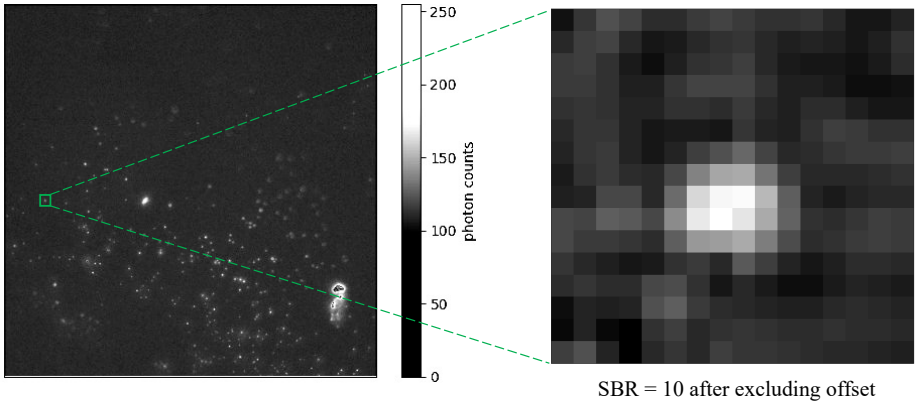


Figure 7·10: Example of image extraction at high SBR from an experimental dataset. The image on the left is the experimental image of which brightness and contrast have been adjusted for easier visualization; the image on the right is the extracted image after excluding offsets.

After running the extracted images through SMC-EM[100], we shift the coordinates back to the global frame of the original image. The two trajectories are shown in Fig. 7·9 for the particle at low SBR and in Fig. 7·10. The trajectories clear

show hallmarks of confinement in both axes. The estimated parameter values were L_x =0.081056 μm, L_y = 0.101468 μm, D_x = 0.010050 μm/s^2, D_y = 0.010050 μm/s^2 in the low SBR setting and L_x =0.100115 μm, L_y = 0.091620 μm, D_x = 0.010050 μm/s^2, D_y = 0.010050 μm/s^2 in the high SBR setting. These trajectories are superimposed back on one of the original images for context in Fig. 7·11 (low SBR) and 7·12 (high SBR).

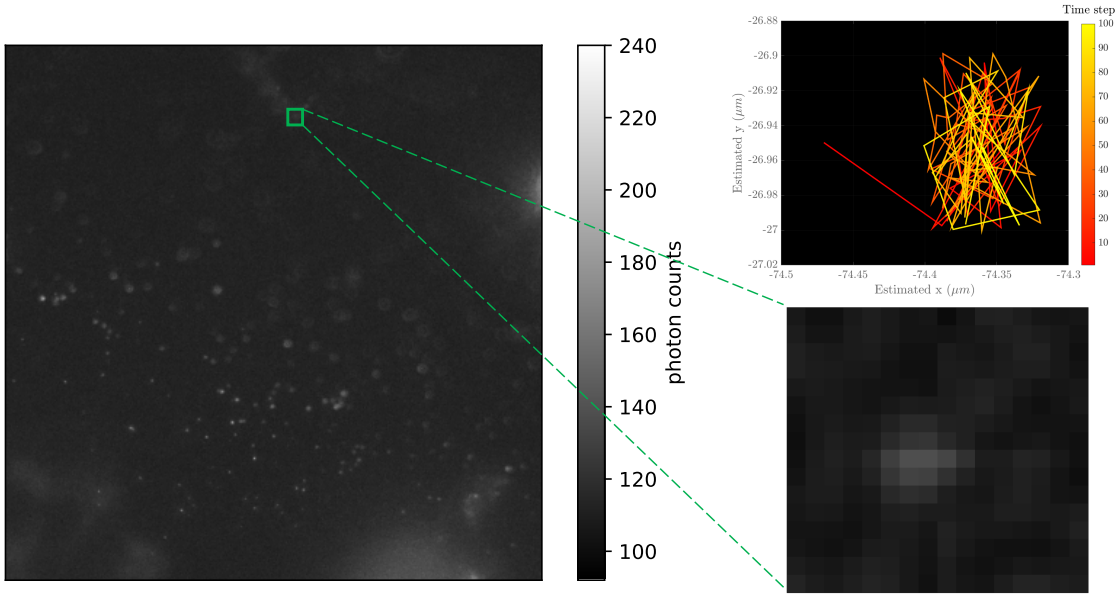


Figure 7·11: Experimental data at low SBR: Trajectory estimation at low SBR level by SMC-EM^{100}.

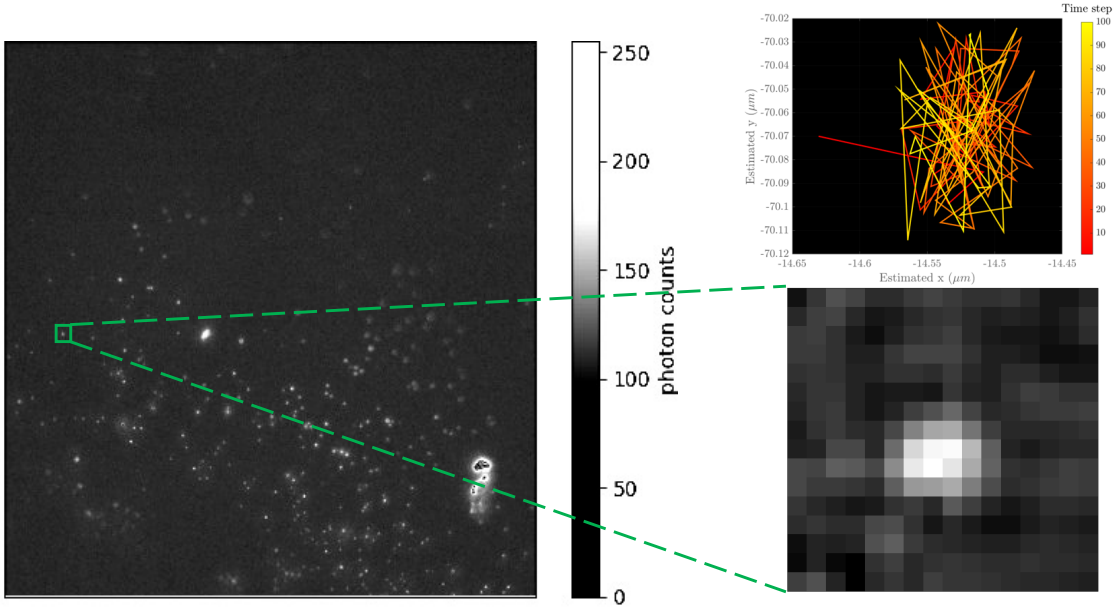


Figure 7·12: Experimental data at high SBR: Trajectory estimation at high SBR level by SMC-EM^{100}.

Chapter 8

Conclusions

8.1 Summary of the Thesis

This thesis presented three sets of contributions centering on the goal of automatic particle detection, extraction, and joint state and parameter estimation in single particle tracking microscopy. In the first set of contributions, we focused on tools to solve the problem of joint localization refinement and physical parameter estimation. In the second set of, we made use of modern machine learning techniques to handle the particle detection and image extraction step to complete the initial step in preparing for SPT analysis. In the last set of contributions, we combined these two sets together to get an end-to-end algorithm that take in raw image data and produces joint localization and parameter estimation for multiple sequences of images each of which contains a single particle.

In terms of joint localization refinement and physical parameter estimation, the novelty of our work lies in two distinct areas. First, a generic EM-based framework is created for localization and parameter estimation is created, allowing for a variety of filtering and smoothing algorithms to be applied that make different trade-offs between computational complexity and performance, allowing the user to select methods that are best suited for their particular data. Through extensive simulation studies, we showed that this approach outperforms the existing state-of-the-art, particularly in the low signal, high background setting which is common in SPT. While the framework can support many different algorithms, we focused on U-EM as a high

performing, low computation choice and SMC-EM for tackling models with stronger nonlinearities which can confound the unscented methods of U-EM. In the class of SMC methods, we introduced the use of computationally efficient particle filtering and particle smoothing algorithms, allowing our methods to be applied to longer data sets, significantly extending their utility to the biophysical community. Finally, we also showed the importance of carefully considering each computational step, simplifying both the motion and measurement models via mathematical approximations to further improve the computational performance. To validate the availability of our EM-based framework, we conduct quantitative analysis among a variety of scenarios including different motion models, camera types, PSFs, SBRs, confinement lengths, diffusion speeds, motion blurs, data lengths.

In terms of particle detection and extraction, the novelty of our work is in the application of modern deep learning network structures to address the problem of particle detection and image extraction. We created three different Particle Identification Networks (PINs): PIN_{CNN} based on a plain CNN, $\text{PIN}_{\text{ResNet}}$ based on a ResNet-50, and PIN_{FPN} based on Feature Pyramid Network. We trained these neural network architectures using simulated datasets, and applied the trained weights to good effect on real experimental datasets. All PINs are evaluated and analyzed under different illumination conditions, different numbers of particles, and different PSFs.

Last but not least, we linked $\text{PIN}_{\text{ResNet}}$ together with our EM-based framework to make an entire procedure for automatic particle detection, extraction, and state estimation. We validated performance through simulations and using biological data.

8.2 Future Directions

Despite the advances made by the work presented in this thesis, there still remain many challenges and open questions to some applications.

8.2.1 Motion Model Identification

Applying the EM based framework requires us to have prior knowledge of the type motion model to estimate and measurement model being used. While it is not unreasonable to assume the user has significant domain knowledge to help guide this choice, it would be better to have automatic model selection, or even use a Bayesian formulation to produce estimates for multiple models and probabilities on those models being good predictors for the data. The choice of model influences not only the most appropriate choice of filtering and smoothing algorithms, but also the biological relevancy of the result.

8.2.2 Sample Initialization for Sequential Monte Carlo

When implementing SMC filters and smoothers, the initial choice of particles (that is, the initial sampling distribution) has a strong effect on the quality of the final estimate. In fact, if the initial particles are too far away from the true location of the particle, there may not be enough information in those locations of the data yield reasonable results. Therefore, it is important to find a reliable way to select this initial distribution. In this work, we used a very direct and simple way to select a rough localization as the initial samples, i.e., the center of the pixel with the maximum number of photon counts in the BW-PSF and A-PSF and the related form in the DH-PSF. This has proved to be a very reasonable approach for the planar coordinates. However, determining and appropriate initial distribution for z is much more challenging. To make these techniques most useful in real data with engineered PSFs, it is important to determine a more accurate way to initialized those samples.

8.2.3 Extend the Comparison Scope of Particle Identification

We have created three different neural network architectures to do particle detection and image extractions, and the results via both simulation and experimental data

validate their performances across a variety of light conditions and emitter densities. However, it still worth to extend the comparison scope of particle identification, for example, using TrackMate, human-eye detection, cross correlation, or template matching.

8.2.4 Optimal Design of Point Spread Function

Our results show that the choice of PSF has a strong effect on the localization and estimation performance. While there has been work on design of engineered PSFs, they have been focused on optimizing localization performance. For SPT data, though, the features of interest are both the locations and the model parameters. Optimal design of PSFs for estimating these parameters, even to the point of tuning them to the specific estimation algorithm being used, could have a significant impact on the field.

References

Abraham, A. V., Ram, S., Chao, J., Ward, E., and Ober, R. J. (2009). Quantitative study of single molecule location estimation techniques. *Optics express*, 17(26):23352–23373.

Anscombe, F. J. (1948). The transformation of poisson, binomial and negative-binomial data. *Biometrika*, 35(3/4):246–254.

Anthony, S. M. and Granick, S. (2009). Image analysis with rapid and accurate two-dimensional gaussian fitting. *Langmuir*, 25(14):8152–8160.

Ashley, T. T. and Andersson, S. B. (2015). Method for simultaneous localization and parameter estimation in particle tracking experiments. *Physical Review E*, 92(5):052707.

Badieirostami, M., Lew, M. D., Thompson, M. A., and Moerner, W. (2010). Three-dimensional localization precision of the double-helix point spread function versus astigmatism and biplane. *Applied physics letters*, 97(16):161103.

Beier, H. T. and Ibey, B. L. (2014). Experimental comparison of the high-speed imaging performance of an EM-CCD and sCMOS camera in a dynamic live-cell imaging test case. *PLOS One*, 9(1):e84614.

Berglund, A. J. (2010). Statistics of camera-based single-particle tracking. *Physical Review E*, 82(1):011917.

Calderon, C. P. (2016). Motion blur filtering: A statistical approach for extracting confinement forces and diffusivity from a single blurred trajectory. *Physical Review E*, 93(5):053303.

Cheng, H.-J., Hsu, C.-H., Hung, C.-L., and Lin, C.-Y. (2021). A review for cell and particle tracking on microscopy images using algorithms and deep learning technologies. *Biomedical Journal*.

Chenouard, N., Smal, I., De Chaumont, F., Maška, M., Sbalzarini, I. F., Gong, Y., Cardinale, J., Carthel, C., Coraluppi, S., Winter, M., et al. (2014). Objective comparison of particle tracking methods. *Nature methods*, 11(3):281–289.

Clarke, D. T. and Martin-Fernandez, M. L. (2019). A brief history of single-particle tracking of the epidermal growth factor receptor. *Methods and protocols*, 2(1):12.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the emalgorithm. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, 39(1):1–38.

Douc, R., Garivier, A., Moulines, E., and Olsson, J. (2011). Sequential Monte Carlo smoothing for general state space hidden Markov models. *Annals of Applied Probability*, 21(6):2109–2145.

Doucet, A., Godsill, S., and Andrieu, C. (2000). On Sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208.

Doucet, A., Johansen, A. M., et al. (2009). A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3.

Ewers, H., Smith, A. E., Sbalzarini, I. F., Lilie, H., Koumoutsakos, P., and Helenius, A. (2005). Single-particle tracking of murine polyoma virus-like particles on live cells and artificial membranes. *Proceedings of the National Academy of Sciences*, 102(42):15110–15115.

Fan, J. and Gijbels, I. (1996). *Local Polynomial Modelling and its Applications*. Chapman& Hall, CRC.

Freeman, M. F. and Tukey, J. W. (1950). Transformations related to the angular and the square root. *The Annals of Mathematical Statistics*, 21(4):607–611.

Gibson, S. and Ninness, B. (2005). Robust maximum-likelihood estimation of multivariable dynamic systems. *Automatica*, 41(10):1667–1682.

Gnedenko, B. V. (2017). *Theory of probability*. Routledge.

Godoy, B. I., Vickers, N. A., Lin, Y., and Andersson, S. B. (2020). Estimation of general time-varying single particle tracking linear models using local likelihood. In *2020 European Control Conference (ECC)*, pages 527–533.

Godsill, S. J., Doucet, A., and West, M. (2004). Monte carlo smoothing for nonlinear time series. *Journal of the american statistical association*, 99(465):156–168.

Granik, N., Weiss, L. E., Nehme, E., Levin, M., Chein, M., Perlson, E., Roichman, Y., and Shechtman, Y. (2019). Single-particle diffusion characterization by deep learning. *Biophysical journal*, 117(2):185–192.

Gupta, P. and Sehgal, N. K. (2021). *Introduction to machine learning in the cloud with python: Concepts and practices*. Springer Nature.

Handschin, J. E. and Mayne, D. Q. (1969). Monte Carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *International Journal of Control*, 9(5):547–559.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Helgadottir, S., Argun, A., and Volpe, G. (2019). Digital video microscopy enhanced by deep learning. *Optica*, 6(4):506–513.

Hilzenrat, G., Pandžić, E., Yang, Z., Nieves, D. J., Goyette, J., Rossy, J., Ma, Y., and Gaus, K. (2020). Conformational states control Lck switching between free and confined diffusion modes in T cells. *Biophysical Journal*, 118(6):1489–1501.

Holcman, D., Parutto, P., Chambers, J. E., Fantham, M., Young, L. J., Marciniak, S. J., Kaminski, C. F., Ron, D., and Avezov, E. (2018). Single particle trajectories reveal active endoplasmic reticulum luminal flow. *Nature cell biology*, 20(10):1118–1125.

Hou, Q., Zhang, D., Jarzylo, L., Huganir, R. L., and Man, H. (2008). Homeostatic regulation of ampa receptor expression at single hippocampal synapses. *Proceedings of the National Academy of Sciences USA*, 105(2):775–780.

Huang, F., Hartwich, T. M., Rivera-Molina, F. E., Lin, Y., Duim, W. C., Long, J. J., Uchil, P. D., Myers, J. R., Baird, M. A., Mothes, W., et al. (2013). Video-rate nanoscopy using scmos camera–specific single-molecule localization algorithms. *Nature methods*, 10(7):653–658.

Julier, S. J. and Uhlmann, J. K. (1997). New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–193. International Society for Optics and Photonics.

Kadri, U. (2019). Multiple-location matched approximation for bessel function j0 and its derivatives. *Communications in Nonlinear Science and Numerical Simulation*, 72:59–63.

Kao, H. P. and Verkman, A. S. (1994). Tracking of single fluorescent particles in three dimensions: use of cylindrical optics to encode particle position. *Biophysical Journal*, 67(3):1291–1300.

Ketkar, N. and Santana, E. (2017). *Deep learning with Python*, volume 1. Springer.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kotecha, J. H. and Djuric, P. M. (2003). Gaussian particle filtering. *IEEE Transactions on Signal Processing*, 51(10):2592–2601.

Kurzthaler, C., Devailly, C., Arlt, J., Franosch, T., Poon, W. C., Martinez, V. A., and Brown, A. T. (2018). Probing the spatiotemporal dynamics of catalytic janus particles with single-particle tracking and differential dynamic microscopy. *Physical review letters*, 121(7):078001.

Kusumi, A., Sako, Y., and Yamamoto, M. (1993). Confined lateral diffusion of membrane receptors as studied by single particle tracking (nanovid microscopy). effects of calcium-induced differentiation in cultured epithelial cells. *Biophysical Journal*, 65(5):2021–2040.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017a). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017b). Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007.

Lin, Y. and Andersson, S. B. (2019). Simultaneous localization and parameter estimation for single particle tracking via sigma points based em. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 6467–6472. IEEE.

Lin, Y. and Andersson, S. B. (2021). Computationally efficient application of sequential monte carlo expectation maximization to confined single particle tracking. In *2021 European Control Conference (ECC)*, pages 1919–1924.

Lin, Y., Sharifi, F., and Andersson, S. B. (2021). Three-dimensional localization refinement and motion model parameter estimation for confined single particle tracking under low-light conditions. *Biomedical optics express*, 12(9):5793–5811.

Liu, S. L., Wang, Z. G., Xie, H. Y., Liu, A. A., Lamb, D. C., and Pang, D. W. (2020). Single-Virus Tracking: From Imaging Methodologies to Virological Applications. *Chemical Reviews*, 120(3):1936–1979.

Loader, C. (1999). *Local Regression and Likelihood*. Springer.

Long, C. J., Brown, E. N., Triantafyllou, C., Wald, L. L., and Solo, V. (2005). Nonstationary noise estimation in functional MRI. *Neuroimage*, 28:890–093.

Long, F., Zeng, S., and Huang, Z.-L. (2012). Localization-based super-resolution microscopy with an sCMOS camera Part II: Experimental methodology for comparing sCMOS with EM-CCD cameras. *Optics Express*, 20(16):17741–17759.

Makitalo, M. and Foi, A. (2012). Optimal inversion of the generalized anscombe transformation for poisson-gaussian noise. *IEEE transactions on image processing*, 22(1):91–103.

McLachlan, G. J. and Krishnan, T. (2007). *The EM algorithm and extensions*, volume 382. John Wiley & Sons.

McLachlan, G. J. and Krishnan, T. (2008). *The EM algorithm and its extensions*. John Wiley& Sons, 2nd edition.

Merwe, R. v. d. and Wan, E. A. (2004). Sigma-point kalman filters for integrated navigation. In *Proceedings of the 60th annual meeting of the institute of navigation (2004)*, pages 641–654.

Michalet, X. (2010). Mean square displacement analysis of single-particle trajectories with localization error: Brownian motion in an isotropic medium. *Physical Review E*, 82(4):041914.

Michalet, X. and Berglund, A. J. (2012). Optimal diffusion coefficient estimation in single-particle tracking. *Physical Review E*, 85(6):061916.

Mortensen, K. I., Churchman, L. S., Spudich, J. A., and Flyvbjerg, H. (2010). Optimized localization analysis for single-molecule tracking and super-resolution microscopy. *Nature methods*, 7(5):377–381.

Nehme, E., Weiss, L. E., Michaeli, T., and Shechtman, Y. (2018). Deep-storm: super-resolution single-molecule microscopy by deep learning. *Optica*, 5(4):458–464.

Newby, J. M., Schaefer, A. M., Lee, P. T., Forest, M. G., and Lai, S. K. (2018). Convolutional neural networks automate detection for tracking of submicron-scale particles in 2d and 3d. *Proceedings of the National Academy of Sciences*, 115(36):9026–9031.

Park, H. Y., Buxbaum, A. R., and Singer, R. H. (2010). Single mrna tracking in live cells. In *Methods in enzymology*, volume 472, pages 387–406. Elsevier.

Pavani, S. R. P., DeLuca, J. G., and Piestun, R. (2009a). Polarization sensitive, three-dimensional, single-molecule imaging of cells with a double-helix system. *Optics Express*, 17(22):19644–19655.

Pavani, S. R. P., Thompson, M. A., Biteen, J. S., Lord, S. J., Liu, N., Twieg, R. J., Piestun, R., and Moerner, W. E. (2009b). Three-dimensional, single-molecule fluorescence imaging beyond the diffraction limit by using a double-helix point spread function. *Proceedings of the National Academy of Sciences*, 106(9):2995–2999.

Peerboom, N., Schmidt, E., Trybala, E., Block, S., Bergstrom, T., Pace, H. P., and Bally, M. (2018). Cell membrane derived platform to study virus binding kinetics and diffusion with single particle sensitivity. *ACS Infectious Diseases*, 4(6):944–953.

Rösch, T. C., Altenburger, S., Oviedo-Bocanegra, L., Pediaditakis, M., Najjar, N. E., Fritz, G., and Graumann, P. L. (2018). Single molecule tracking reveals spatio-temporal dynamics of bacterial dna repair centres. *Scientific reports*, 8(1):1–14.

Särkkä, S. (2013). *Bayesian filtering and smoothing*. Cambridge university press.

Saunter, C. D. (2010). Quantifying subpixel accuracy: an experimental method for measuring accuracy in image-correlation-based, single-particle tracking. *Biophysical journal*, 98(8):1566–1570.

Saxton, M. J. and Jacobson, K. (1997). Single-particle tracking: applications to membrane dynamics. *Annual review of biophysics and biomolecular structure*, 26(1):373–399.

Schechner, Y. Y., Piestun, R., and Shamir, J. (1996). Wave propagation with rotating intensity distributions. *Physical Review E*, 54(1):R50.

Shen, H., Tauzin, L. J., Baiyasi, R., Wang, W., Moringo, N., Shuang, B., and Landes, C. F. (2017). Single particle tracking: from theory to biophysical applications. *Chemical reviews*, 117(11):7331–7376.

Simson, R., Sheets, E. D., and Jacobson, K. (1995). Detection of temporary lateral confinement of membrane proteins using single-particle tracking analysis. *Biophysical journal*, 69(3):989–993.

Smal, I., Loog, M., Niessen, W., and Meijering, E. (2009). Quantitative comparison of spot detection methods in fluorescence microscopy. *IEEE T Med Imaging*, 29(2):282–301.

Smith, C. S., Joseph, N., Rieger, B., and Lidke, K. A. (2010). Fast, single-molecule localization that achieves theoretically minimum uncertainty. *Nature methods*, 7(5):373–375.

Tanizaki, H. (2003). Nonlinear and non-Gaussian state-space modeling with Monte Carlo techniques: A survey and comparative study. *Handbook of Statistics*, 21:871–929.

Taylor, R. W. and Sandoghdar, V. (2019). Interferometric Scattering (iSCAT) Microscopy and Related Techniques. In Astratov, V., editor, *Label-Free Super-Resolution Microscopy*, pages 25–65. Springer International Publishing.

Thompson, R. E., Larson, D. R., and Webb, W. W. (2002). Precise nanometer analysis for individual fluorescent probes. *Biophysical Journal*, 82(5):2775–2783.

von Diezmann, L., Shechtman, Y., and Moerner, W. (2017). Three-dimensional localization of single molecules for super-resolution imaging and single-particle tracking. *Chemical reviews*, 117(11):7244–7275.

Watanabe, S., Takahashi, T., and Bennett, K. (2017). Quantitative evaluation of the accuracy and variance of individual pixels in a scientific CMOS (sCMOS) camera for computational imaging. In *Single Molecule Spectroscopy and Superresolution Imaging X*, volume 10071, page 100710Z. International Society for Optics and Photonics.

Zhang, B., Zerubia, J., and Olivo-Marin, J.-C. (2007). Gaussian approximations of fluorescence microscope point-spread function models. *Applied Optics*, 46(10):1819–1829.

Zhong, Y. and Wang, G. (2020). Three-dimensional single particle tracking and its applications in confined environments. *Annual Review of Analytical Chemistry*, 13:381–403.

Curriculum Vitae [redacted]

[redacted]