LU TP 22-53
August 2022

Diffusion and Camera-Noise Modelling for Analysis of Single-Particle Tracking Movies

Erik Clarkson

Computational Biology and Biological Physics
Department of Astronomy and Theoretical Physics

Master thesis supervised by Tobias Ambjörnsson

LUNDS UNIVERSITET

Abstract

Interactions between T-cells and other body cells is an essential part of the immune system. It involves the binding between surface receptors of the two cells. Specifically, T-cell receptors (TCRs) bind onto pMHC (peptide-loaded major histocompatibility complex) molecules on the partnering cell surface. This way, a cell signal from the TCR to the nucleus can be initiated. Depending on the "message" of the signal, cells may respond in various ways. While it has been found that the signal message is dependent on how, where and when the signalling occurs, the interaction details and kinetics remain largely unclear. A specific observable that has been found experimentally is the average time duration of a binding event, called the "lifetime" of the interaction.

To measure biophysical parameters in practise, e.g. diffusion constants or lifetimes, is not as straightforward as it may appear. In here, we consider an experimental setup which utilises fluorescence microscopy in a controlled environment. The TCRs are allowed to move on a supported lipid bilayer (mimicking a real cell membrane) and have been labelled with a fluorescent dye. The partnering pMHCs are free to move on the surfaces of live T-cells. As such, the motion of these proteins is effectively two-dimensional and the resultant pMHC trajectories can be imaged with a fluorescence microscopy setup with a suitable camera.

After tracking the particles in a tracking program, we are left with data in the form of single-particle tracks. At this point, most conventional methods start by trying to estimate the diffusion constants (corresponding to free or bound TCR, respectively). For instance, this can be done with a usual mean-square-displacement analysis. Despite the importance of this data analysis step, many of its difficulties are often overlooked. These include systematic bias, such as tracking errors in the form of dot detection and dot linking. Moreover, there are uncharted errors involved in the analysis procedures and it is hard to assess the reliability and associated errors of estimated parameters.

In this study, we assess the ability of analysis methods for estimating diffusion constants and the fraction of steps spent in a free versus a bound state. To be able to estimate the errors on parameter estimates and potential biases in these, we analyse tracking data from synthetic movies. To this end, we include the diffusive motion, binding as well as photon statistics and camera-induced noise in the imaging system. As a benefit, the synthetic movies allow us to test and assess less conventional analysis methods on the data. In particular, we test a hidden Markov model analysis scheme and systematically compare it to a step-size distribution analysis. Furthermore, reliable simulated results could help us mitigate the uncertainty involved in the analyses by indicating sound experimental modifications. This can be done by calibrating all simulation parameters to agree with experiments, such that whatever errors and biases we obtain when analysing synthetic movies, will correspond well to those obtained with experimental movies.

Acknowledgements

I firstly want to thank my supervisor Tobias Ambjörnsson for his great overall support and for providing explanations based on common sense. In particular, his sense of including new material and ideas at the right moment has enabled widening, as well as deepening of this project. It has also been rewarding to work with Peter Jönsson's group at the Department of Physical Chemistry in Lund. Thank you Peter, Tommy Dam and Manto Chouliara for constructive and amusing discussions, and Tommy again for supplying experimental tracking films and calibration experiments and for giving feedback on the text regarding the experiments. Lastly, thanks to Albertas Dvirnas for early discussions regarding camera response simulations and associated code.

Populärvetenskaplig sammanfattning

I samband med att vi människor bor allt tätare och träffas allt oftare, ökar tyvärr även spridningen av virus. Det som tidigare i historien var en epidemi, blir nu snabbt en pandemi (t.ex. coronapandemin med start 2020). Mot denna bakgrund står det klart att vårt immunförsvar spelar en avgörande roll för vår hälsa och i sin tur samhällets funktion. Vår bästa metod är som bekant att träna upp vårt immunförsvar mot en specifik inkräktare, t.ex. ett virus, genom att vaccinera oss. Att läran om immunförsvaret är mer relevant än någonsin är därför ingen överdrift.

Men för att t.ex. kunna producera ett effektivt vaccin, behöver vi förstå hur vårt immunsystem fungerar. I synnerhet behöver immunsystemets celler kunna kommunicera med varandra. För det ändamålet har cellerna receptorer fästa på sina utsidor; informationsöverföringen sker genom att en sådan receptor från den ena cellen kemiskt binder till en motsvarande receptor på den andra cellen. Receptorerna är proteinkomplex som sträcker sig ner mot cellkärnan.

Denna typ av bindning är dock svår att studera, till följd av den mikroskopiska längdskalan och proteinernas komplexa former. Två relevanta storheter i sammanhanget är hur snabbt proteinerna rör sig i fritt och bundet tillstånd, och hur ofta ett protein byter mellan dessa tillstånd. Även tidsintervallet då proteinerna är bundna till varandra, kallad deras livstid, är viktigt för att förstå bindningarna. I artikeln [J. J. Y. Lin. et al, "Mapping the stochastic sequence of individual ligand-receptor binding events to cellular activation: T cells act on the rare events". I: Science Signaling 12.564 (2019)] visas att det är interaktionerna med ovanligt långa livstider som T-cellen reagerar på.

För att kringgå svårigheterna med att studera enskilda bindningar i detalj, studerar Peter Jönsson m.fl. vid fysikalisk kemi på kemicentrum vid Lunds universitet händelserna ur ett statistiskt perspektiv. De imiterar den ena celltypens cellmembran, på vilken de fäster många ytproteiner. Dessa proteiner har färgats in med ett självlysande ämne, sådant att de syns i en mörk omgivning. På så vis kan Jönssons grupp filma proteinernas tvådimensionella rörelser genom ett mikroskop. Idén bakom detta är att proteinerna bör röra sig mindre i bundet tillstånd, vilket syns om man analyserar filmen genom att

spåra enskilda proteiner.

Vilken sorts rörelse uppvisar egentligen proteinerna i en stilla, enformig vattenmiljö (utan några bindningar)? Då vattnet har en viss värme, rör sig faktiskt vattnets molekyler sett på en mikroskopisk längdskala. Denna rörelse leder i sin tur, via massor av krockar, till en observerad slumpmässig rörelse - diffusion - hos proteinerna. Diffunderingen är fullständigt oregelbunden, till synes likt en flygdrakes rörelser under snabbt varierande vindar. Utmaningen är att urskilja effekten av bindningen från denna slumpmässiga rörelse.

Att identifiera de bundna komplexen från att de diffunderar långsammare än de fria proteinerna försvåras av de många fler större stegen, tillsammans med diverse brus (t.ex. från kameran) som ingår i mätningarna. Mot den bakgrunden går projektet i denna uppsats ut på att undersöka förutsättningarna för att bestämma diffusionskonstanterna, sannolikheterna för att byta tillstånd och livstiden. Detta görs genom att simulera hela experimentet på en dator, för att sist analysera denna data med olika statistiska metoder. För detta behövs modeller av proteinernas rörelser och bindningar, mikroskopet och kameran som används i experimentet. Att simulera förloppet har nämligen fördelen att all relevant information är känd, exempelvis proteinernas positioner vid varje tidpunkt.

Contents

1 Introduction

Immune cells need to be able to communicate in order to mount an effective defence against harmful invaders. For this purpose, cell surface receptors bind to peptides displayed on MHC (major histocompatibility complex) molecules, expressed on the surfaces of other immune cells. These peptides carry information of what is happening inside the cell [1]. For instance, a virus-infected cell may display viral peptides, such that a killer T-cell can then scan them, committing it to kill the infected cell[1]. That is, T-cell responses are initiated by the triggering of signalling events from T-cell receptor - peptide MHC (TCR-pMHC) bindings.

The overall functioning of TCRs, such as those of killer T-cells, is understood fairly well. For instance, it is known that they consist of an external binding region and an internal signalling region. But despite experimental effort, the interaction kinetics of TCR-pMHC binding events and its connection to biological function are largely unclear [2]. The interactions take place on the two-dimensional plane of the meeting cell surfaces, which complicates the study compared to measurements in solution. How factors such as protein density and auxiliary binding molecules influence the TCR-pMHC binding is not yet fully understood [2]. What we do know is that the activation of T-cells depends on the TCR-pMHC binding parameters [3]. Another study revealed that T-cell activation results from binding events with rare long lifetimes [4].

Important to note, is that TCR-pMHC interactions is only one example of binding between cell surface proteins of T-cells and other body cells. There also exist adhesive surface proteins that help stabilise the TCR-pMHC interaction. An example is rCD2 that binds to rCD48 proteins, which is studied by Peter Jönsson's group at the Department of Physical Chemistry in Lund, see e.g. [5]. In fact, TCR-pMHC interactions are strengthened by rCD2 adhesion molecules, which act to align the two cell membranes [5].

A standard overall method to study these reactions, say the rCD2-rCD48 for definiteness, is to label one of the proteins, say the rCD2, with fluorescent dyes. The rCD2 are confined to the surface of a supported lipid bilayer (SLB), while the rCD48 are confined to the surfaces of live T-cells allowed to approach the SLB. In the cell contact, which forms once a T-cell has reached the SLB, the protein motion is effectively two-dimensional. The rCD2 motion can then be followed through fluorescence microscopy and its positions imaged with a camera. In these images, the rCD2 molecules appear as bright dots on a dark and noisy background. Tracking data is obtained with a single-particle tracking analysis, through a software that locates and links the dots between frames. This tracking can be done in a variety of ways, see [6] for a review.

Data analysis of rCD2-rCD48 tracking experiments is based on the displacements of the tracked particle, rCD2, which depend on the rCD2-rCD48 binding kinetics. This is typically done with a mean-square-displacement (MSD) analysis. In this kind of analysis, one plots the MSD of single particle tracks against time, to obtain the diffusion coefficient from the slope of a linear fit to the displacement data. In the simple case of no binding, this method works well. But in the present case of rCD2-rCD48 tracking, there is binding involved that lead to two separate diffusion constants, corresponding to free or bound

rCD2. Transitions between these two states make it more difficult to distinguish the two characteristic diffusion constants with an MSD analysis. Moreover, there are systematic errors, mostly related to dot detection and linking, in which experimenters lack insight. Thus, experimental researchers have little guidance in anticipating the consequences of tuning experimental parameters, estimating tracking errors, and of picking an optimal data analysis method.

To remedy the above lack of guidance available to researchers, stochastic computer simulations of reaction and diffusion processes are being increasingly employed. Most of these are, however, implemented directly from knowledge of the experimental system and its processes (e.g. diffusion), as in e.g. [7]. These simplifications confine their use, primarily to experimental consistency-checks of specific quantities that may be difficult to measure in practice. Further examples are found in [8] and [9].

In this study, we aim to provide a more rigorous and complete "testing ground" for the evaluation of different analysis methods. We do this through step-by-step simulations of the experiment, emanating in synthetic movies. First, we model the reaction-diffusion as a physical system with moving particles (Sec. 2.2.2). Then, we consider photon statistics relevant for hitting the camera pixels (Sec. 2.3.2). On this data, we impose the response of an sCMOS-camera (scientific complementary metal–oxide–semiconductor camera) filming the system (Secs. 2.3.5, 2.3.6, 2.3.7) together with the point spread function of the microscope (Sec. 2.4). As a demonstration, we implement these steps based on Jönsson's experimental setup. The camera-related parameters are then realistically selected based on the specific camera used (Photometrics Prime 95B). The other parameters (Sec. 2.2.3), related to the reaction-diffusion system, are chosen more freely to correspond to a bimolecular and reversible cell-to-cell reaction. We thereby produce high-quality synthetic movies.

Using our synthetic movies, we then test different data analysis procedures. Accordingly, we first generate particle trajectories from the synthetic movies (herein, we use the software TrackMate). These trajectories are then analysed using two methods: a hidden Markov model (HMM) analysis (Sec. 2.5.2) and a step-size distribution analysis (Sec. 2.5.3). By virtue of having full control over the simulated environment, we systematically test and study the effect of track length and fluorophore emission intensity on analysis uncertainties (Sec. 3.2). Many more quantities can be investigated in this way.

2 Methods

2.1 Outline

In this section, we outline the thesis and the questions that it aims to answer. The goal of all simulations, box **A.** and box **B.** in Fig. 1, is to produce realistic synthetic movies. By tracking the particles with a third-party software, box **C.**, we obtain trajectories. We then analyse the trajectories with two methods (step-size distribution analysis and HMM, box **D.**) to estimate physical parameters and draw conclusions regarding the accuracy of the methods in various cases.

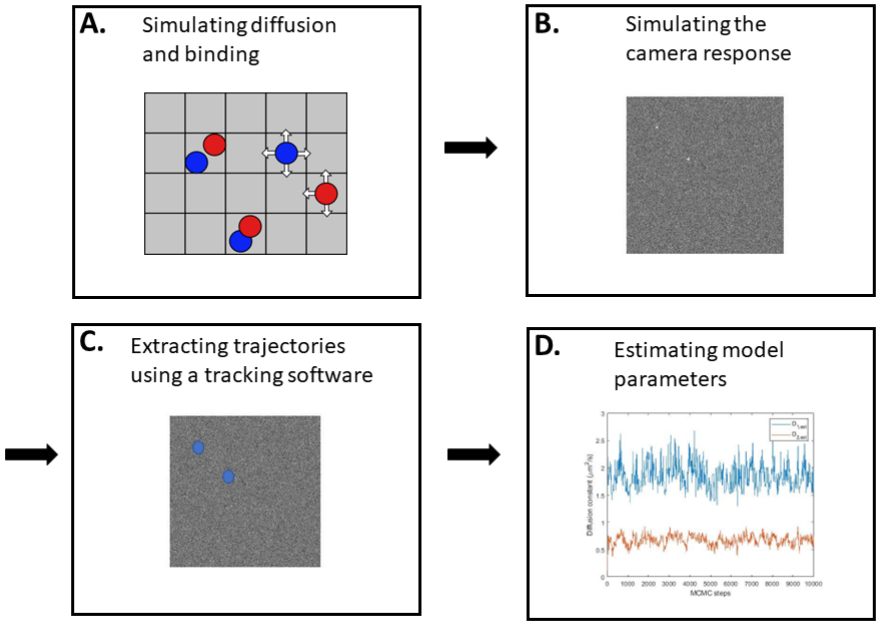


Figure 1: Schematic of the methods outline. We generate random particle trajectories, including stochastic binding, by simulating the motion of fluorescent proteins (box **A.**). With these positions known at every time, we generate associated images at different times, based on a calibrated camera response model (box **B.**). With synthetic movies at hand, we track the particles in an image analysis program (box **C.**). In box **D.**, we analyse the so-produced trajectories. Since all quantities involved in the synthetic movies are known, we can draw conclusions regarding the performance of the analysis methods by testing them on these movies. We can also learn what experimental parameters are better suited for a particular analysis scheme, so that a real fluorescence microscopy experiment can be optimally conducted.

In Fig. 1 box **A.**, we simulate the motion of the fluorescent (and non-fluorescent) proteins. The purpose of this is to generate realistic particle tracks. In the simulations, we

consider single jumps across a discretised space, where the lattice spacing is much smaller than the spatial resolution set by the imaging system. The tracked proteins can be in one of two possible states: freely diffusing with diffusion constant D_{A} or diffusing as a complex, bound to a substrate, with diffusion constant D_{C}. Our simulations of this process yields all particle positions at all times. The details of this simulation procedure is found in Secs. 2.2.2-2.2.3.

With access to simulated particle tracks from box **A.**, we want to make this data more realistic by including errors in the particle tracking. Therefore, in Fig. 1 box **B.**, we generate a synthetic film of noisified artificial images. These are based upon the particle positions obtained from box **A.** and the response of Jönsson's camera. When generating these synthetic movies, we consider photon statistics from the fluorophores to the camera pixels, the point spread function of the microscope and the read noise of the camera. The details of methods for generating synthetic movies is found in Sec. 2.4.

The overarching aim of our framework in Fig. 1 is to provide a testing ground for our analysis methods, so that we can obtain insight into how well they work in different situations. In Fig. 1 box **C.**, we upload the movies obtained in box **B.** into an image analysis software (ImageJ). Then, we track the particles (TrackMate plug-in) and extract tracking data in the form of trajectories. On this tracking data, we apply our analysis methods - step-size distribution analysis and HMM (box **D.**). Details of the methods for trajectory analysis are found in Secs. 2.5.2-2.6. For the synthetic movies we know all quantities of interest, which allow us to compare the performances of the two trajectory analysis methods.

2.2 Reaction-diffusion simulations

Reaction-diffusion processes are at the heart of realistic particle trajectories. In the following, we consider such processes involving precisely two types of particles (A and B). To be specific, the A-B particle reaction is modelled as a bi-molecular, reversible one. In-between these reactions, the particles diffuse freely with a diffusion constant depending on their particle type.

We use various methods for simulating diffusion with occasional binding. Beginning with Sec. 2.2.1, we generate the tracks independently and with Gaussian displacements. In these tracks, there are transitions between a free and a bound diffusive state. Then, in Sec. 2.2.2, we increase the realism of tracks by simulating a physical system of particles. The purpose of these methods is to generate ever more realistic data, on which we later apply our analysis methods. Note that, for the A-particles, we separate between a free state A (free diffusion) and a bound state C (diffusion of bound complex).

2.2.1 T1: Two-state diffusion simulations

As a simple way of producing two-state particle trajectories, we simulate a random walk in the continuous limit. The jump lengths are then distributed differently depending on which state the particle is in. Specifically, we follow the method outlined in [10]; a particle

is assumed to change state only in-between steps. Hence, we first generate a sequence of states $\mathcal{S} = s_1, s_2, ..., s_N$, each state associated with one of the N steps. This is actually a Markov chain, generated using Algorithm 1. It requires the input of transition probabilities p_{AC} (p_{CA}) from state A (C) to state C (A). Then, we draw two random numbers from Eq. (A.40) in Appendix A, one for each dimension, with the diffusion constant (D_{A} or D_{C}) associated with the state of that step.

If we were to analyse a two-state system with a one-state model, the single diffusion constant (*effective* diffusion constant), under the assumption of a homogeneous system, is [10]

$$D_{\mathrm{eff}} = \pi_{\mathrm{A}} D_{\mathrm{A}} + \pi_{\mathrm{C}} D_{\mathrm{C}}, \tag{2.1}$$

with π_{A} (π_{C}) being the probability to be in state A (state C). We have that $\pi_{\mathrm{A}} = \frac{p_{\mathrm{CA}}}{p_{\mathrm{CA}}+p_{\mathrm{AC}}}$ and $\pi_{\mathrm{C}} = 1 - \pi_{\mathrm{A}}$.

State sequence algorithm

Calculate stationary probabilities $\pi_{\mathrm{A}} = p_{\mathrm{CA}}/(p_{\mathrm{CA}} + p_{\mathrm{AC}})$, $\pi_{\mathrm{C}} = 1 - \pi_{\mathrm{A}}$
Generate a rectangular random number $r_1 \in U(0, 1)$
if $r_1 \leq \pi_{\mathrm{A}}$ **then**
    $s_1 = 1$
else
    $s_1 = 2$
end if
for $i = 2$ to $i = N$ **do**
    Generate a rectangular random number $r_i \in U(0, 1)$
    if $s_{i-1} = 1$ AND $r_i \leq p_{\mathrm{AC}}$ **then**
        $s_i = 2$
    else if $s_{i-1} = 2$ AND $r_i \leq p_{\mathrm{CA}}$ **then**
        $s_i = 1$
    else
        $s_i = s_{i-1}$
    end if
end for

Algorithm 1: Generating a 2-state state sequence. By using transition probabilities p_{AC} and p_{CA}, the rules of a Markov chain are applied. At each step, the state either changes to the other one, or stays the same. 1 corresponds to state A and 2 corresponds to state C.

2.2.2 T2: Gillespie simulations

To include reactions and model protein motion more accurately, we now include more aspects in the simulations. The T1 simulations described above are namely idealised in two important aspects. Primarily, because state transitions are constructed to happen instantaneously in-between time frames (steps), but also because exponential waiting times between binding/unbinding are presumed. We will in this section introduce a more advanced method for generating trajectories, that lets us drop those two idealisations.

The dynamics is as follows. Firstly, there are two particle types: A and B diffusing with constants D_A and D_B, respectively. For simplicity, two particles of the same kind are not allowed to occupy the same square. However, an A and a B may. Such a double-occupancy is a necessary condition for a molecular reaction to happen. A bound complex of the two particles can then form. This binding happens at a rate Q_{on} and likewise, unbinding happens at a rate k_{off}. In similarity with free particles, these bound complexes also diffuse. Lastly, reflecting boundary conditions are imposed on the system edges (the edges of a cell contact region modelled as a square).

At this point, a method is needed that can handle this complexity. To generate stochastic trajectories from the model described above, we use the Gillespie method. These are flexible simulations, wherein, at a given time, one lists all possible "reactions". By a reaction, we refer to any possible event: a diffusive step or binding/unbinding. A reaction is then selected at random, with a probability proportional to its rate. After a reaction has happened and the system has been updated accordingly, the time is updated based on the inverse sum of reaction rates. By repeating these steps, a stochastic realisation of the time evolution of the chemical system simulated is obtained. Our model is as follows: there are four types of "particles" - an A-particle alone on its square (A-particle), a B-particle alone on its square (B-particle), an A- and a B-particle unbound on the same square (AB_u-particle) and bound on the same square (AB_b-particle). The A-particle has four possible "reactions" (hop diffusion), each with a rate k_A. Likewise for the B-particle, with rate k_B. The AB_u-particle has nine different reactions - hop diffusion in either of four directions for either particle, each with rate k_A and k_B respectively, and binding with rate Q_{on}. Lastly, the AB_b-particle has 5 different reactions - hop diffusion in either direction as a complex with rate k_{AB}, or unbinding with rate k_{off}.

Unfortunately, slow computing speed is a major drawback for systems with at least one large molecular population using the original Gillespie algorithms. For cellular systems, this is almost always the case [11]. A reason for it being slow, is that the cumulative (partial) sum of all rate constants must be recalculated at each step. For instance, if an A-particle has another A-particle as its neighbour, the rate constant for diffusion in that direction must be updated and set equal to zero to forbid that reaction. Since the first Gillespie algorithms were presented, several newer versions have improved upon them [11]. We decided to use a modified version of the Gillespie algorithm presented in [12]. It is called the "trial-and-error method" and is computationally efficient for our system, where most lattice sites are unoccupied.

Our new method is tailored for interacting systems of particles with different diffu-

sion constants. While the original Gillespie simulation methods were intended to handle molecular reactions directly as concentrations [13][11], we here use the underlying ideas to handle the situation of discrete diffusion and binding as part of a bi-molecular reaction. A Gillespie algorithm is then an example of a kinetic Monte Carlo (KMC) method [14]; here essentially the same as a time-dependent rejection-KMC simulation. We use it as a way of generating stochastic trajectories for interacting particles.

To simulate the process described above, we introduce a fast version of the Gillespie algorithm that avoids repeated calculations of partial sums of rate constants (see Eq. (2.8) below). In each step we draw first a waiting time and then we pick a reaction at random, based on probabilities resulting from the rates. If the reaction is allowed, we update the system, time and rates, and draw a new waiting time. Else, we increase the waiting time for the next reaction. The simulation is stopped when it reaches a chosen stop time t_{stop}. So, instead of listing all possible reactions we use a "factorising method". To this end, we first pick a particle type (A, B, AB_u or AB_b), then a specific particle of that type, and lastly a reaction from the list of possible reactions for that particle type. The mean waiting time between reactions, allowed or forbidden, is

$$\tau_{\text{TE}} = 1/(4N_\text{A}k_\text{A} + 4N_\text{B}k_\text{B} + N_{\text{AB}_\text{u}}(4k_\text{A} + 4k_\text{B} + Q_\text{on}) + N_{\text{AB}_\text{u}}(4k_\text{AB} + k_\text{off})), \tag{2.2}$$

with N_A, N_B, N_{AB_u} and N_{AB_b} being the number of particles of the type denoted in their indices. Note that these numbers change during the simulation run and that τ_{TE} is the inverse sum of all possible reaction rates. Only if the number of particles of a certain type changes during a reaction, do we need to recalculate τ_{TE}. And because diffusion is much more common than binding/unbinding, this recalculation is infrequent. The probabilities corresponding to each particle type are now

$$P_\text{A} = 4N_\text{A}k_\text{A}\tau_{\text{TE}} \tag{2.3}$$
$$P_\text{B} = 4N_\text{B}k_\text{B}\tau_{\text{TE}} \tag{2.4}$$
$$P_{\text{AB}_\text{u}} = N_{\text{AB}_\text{u}}(4k_\text{A} + 4k_\text{B} + Q_\text{on})\tau_{\text{TE}} \tag{2.5}$$
$$P_{\text{AB}_\text{b}} = N_{\text{AB}_\text{u}}(4k_\text{AB} + k_\text{off})\tau_{\text{TE}}. \tag{2.6}$$

That is, the probability for picking a reaction involving the particle type indicated. All these probabilities have the same basic structure: they are proportional to the number of ways that they can happen (e.g. 4 directions for every N_A particle for P_A) and also to their respective rate (or sum of rates) and to the waiting time. The waiting time is drawn from the distribution (B.5), simply by using the continuous transformation method. Similarly, a specific reaction is drawn from (B.6), this time with the discrete transformation method. Note that the transformation method uses the cumulative density function (CDF) of the relevant distribution.

The exact procedure, specifying the steps above, are listed below.

1. Initialise random particle positions and set time $t = 0$.

2. Compute the discrete CDF of picking a certain particle reaction. This is done only for $\mathrm{AB_u}-$ and $\mathrm{AB_b}-$particles.

3. Compute the discrete CDF of picking a certain particle type. This is done by computing the partial sums

$$p_0 = 0 \tag{2.7}$$

$$p_\mu = \sum_{\nu=0}^{\mu-1} p_\nu, \quad \mu = 1, 2, 3, 4 \tag{2.8}$$

of probabilities associated with particle types of different μ, see Eqs. (2.3)-(2.6).

4. Set the waiting time equal to zero: $\tau = 0$.

5. Update the waiting time by drawing a random number $r_1 \in U(0, 1)$ and updating

$$\tau \to \tau + \tau_{\mathrm{TE}} \log(\frac{1}{r_1}).$$

6. Pick a particle type according to the probabilities in Eqs. (2.3) - (2.6). This is done by drawing a random number $r_2 \in U(0, 1)$ and searching for the type μ satisfying

$$p_\mu < r_2 p_4 \leq p_{\mu+1}. \tag{2.9}$$

7. Pick a specific particle by drawing a random integer between 1 and the current number of such particles.

8. Pick a reaction.

   (a) If it is an A- or B-particle, pick a jump direction by drawing a random integer between 1 and 4.

   (b) If it is an $\mathrm{AB_u}$- or an $\mathrm{AB_b}$-particle, draw a random number $r_3 \in U(0, 1)$ and search for the reaction satisfying the respective equation analogous to Eq. (2.9).

9. Check if the reaction is allowed (not colliding with another particle or violating BCs). If it is forbidden, return to step 5.

10. If $t \leq t_{\mathrm{stop}}$, return to step 3 or 4, depending on whether the number of particles of different kinds have changed. Else, end the simulation.

The random initial particle positions are computed using the algorithm due to Bebbington [15]. Since no bindings are present at the start, we let the system reach equilibrium. The time to reach equilibrium is estimated in Appendix C. A schematic of the algorithm can be seen in Fig. 2 below.

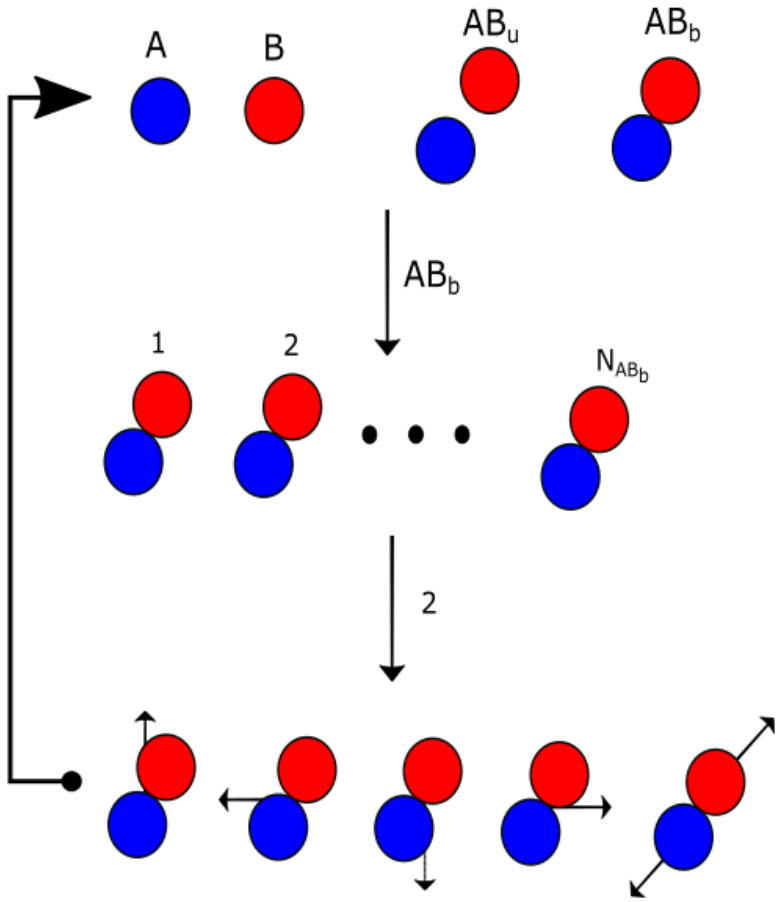


Figure 2: Schematic of our Gillespie/T2 simulations. Picking a reaction is factorised into three individual random picking steps. One first selects one out of four possible "particle" types (A, B, AB_u, AB_b). Each particle represents a possible state of a non-empty lattice site (in the figure, we pick an AB_b-particle, representing a bound complex of two particles). In the second step, we list all existing AB_b-particles and pick one of those (number 2 in the figure). Lastly, we list all possible "reactions" available to the AB_b-particle. From this list of hop diffusion in either direction and unbinding, we pick one reaction.
If the reaction happens to be forbidden, one increases the waiting time and draws a new reaction.

Table 1: Experimental parameters. Parameters that we consider as given for the experimental system that we want to simulate. Note that these values are reasonable, albeit not corresponding to a particular real-life experiment.

Parameter	Numerical value	Unit	Description
c_A	0.14	$1/\mu\text{m}^2$	A-particle concentration
c_B	99	$1/\mu\text{m}^2$	B-particle concentration
R	0.0075	μm	protein radius
A	71	μm^2	cell contact area
D_A	1.5	$\frac{\mu\text{m}^2}{s}$	A-particle diffusion constant
D_B	1.2	$\frac{\mu\text{m}^2}{s}$	B-particle diffusion constant
D_C	0.20	$\frac{\mu\text{m}^2}{s}$	AB-complex diffusion constant
k_{off}	10	1/s	dissociation rate
k_{on}	0.30	$\frac{\mu\text{m}^2}{\text{s}\cdot\text{molecule}}$	association rate
Δt	5	ms	camera exposure time

Table 2: T1 simulation parameters. All parameters are chosen to correspond to the experimental ones in Table 1 with long tracks.

Parameter	Numerical value	Unit	Description
D_A	1.5	$\frac{\mu\text{m}^2}{s}$	A-particle diffusion constant
D_C	0.20	$\frac{\mu\text{m}^2}{s}$	AB-complex diffusion constant
p_{AC}	0.0929		transition probability from free to bound
p_{CA}	0.0464		transition probability from bound to free
N	4000		track length (steps)
L	10		number of tracks

2.2.3 Choice of simulation parameters

Picking values for our T1 simulation parameters is simple. We set the diffusion constants equal to the experimental ones in Table 1, and decide upon how many tracks we want and how long we want them to be (see Table 2). Note that in this thesis, the experimental values have been manually and reasonably selected, without referring to a specific experiment. The only challenge is how to pick the transition probabilities. We do this after choosing simulation parameters for our T2 simulations, through Eqs. (2.12)-(2.13) in this section.

As for the T1 simulations, we need to pick in-simulation parameter values for the T2 simulations. By proceeding from the experimental values summarised in Table 1, we determine the simulation parameters in Table 3.

Table 3: T2 simulation parameters. All parameters are determined as described in section 2.2.3.

Parameter	Numerical value	Unit	Description
N_A	10		number of A-particles
N_B	7000		number of B-particles
M	1120		number of lattice squares per side
t	20	s	simulation time
k_A	$2.7 \cdot 10^4$	1/s	A-particle hop rate
k_B	$2.1 \cdot 10^4$	1/s	B-particle hop rate
k_C	$3.6 \cdot 10^3$	1/s	AB-complex hop rate
a	0.0075	μm	lattice constant
$\alpha = a^2$	$5.6 \cdot 10^{-5}$	$\mu\mathrm{m}^2$	voxel area
k_off	10	1/s	dissociation rate
k'_on	$3.6 \cdot 10^3$	1/s	association rate

Hereon, we prescribe the undetermined parameters. Let us start by choosing the lattice constant (side length of a single lattice square), a. We set it equal to the radius of one of the proteins, R (assuming that they are nearly identical in size).

Next, we want all particles to diffuse with realistic diffusion constants. Beginning with the A-particles, their diffusion constant is proportional both to their lattice hope rate k_A and to the square of the lattice constant a:

$$D_A = k_A a^2, \tag{2.10}$$

so that we can solve for k_A. In contrast, to determine k_B - the analogue of k_A for B-particles, we realise that this is a "hidden variable". Because the B-particles are not labelled, experiments have not determined their diffusion constant. Nevertheless, we do assume that they are approximately of the same size as the A-particles, but possibly slowed down by other surface molecules as explained in [16]. We therefore assume a lower diffusion constant, see Table 1. For the bound complexes, we follow exactly the same reasoning as for A-particles.

Moving on, we now want to determine the *simulation* rates of reaction between the two proteins. The simulated dissociation rate of bound complexes is simply equal to its experimental value k_off. For the association rate, the relation between the experimental rate k_on and the simulation rate k'_on is [17]

$$k_{\text{on}} = \alpha k'_{\text{on}}, \tag{2.11}$$

under the assumption of perfect mixing. k'_{on} is the binding rate for an A- and a B-particle on the same site, α is the voxel area a^2 (physical area of a lattice square) and k_{on} is the resulting rate constant for the system. Hence, to determine k'_{on}, we need only solve for it in Eq. (2.11) above.

Finally, let us determine the last parameters. We set the number of lattice squares per side M to yield the actual cell contact area A, i.e. $M = A/\alpha$. The particle numbers $N_{\text{A}}, N_{\text{B}}$, we set to match their concentrations $c_{\text{A}}, c_{\text{B}}$, respectively.

We end this section with some remarks. Firstly, although the perfect mixing assumption is almost fulfilled in these simulations, by construction, two particles of the same kind may not occupy the same square, leaving fewer available squares. Thus, there should be a small correction factor on the RHS, that is a little higher than unity. Secondly, the relatively large concentration of B-particles yields an excess of binding sites, allowing us to follow [10] and consider the bi-molecular reaction as a reversible uni-molecular one. That is, transitions of A-particles between free and bound states. The transition probabilities for a sample time Δt are then given by

$$p_{\text{AC}} = \frac{Q_{\text{on}}}{Q_{\text{on}} + k_{\text{off}}}\left(1 - e^{-(Q_{\text{on}}+k_{\text{off}})\Delta t}\right), \tag{2.12}$$

$$p_{\text{CA}} = \frac{k_{\text{off}}}{Q_{\text{on}} + k_{\text{off}}}\left(1 - e^{-(Q_{\text{on}}+k_{\text{off}})\Delta t}\right). \tag{2.13}$$

2.3 Estimating camera and light parameters

Our goal here is to completely characterise the response of the sCMOS camera used in Jönsson's experiments, i.e. "Teledyine Photometrics Prime 95B". To do so, we wish to obtain a probability density function (PDF) for the final image count, based on the photon flux and all noise sources. A schematic of this procedure is shown in Fig. 3. We calculate the PDF based on the useful properties of characteristic functions (CFs), displayed in Appendix G.

2.3.1 Calibration experiments

In order to mimic the specific camera used in Jönsson's laboratory, we need not only a camera model, but also accurate model parameters. Therefore, we analyse simple camera experiments, which is a process we call "camera calibration". We make use of three kinds of experiments to determine all unknown model parameters. These unknowns include a gain parameter g and an offset o from Eq. (2.14), as well as the parameters of the distributions of X (mean number of background photon counts) and Y (shape and scale parameters for the read noise). Below, we describe three calibration experiments for this purpose.

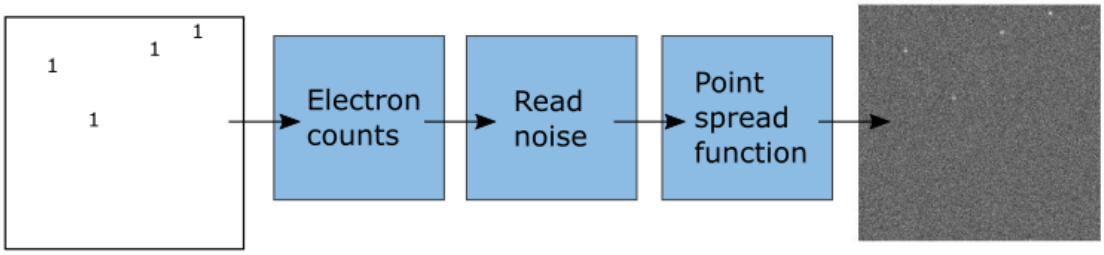


Figure 3: Schematic of camera simulation procedure. One begins with all emitter positions (represented by 1's in the top left) on a grid of pixels . Then, one adds background and emitter counts, camera read noise and blurring caused by the point spread function. Thus, one has generated a single time frame of a synthetic movie based on a fluorescence microscopy experiment.

- **Fluorescence microscopy experiment** The first experiment is done by filming fluorescent, diffusing emitters through a microscope. These emitters (A-particles) are part of a bi-molecular, reversible reaction happening inside a cell contact. In particular, these experimental movies contains information about the signal (fluorophore) and background Poisson parameters.
- **Dark frames experiment** In the second experiment, no light is incident on the camera lens. Images are simply taken in a dark room with the camera cap on. This procedure yields "dark frames". 100 frames were taken in this way with an exposure time of 15 ms.
- **Stationary frames experiment** In the third experiment, a stationary scene of a partly lit wall inside a room is imaged many times. Contrary to [18], we do not repeat our procedure for varying overall light levels, because the variation in light intensity is already large enough. In doing so, we use a 15 ms exposure time, which leaves all but the brightest regions rather dark, much like in a real tracking experiment. By taking 100 such frames, we obtain our "stationary frames".

2.3.2 Photon statistics

The number of background photons λ_{Bg} hitting a given pixel is Poisson-distributed. The same holds for "signal" photons λ_{S} that come from the fluorophores, see Appendix F.

2.3.3 Camera response model

Let us consider the output value of a pixel. The quantitative components that make up a pixel output are presented in Appendix F. In brief, photon counts multiplied by a quantum efficiency (QE) factor gives "photo-electron counts". That is, the number of electrons liberated due to the photoelectric effect, are proportional to the photon count. In the conversion of these counts to voltages, the process goes through various stages in the camera (the read-out structure), resulting in a collective "read noise". At the end, there is a round-off error to give integer digital counts, called "quantisation noise".

So first, the light hits the pixels, secondly there is a read-out noise in the camera and thirdly, there is a rounding to integers. We neglect the rounding and formulate this mathematically as a sum of random variables:

$$I = gX + Y + o, \tag{2.14}$$

where I is the final pixel value in digital counts (ADU = analog-to-digital units), X is a Poisson random variable in units of e^- (photo-electrons), Y is a Tukey-lambda random number in units of ADU. Tukey-lambda is a family of distributions that can approximate numerous symmetric distributions and was included in [18] to model the read noise. It is characterised by two parameters - a shape parameter λ_{TL} and a scale parameter σ_{TL}; further details are given in Sec. 2.3.7. g and o are constants; g is the conversion factor from photo-electrons to digital image counts given in units of ADU/e^-, and o is an offset in units of ADU added to every pixel to make each count non-negative. Eq. (2.14) demonstrates what components (gX, Y, o) comprise a final pixel count I. In the course of the exposure time, during which all light for a frame is collected, a mean number of photons will have hit the pixel. By converting these to the right units, and adding camera read noise and offset, we obtain a final pixel value I.

2.3.4 Camera PDF

In constructing the camera PDF, we need a way to compute the distribution for the random variable I in Eq. (2.14). For that reason, we consider the CF of I, $\Phi_I(k) = \langle e^{ikI} \rangle$. The virtue of doing this lies in the fact that the CF of a sum of random variables factorise. In the end, we obtain the PDF of I by Fourier-inverting $\Phi_I(k)$. Thus, the CF of our noise model $I = gX + Y + o$ becomes

$$\Phi_I(k) = e^{iok} \Phi_X(gk) \Phi_Y(k), \tag{2.15}$$

$\Phi_X(gk)$ being the CF of the background light noise and $\Phi_Y(k)$ being the CF of the read noise. At this point we use Eq. (G.6) to invert to the camera PDF. With the CF in general being complex (see Appendix H), this formula is particularly useful for numerical implementation. The integration is done with a trapezoidal quadrature. We use this PDF as a consistency-check against background from a real experiment, by visually checking that the background counts are indeed generated (roughly) from the PDF. If that is the case,

it confirms the validity of our noise model and implies that we can expect the background of the simulated images to resemble those of the real experiment.

2.3.5 Estimating offset

Let us start by estimating the offset, since it is independent of all other camera-model parameters. The offset balances the read noise, which is symmetric about zero and thus could introduce negative counts. Moreover, it can be conveniently estimated from dark frames [19] (see the "dark frames experiment" data set in Sec. 2.3.1). Because no light is incident on the lens, all variations in image count must be due to the read noise. Therefore, the offset o is simply calculated as the statistical mean of the dark frames:

$$o = \frac{1}{N}\sum_{i=1}^{N} J_i, \tag{2.16}$$

where N is the number of frames and J_i is the mean image count for frame i.

2.3.6 Estimating gain

In a camera, the final counts should be proportional to the number of photo-electrons set free by the incident light; this constant of proportionality is the gain g. We determine g via a mean-variance analysis, as in [18]. As such, we calculate the mean and variance of Eq. (2.14):

$$\langle I\rangle = g\langle X\rangle + \langle Y\rangle + o \tag{2.17}$$

$$\mathrm{Var}(I) = \langle I^2\rangle - \langle I\rangle^2 \tag{2.18}$$

At this point, we could insert Eqs. (2.14) and (2.17) into Eq. (2.18) and develop the squares. However, it is quicker to utilise the general formula

$$\mathrm{Var}(aX + bY) = a^2\mathrm{Var}(X) + b^2\mathrm{Var}(Y) + 2ab\mathrm{Cov}(X, Y),$$

where $\mathrm{Cov}(X, Y) = \langle (X - \langle X\rangle)(Y - \langle Y\rangle)\rangle = \langle XY\rangle - \langle X\rangle\langle Y\rangle$ is the covariance of X and Y, and also $\mathrm{Var}(a) = 0$. In these formulae, X, Y are any random numbers and a, b are constants. Now, given that the noise sources follow sequentially in time and have different physical origins, we assume X, Y to be uncorrelated. By definition, the covariance term is then zero. As such, Eq. (2.18) simplifies to

$$\mathrm{Var}(I) = g^2\mathrm{Var}(X) + \mathrm{Var}(Y). \tag{2.19}$$

A known property of the Poisson distribution is that $\mathrm{Var}(X) = \lambda = \langle X\rangle$, X being Poisson distributed and λ being the distribution parameter. With this fact at hand, we combine Eqs. (2.17) and (2.19) to yield

$$\mathrm{Var}(I) = g\langle I\rangle + c_1, \tag{2.20}$$
$$c_1 = \sqrt{\mathrm{Var}(Y) - \langle Y\rangle} - o.$$

In Eq. (2.20), we see that g is the slope of a line in a variance versus mean plot. We can now make use of the stationary frames (see the "stationary frames experiment" data set in Sec. 2.3.1) in determining the gain. From the frames, we follow each pixel's value in a time-series. For each series, we compute the variance and mean, obtaining a point in a mean-variance plot. At this stage, we do a linear regression to obtain an estimate of g.

2.3.7 Estimating read noise parameters

Next, we characterise the read noise (Y in Eq. (2.14)), by estimating the shape parameter λ_{TL} and scale parameter σ_{TL} of its distribution. We assume this distribution to belong to the Tukey-lambda family of distributions, as done in Appendix F. In short, the Tukey-lambda family can approximate many known symmetric distributions, e.g. a uniform, Gaussian, or heavy-tailed Gaussian. We here follow the methods presented in [18], and make use of the dark frames (see the "dark frames experiment" data set in Sec. 2.3.1) once again as our data set.

The first step is to decide upon what specific distribution within the Tukey-lambda family to use. Equivalently, we want to find the value of the shape parameter, λ_{TL}, that best suits our data. This can be done with the help of a graphical technique, known as a PPCC-plot (probability plot correlation coefficient plot) [20]. It works like this: consider a specific value of λ_{TL}. Construct a probability plot (Quantile-Quantile plot, QQ-plot, with one theoretical distribution) of the quantiles of our dark frames data versus the theoretical quantiles of the chosen distribution. Then, calculate a Pearson correlation coefficient (PCC) of the plot "curve". (Note that if our data was drawn from exactly this distribution, the line should be linear with unit slope.) This PPC "score" assesses the linear correlation between the data and the chosen distribution, such that a value of 1 means perfect correlation, a value of 0 means no correlation (and -1 means perfect anti-correlation). Now, we repeat these steps for many values of λ_{TL} and finally plot the PCCs versus the corresponding λ_{TL}s, which is precisely our PPCC-plot. From this plot, we graphically see which λ_{TL} corresponds to the highest PCC, and pick this particular value.

With the shape parameter known, we want to estimate the scale parameter σ_{TL}. Again, we make use of a QQ-plot, this time only for our chosen value of λ_{TL}. It turns out that a linear regression of the plotted points is closely related to the scale parameter: its slope is an estimate of scale (and its intercept is an estimate of location) [20]. Thus, we directly obtain σ_{TL} (and the location parameter which estimates the offset).

2.3.8 Estimating background photon counts

Moving on to parameters related to photon statistics, we estimate λ_{Bg}, i.e. the background mean photon counts per pixel during exposure time. From Eq. (2.17) we have

$$\langle I \rangle = g\lambda_{\mathrm{Bg}} + o \iff \lambda_{\mathrm{Bg}} = \frac{\langle I \rangle - o}{g}. \tag{2.21}$$

We utilise Eq. (2.21) in order to estimate λ_{Bg} in the following way: consider a background region from an actual experiment (the "fluorescence microscopy experiment" data set in Sec. 2.3.1) and follow each such pixel in a time-series. By computing the mean of each time-series we have estimated $\langle I \rangle$ and can thus solve for λ_{Bg}. All these approximations make up a histogram and its mean value is our final estimate of λ_{Bg}. The width of this histogram decreases as the number of images considered increases; this width represents the analysis uncertainty.

2.3.9 Estimating signal photon counts

Not only does a final image have background photon counts, but also "signal" photon counts λ_{S}, associated with every emitter. It represents the mean number of emitted photon counts per emitter during exposure time. These counts are distributed over nearby pixels; how many pixels to consider, is determined by the standard deviation of the microscope point spread function (PSF), σ_{PSF}, and our acceptable confidence interval ($6\sigma_{\mathrm{PSF}}$). Generalising Eq. (2.21) for a region around n emitters, we have

$$\langle I_{\mathrm{tot}} \rangle = ng\lambda_{\mathrm{S}} + Ng\lambda_{\mathrm{Bg}} + No \iff \lambda_{\mathrm{S}} = \frac{1}{ng}(\langle I_{\mathrm{tot}} \rangle - N(g\lambda_{\mathrm{Bg}} + o)). \tag{2.22}$$

Here, I_{tot} is the total intensity (image counts) of the region, $\langle I_{\mathrm{tot}} \rangle$ is the average of this over several frames and N is the number of pixels in the image region. The data set used was the "fluorescence microscopy experiment" in Sec. 2.3.1. By using Eq. (2.22), we obtain a single number for λ_S.

2.4 T3: Generating artificial images

Consider a two-dimensional physical system of diffusing fluorophores, which occasionally bind. These molecules emit photons that travel through a microscope and onto a camera sensor. One might wonder "what will the resulting images look like?". As we have shown, the answer depends on the expected number of incoming background photons, noise from the camera internal functioning (read noise) and a constant value added to every pixel (offset). Moreover, the answer also depends on the microscope: the light wavelength acts in conjunction with the microscope in a way characterised by the point spread function (PSF). The PSF results in that a point particle appears as a Gaussian with width σ_{PSF}, and is explained in more detail in Appendix E. Thus, given a quantitative camera model such as Eq. (2.14) above, we here seek to generate representative images of this physical system on a computer. We do this as follows:

1. Consider a pixel.
2. Consider an emitter.

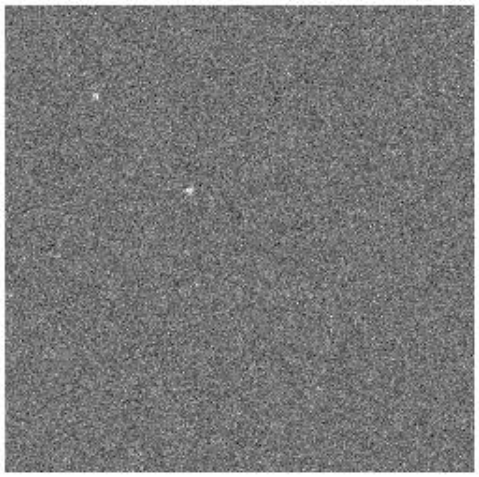


Figure 4: Simulated camera response. This is an example image generated as described in section 2.4. The following parameters were used: $\lambda_{\mathrm{Bg}} = 2, g = 1.8, \sigma_{\mathrm{PSF}} = 1.3, \sigma_{\mathrm{TL}} = 1.9, o = 99$.

.

3. Check if emitter k is close enough to affect pixel i, j.
4. Calculate the probability that the photons from emitter k will hit the pixel.
5. Multiply with the expected number of photon counts.
6. Add background, read noise and offset.
7. Repeat steps 2-6 for every emitter.
8. Repeat steps 1-7 for every pixel.

Eq. (E.3) in Appendix E is used to calculate the probability of photon counts in step 4 above. An example image generated in this way can be seen in Fig. 4.

To produce a film of artificial images, we simply collect the individual images in the right order and save them as a single TIF-file. This file-format is the same as produced in experiments and is compatible with most image-analysis software. Our synthetic movies are thus composed of T2-trajectories with added camera-response. The trajectories that are extracted from the movies with a tracking programme, we call "T3" and is our most realistic set of trajectories.

2.5 Estimation of motion model parameters

We list two methods employed to determine diffusion constants and transition rates or fraction of steps in a free versus bound state.

2.5.1 Particle tracking in movies

In order to extract tracking data from a fluorescence microscopy movie, we need to track the particles. Herein, we use the TrackMate plug-in in ImageJ to do this. As detector, we use the Laplacian-of-Gaussian type and as tracker, we use the LAP type. Still, there are many alternative programmes and methods that could be used for tracking.

2.5.2 A1: Hidden Markov model analysis

Our first method for estimating diffusion constants (and binding/unbinding rates) utilises a Hidden Markov model. A Markov chain is a process with a "short memory": the probability for transitioning to another state is conditionally dependent only the current state. Adding the word "hidden", means that we have access only to the data produced by the model [21], but the state variables themselves cannot be observed. That is, as the model is in a certain state, it emits data with a certain probability. An important characteristic of Markov models is that the time spent in each state, i.e. the waiting times, is exponentially distributed. To summarise, we model the A-particle tracks as a Markov process, with model parameters that are hidden to us.

Our analysis is from this point to estimate these parameters. In other words, we are given a track (or several independent ones), considered as a sequence of displacements. By utilising a HMM analysis method, we try to recover the set of model parameters $\theta = \{D_{\mathrm{A}}, D_{\mathrm{C}}, p_{\mathrm{AC}}, p_{\mathrm{CA}}\}$. Note also that the average lifetime $1/k_{\mathrm{off}}$ can be solved for by combining Eqs. (2.12) and (2.13).

To analyse tracks, in particular T2 generated by our Gillespie motion model, we want to compute an associated likelihood. The idea is that, for any set of particle position observations and choice of model parameters, we can estimate a likelihood. Later, this likelihood can be maximised with respect to our model parameters, to infer what parameters are most likely to have yielded the observations. We follow the method given in [10]. The likelihood for a diffusion coefficient D_i for a displacement $\mathbf{r}_j$ is

$$L_i(\mathbf{r}_j) \propto \frac{1}{4\pi D_i \Delta t} \exp\left(-r_j^2/(4D_i\Delta t)\right), \tag{2.23}$$

where, $r_j^2 = x_j^2 + y_j^2$. Up to constant coefficients, the log-likelihood is

$$l_i(\mathbf{r}_j) = -\frac{r_j^2}{4D_i\Delta t} - \log(D_i\Delta t) \tag{2.24}$$

and computed using Algorithm 2. The log-likelihood for an ensemble of M independent trajectories $\{\mathbf{O}^{(1)}, ..., \mathbf{O}^{(M)}\}$, where $\mathbf{O}^{(i)}$ is trajectory number i, is the sum of log-likelihoods for each individual trajectory [10]:

log-likelihood estimation algorithm

1: $\theta = \{D_{\mathrm{A}}, D_{\mathrm{C}}, p_{\mathrm{AC}}, p_{\mathrm{CA}}\}$
2: Generate a rectangular random number $r \in U(0,1)$
3: Define logsum $(x,y) = \log\left(e^x + e^y\right) = \begin{cases} x + \log\left(1 + e^{y-x}\right), & \text{if } x \leq y \\ y + \log\left(1 + e^{x-y}\right), & \text{otherwise} \end{cases}$
4: Calculate $l_i(\mathbf{r}_j)$ for $i = 1, 2$ and $j = 1, ..., N$ using Eq. (2.5.2)
5: $\alpha_1(i) = \log(\pi_i) + l_i(\mathbf{r}_1)$; $i = 1, 2$
6: **for** $j = 2$ to $j = N$ **do**
7: $\quad \alpha_j(i) = \mathrm{logsum}[\alpha_{j-1}(1) + \log(p_{1i}),\ \alpha_{j-1}(2) + \log(p_{2i})] + l_i(\mathbf{r}_j); \quad i = 1, 2$
8: **end for**
9: $L(\theta|\mathbf{O}) = \mathrm{logsum}[\alpha_1(N),\ \alpha_2(N)]$

Algorithm 2: Calculating the log-likelihood of the parameters of a track. α is the so-called forward variable. This algorithm is a modified version of the forward-backward algorithm as seen in e.g. [21]. Here, 1 corresponds to state A and 2 corresponds to state C.

$$L(\theta|\mathbf{O}^{(1)}, ..., \mathbf{O}^{(M)}) = \sum_{k=1}^{M} L(\theta|\mathbf{O}^{(k)}), \tag{2.25}$$

where $\theta = \{D_{\mathrm{A}}, D_{\mathrm{C}}, p_{\mathrm{AC}}, p_{\mathrm{CA}}\}$ is the set of model parameters.

Once we have access to a track, or a set of independent tracks, we want to use Bayesian inference to estimate the parameter set that is most likely to have yielded the specific track. In other words, we seek a way of optimising the likelihood function, Eq. (2.25), for the parameter set θ. We do as in [10] and use a stochastic Markov chain Monte Carlo (MCMC) scheme, based on the Metropolis algorithm. When applying this algorithm, seen in Algorithm 3, we first let the system thermalise. This entails waiting for the MCMC to reach an equilibrium, so that the current values are independent on the initial configuration. The maximum likelihood estimates are then simply computed as the sample mean. We tweak the displacement scale number so that the number of accepted moves divided by the number of proposed moves along a parameter axis after thermalisation to $30 - 50\%$.

Optimisation algorithm

$n \longleftarrow$ Number of MCMC steps.
$d = \{d_1, d_2, d_3, d_4\} \longleftarrow$ Scale for displacement along each parameter axis.
Pick a random initial position in parameter space $\theta^{(0)} = \{\theta_1^{(0)}, \theta_2^{(0)}, \theta_3^{(0)}, \theta_4^{(0)}\}$
Calculate the log-likelihood of the guessed parameter set, i.e. $L(\theta^{(0)}|\mathbf{O})$, as in Algorithm 2.
for $i = 1$ to $i = n/4$ **do**
  for $k = 1$ to $k = 4$ **do**
    $l = 4(i-1) + k - 1 \longleftarrow$ Number of current counts.
    Propose a displacement $\delta\theta_k$ along the kth parameter axix, drawn from a normal distribution with mean 0 and variance s_k: $\theta^{(\text{proposed})} = \theta^{(l)} + \delta\theta_k$.
    Calculate $L(\theta^{(\text{proposed})}|\mathbf{O})$.
    if $L(\theta^{(\text{proposed})}|\mathbf{O}) \geq L(\theta^{(l)}|\mathbf{O})$ **then**
      $\theta^{(l+1)} = \theta^{(\text{proposed})}$
    else
      Generate a rectangular random number $r \in U(0,1)$
      if $\log(r) \leq L(\theta^{(\text{proposed})}|\mathbf{O}) - L(\theta^{(\text{l})}|\mathbf{O})$ **then**
        $\theta^{(l+1)} = \theta^{(\text{proposed})}$
      else
        $\theta^{(l+1)} = \theta^{(l)}$
      end if
    end if
  end for
end for

Algorithm 3: Finding optimal parameter values. A Metropolis algorithm as a stochastic method for calculating our maximum likelihood estimates.

2.5.3 A2: Analysis based on Rayleigh's distribution

This approach for determining D focuses on the observable

$$s = \sqrt{(\Delta x)^2 + (\Delta y)^2},$$

i.e. the step length. Following the derivation in Appendix D, the PDF for s is

$$\phi(s) = \frac{s}{b^2} \exp\big(-\frac{s^2}{2b^2}\big).$$

In words, it follows a Rayleigh distribution where D appears as parameter through the relation $b^2 = 2D\Delta t$. Therefore, if we have determined b, we have also determined D.

Finding b turns out to be easy. With access to tracks, we compute and store every individual step length s. Note that we here do not make use of information regarding to which track an individual displacement belongs. Then, for data with only one diffusion constant, we would plot the histogram of s together with a maximum-likelihood-estimate (MLE) fit of a Rayleigh curve, estimating b.

This procedure has the potential of resolving both diffusion constants from 2-state data. One then fits two separate Rayleigh curves to the histogram. Our final function, which we MLE-fit to the distribution, can be written as a weighted sum of Rayleigh-distributions:

$$f(\pi_{\mathrm{A}}, b_{\mathrm{A}}, b_{\mathrm{C}}) = \pi_{\mathrm{A}}\phi(b_{\mathrm{A}}) + (1 - \pi_{\mathrm{A}})\phi(b_{\mathrm{C}}), \tag{2.26}$$

where b_{A}, b_{C} are our maximum-likelihood estimations related to the free and bound diffusion constants, respectively. π_{A} is the fraction of steps spent in the free state, c.f. Eq. (2.1).

2.6 Track segmentation

The A1 analysis introduced in Sec. 2.5.2 can, in contrast to the A2 analysis, output segmented tracks. Suppose that we have analysed a given track $\mathbf{O}$ and thus obtained a corresponding set of maximum likelihood estimates $\hat{\theta} = \{\hat{D}_{\mathrm{A}}, \hat{D}_{\mathrm{C}}, \hat{p}_{\mathrm{AC}}, \hat{p}_{\mathrm{CA}}\}$. We then want to associate each step (displacement) with a particle state - a procedure called "track segmentation". For each step $j = 1, ...N$ along a track, we estimate the most likely state $\hat{s}_j$ that the particle is in. As in [10], we use a modified version of the forward-backward algorithm, Algorithm 4. This way, we obtain a state sequence for each track. By storing the true states in a T2 simulation, we can in the end compare estimated state sequences to true state sequences.

Track segmentation algorithm

1: Calculate the forward variable $\alpha_j(i)$ for $j = 1, ..., N$ and $i = 1, 2$ with $\theta = \hat{\theta}$.
Use the procedure in Algorithm 2.
2: $\beta_N(i) = 0$; $i = 1, 2$
3: **for** $j = N - 1$ to $j = 1$ **do**
4: $\beta_j(i) = \text{logsum}[\log(p_{i1}) + l_1(r_{j+1}) + \beta_{j+1}(1),\ \log(p_{i2}) + l_2(r_{j+1}) + \beta_{j+1}(2)]; \quad i = 1, 2$
With $\theta = \hat{\theta}$.
5: **end for**
6: $\hat{s}_j = \underset{i=1,2}{\text{argmax}}[\alpha_j(i)\beta_j(i)]$ for $j = 1, ..., N$.

Algorithm 4: Segmenting a given track. We are given a track and its maximum likelihood estimates. Each step j along the track is associated with its most likely state: $i = 1$ (A) or $i = 2$ (C).

3 Results

In this section, we demonstrate results regarding camera response simulation and analysis of generated tracks. Beginning with the camera response in Sec. 3.1, we first collect results related to the calibration of our camera model. At the end, we show how well our camera noise model fairs against experimental noise, both quantitatively and qualitatively.

Continuing with analysis results in Sec. 3.2, we first include waiting times from within our motion model. We then include diffusion constant estimations from both the A1 and the A2 analysis. Furthermore, we test a track segmentation. This analysis we do in three steps: with T1 data generated by a 2-state Markov model, T2 data generated by motion model simulations, and T3 data with a camera response and real tracking of a synthetic movie with T2 trajectories.

3.1 Camera and photon statistics parameters

As noted in Sec. 2.3, a key to be able to simulate realistic-looking movies in the computer is to correctly handle photon statistics and the processing of the input photons by the imaging system's camera. These effects are modelled as described in Sec. 2.3.3, where we found that we need seven model parameters: λ_{Bg} and λ_{S} - mean number of background and emitter photons, respectively, hitting a given pixel during exposure time, σ_{PSF} - the point spread function standard deviation, g - gain, a proportionality factor in the camera conversion to final pixel counts, read noise in the camera processing with a shape parameter λ_{TL} and a scale parameter σ_{TL} for that distribution, and offset o - a constant count added to every pixel.

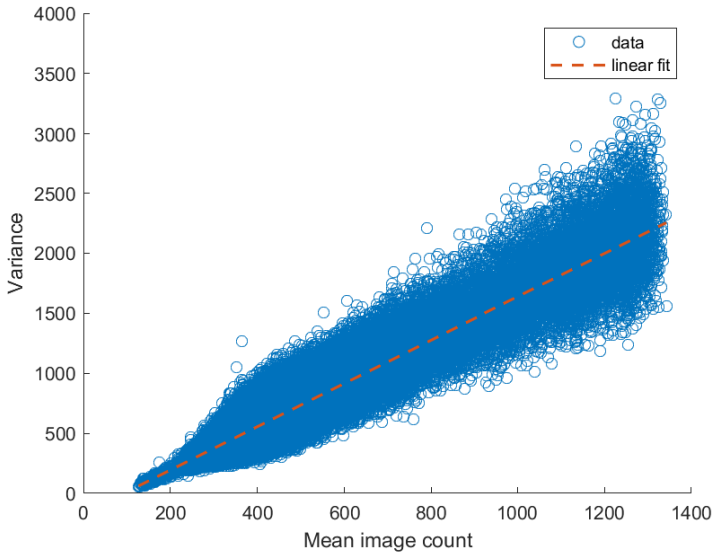


Figure 5: Mean-variance plot for estimating gain. A scatter plot consisting of time-series data from 1.44 million pixels and 100 frames. The camera used was a Teledyne Photometrics Prime 95B. Camera data are "stationary frames", which consist of groups frames, each with its own constant light level. The slope of this plot is the camera gain, as described in Sec. 2.3.6.

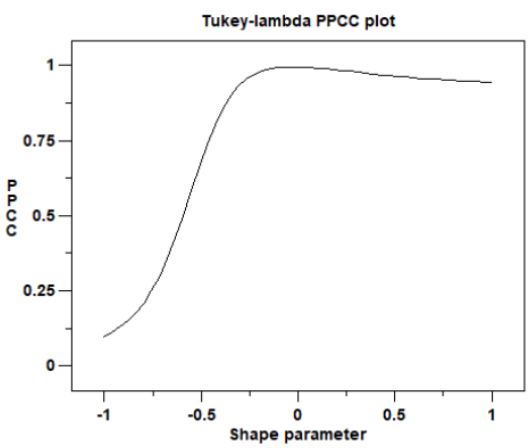
3.1.1 Results for camera parameters

The first camera parameter, offset, is simply obtained from the dark frames (Sec. 2.3.5). By computing the mean of every frame, we obtain $o = 99.2$. This value is reasonable, given that the manufacturer quotes a value of around 100 in [22].

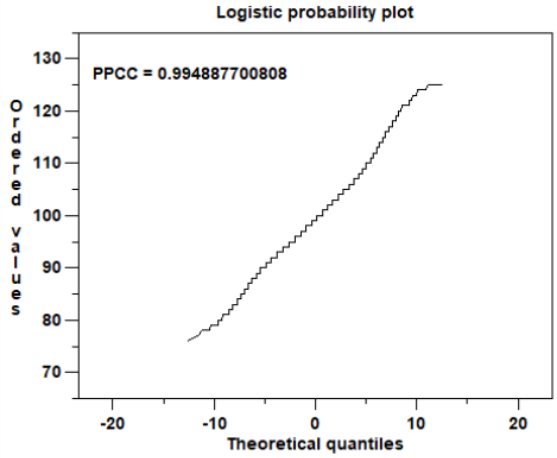
Our next result relates to the camera electron-to-digital conversion and is based on the stationary frames image data set (Sec. 2.3.1). Here, the gain is obtained as the slope of a mean-variance plot (see Sec. 2.3.6). In Fig. 5, we show such a mean-variance plot, and see that the line is approximately linear, as predicted by the analysis in Sec 2.3.6. The slope of this blue line is then our result for the camera gain, i.e. $g = 1.8$. This value is in the range of tested cameras in [19].

The final camera-related parameters to determine are those characterising the read noise. To obtain its shape and scale parameters, we make a PPCC plot and a QQ-plot, respectively (see Sec. 2.3.7). These two plots, which are based on the dark frames data set (Sec. 2.3.1), can be seen in Fig. 6. In Fig. 6a, we find that a shape parameter approximately equal to zero is the optimal one. We pick this exactly as zero, $\lambda_{\mathrm{TL}} = 0$, since this corresponds exactly to a logistic distribution [23]. The corresponding scale value is then obtained from Fig. 6b, yielding $\sigma_{\mathrm{TL}} = 1.9$.

For an experimental and an assumed distribution that are identical, the values in a QQ-plot should follow a straight line with slope equal to one. The line in Fig. 6b approximately follows this description, except that its ends deviate somewhat. This small deviation in tail



(a) PPCC-plot for determining the shape parameter of the read noise distribution.



(b) QQ-plot for determining the scale parameter of the read noise distribution.

Figure 6: Plots for determining read noise parameters. Camera data are same "dark frames" in both panels, which consist of frames with zero expected incoming light. The shape parameter for which the PPCC is highest is the optimal one. In (a), we see that this occurs close to a shape parameter equal to zero. In (b), we see a probability plot of the data. From a linear regression, we obtain approximations of the scale and location parameters. These methods are described in Sec. 2.3.7.

behaviour implies that one of the distributions has a slightly heavier tail than the other, or is slightly more skewed. But since the Tukey-lambda family can generate heavier tails than the current value of λ_{TL}, we draw the conclusion that the actual read noise is slightly skewed rather than being fully symmetric as the Tukey-lambda distributions, in agreement with [22].

3.1.2 Results for photon statistics parameters

Beginning with the background photons, Fig. 7 shows a histogram over calculated λ_{Bg} using Eq. (2.21) in Sec. 2.3.8. The data seen is based on the "fluorescence microscopy experiment" data set in Sec. 2.3.1. The mean value of the histogram, $\lambda_{\mathrm{Bg}} \approx 2.1$, is close to the integer value $\lambda_{\mathrm{Bg}} = 2$, which is our result.

Unlike background photon counts, signal photon counts are not completely stable, due to e.g. photo-bleaching and photo-blinking. We found λ_{S} to vary between time frames and emitters, and so decided to settle with the approximate value $\lambda_{\mathrm{S}} = 130$, using Eq. (2.22) in Sec. 2.3.9.

Although determining σ_{PSF} quantitatively can be done, we approximated it visually. As such, we set $\sigma_{\mathrm{PSF}} = 1.3$ pixels.

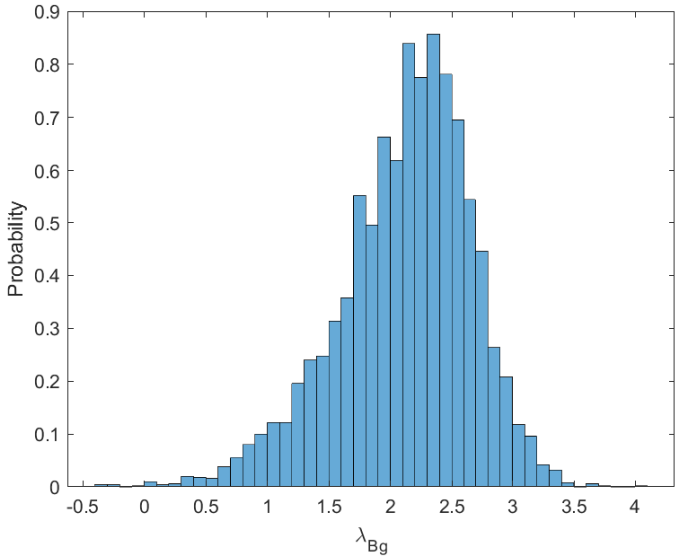


Figure 7: Histogram of estimated background Poisson parameter. Each pixel's time-series have been used to calculate λ_{Bg} using Eq. (2.21) in Sec. 2.3.8. This data is from a background region of a real experiment.

3.1.3 Consistency check of camera and photon statistics parameters

Our estimated parameter values for the camera and photon statistics completely determine the camera response model. With our camera model being fully calibrated, we test it against experimental images. The result of this test can be seen in Fig. 8. We see that the model (orange curve) fits the data (histogram) well. The orange curve was computed from the model CF, Eq. (2.15), as described in section 2.3.4. The good agreement seen in Fig. 8, indicate that our choice of camera parameters and λ_{Bg} are able to produce image counts similar to those in experiments.

What is left then, is to validate our choices for σ_{PSF} and λ_{S}. To this end, let us do a visual comparison between a simulated and an experimental image. The result is shown in Fig. 9. The emitters share roughly the same size and brightness between the two images, and the background regions look qualitatively similar. Note that there are only two emitters in Fig. 9a and many more in Fig. 9b.

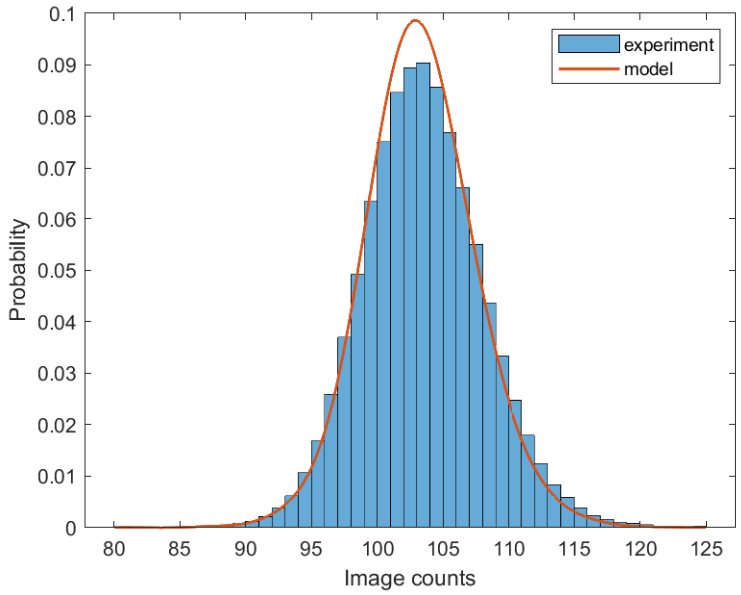
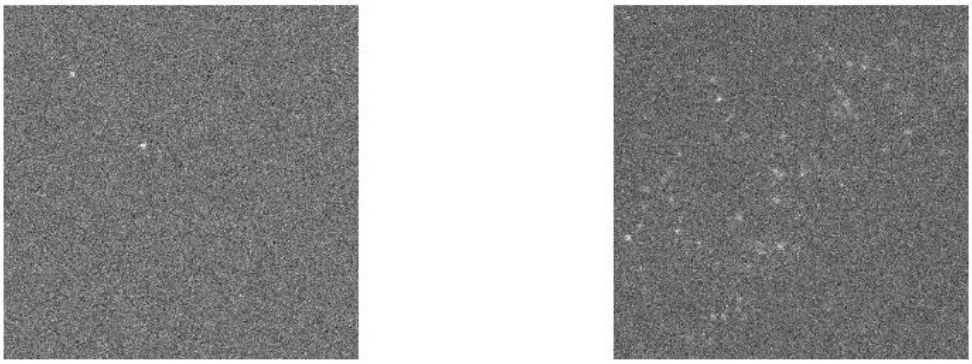


Figure 8: Camera noise PDF versus experimental background. A calibrated camera PDF (see Secs. 2.3.4 and 2.3), compared to experimental background data for 10 frames. This plot works as consistency check for the camera model (Sec. (2.3.3) and its calibration.



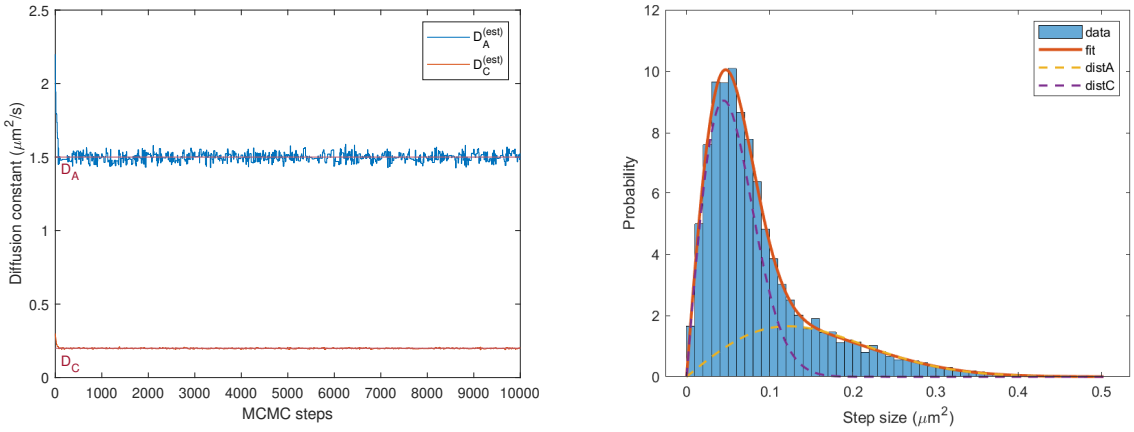
(a) Simulated camera-response. **(b)** Experimental image.

Figure 9: Comparison of a simulated image and an experimental image. The background should look qualitatively similar, since the camera model is based upon the experimental data from a film including the frame in (b). The process behind the generation of subfigure (a) is explained in detail in Sec. 2.3.

3.2 Motion model

In this section, we analyse tracking data with different models and methods. First, we check that waiting times in simulations are indeed exponential (recall Sec. 2.5.2). Then, we compare the performance of the HMM (A1) and step-size distribution (A2) analyses on three increasingly realistic data sets (T1, T2, T3) by comparing to ground truth parameters. For every data set, we estimate the diffusion constants for free and bound A-particles for full tracks with a sampling time of 5 ms, then for truncated tracks with varying number of frames (steps), and in the end run a track segmentation analysis.

3.2.1 Analysis of T1 trajectories



(a) A1 estimation on the complete data.

(b) A2 estimation on the complete data.

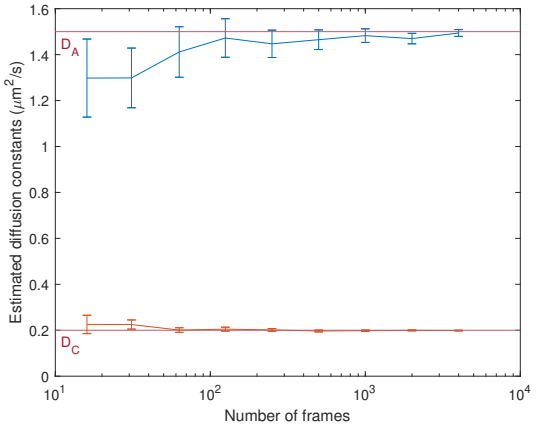
Figure 10: Display of analysis methods. 10 tracks with 1000 positions each were generated directly via a Markov-sequence, as described in Sec. 2.2.1. The parameters used were $D_\mathrm{A} = 1.5\ \frac{\mu\mathrm{m}^2}{s}$, $D_\mathrm{C} = 0.2\ \frac{\mu\mathrm{m}^2}{s}$, $p_\mathrm{AC} = 0.0929$, $p_\mathrm{CA} = 0.0464$. In (a), a Metropolis-like optimisation method is employed to find the optimal parameter values of the free (D_A) and bound (D_C) diffusion constants and transition probabilities (not plotted). The final estimations are calculated as the means after an initial thermalisation phase. In (b), a step-size distribution analysis is instead employed. An MLE fit of the histogram yields estimates of both diffusion constants and the bound fraction of particles. The two individual step size distributions are dashed.

Let us now test our A1 and A2 analyses. Here, we simulate tracks directly from the two-state sequence model T1 as described in Sec. 2.2.1. This provides us with a simple test on ideal data. A demonstration of the analysis methods is shown in Fig. 10. In subfigure (a), we see the optimisation scheme for the A1 analysis, wherein the final estimates are obtained as mean values. The fitting of Rayleigh curves for the A2 analysis is visible in subfigure (b), where we see how the distributions behind the dotted curves make up the final histogram.

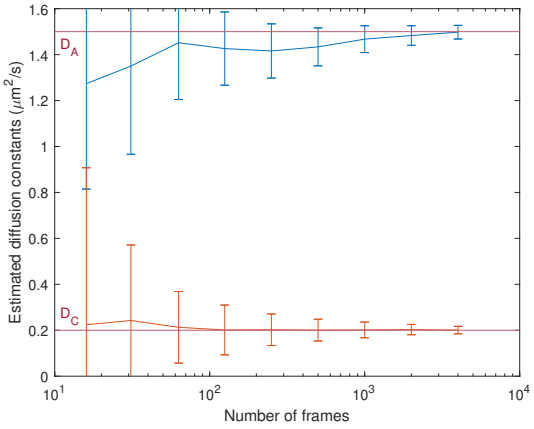
The results of the track analysis can be seen in Fig. 11 below, where we double the number of time frames for every plotted value. We see in Figs. 11a-11b, that both analysis methods are successful at inferring the diffusion constant of the bound fraction, D_{C}, given sufficiently many time frames. The estimations of D_{C} converge faster than those of D_{A}, because each particle spends twice the amount of time in that state. Likewise, the associated error is smaller for D_{C} than for D_{A}. After about 200 time frames, we obtain good approximations of both diffusion constants, using either method. With the full data set - 4000 time frames - excellent estimations are achieved.

The results of estimations of transition probabilities/bound fraction can be seen in Figs. 11c-11d. Compared to the estimation of diffusion constants, convergences are slower and there is a stronger dependence on the number of time frames considered. For the A1 method, we see a relatively smooth convergence for p_{CA} and a more rough one for p_{AC}. At 500 frames and above, we obtain good estimations. The A2 method requires as many time frames to successfully converge and determine the fraction of bound particles, π_{C}. With 4000 time frames, both methods give very precise estimations.

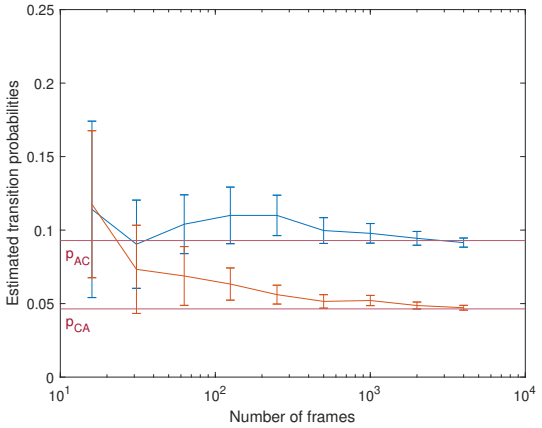
The track segmentation can be seen in Figs. 12a-12b. In the evaluation of these, we have used the A1 estimations from 4000 time frames to obtain state sequences for 4000 time frames (although the sequences of the first 1000 steps are plotted for easier visuals). Qualitatively, all the true features are present in the estimated sequence. With an average correctness of 93%, we find that track segmentation works rather well. A high "score" is expected, given that the T1 data is both generated and analysed with a Markov model. From testing different systems however, we found that the score increases further with smaller transition probabilities.



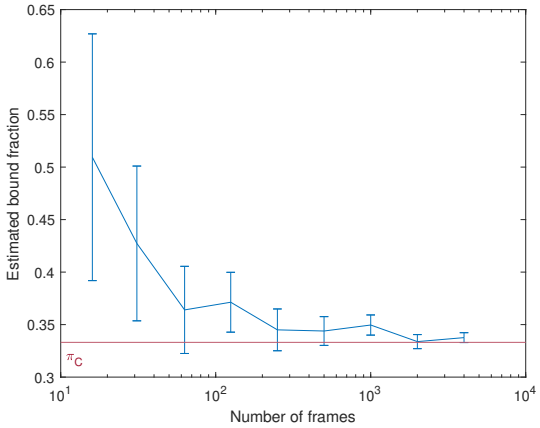
(a) A1 estimates of free (D_A) and bound (D_C) diffusion constants.



(b) A2 estimates of free (D_A) and bound (D_C) diffusion constants.

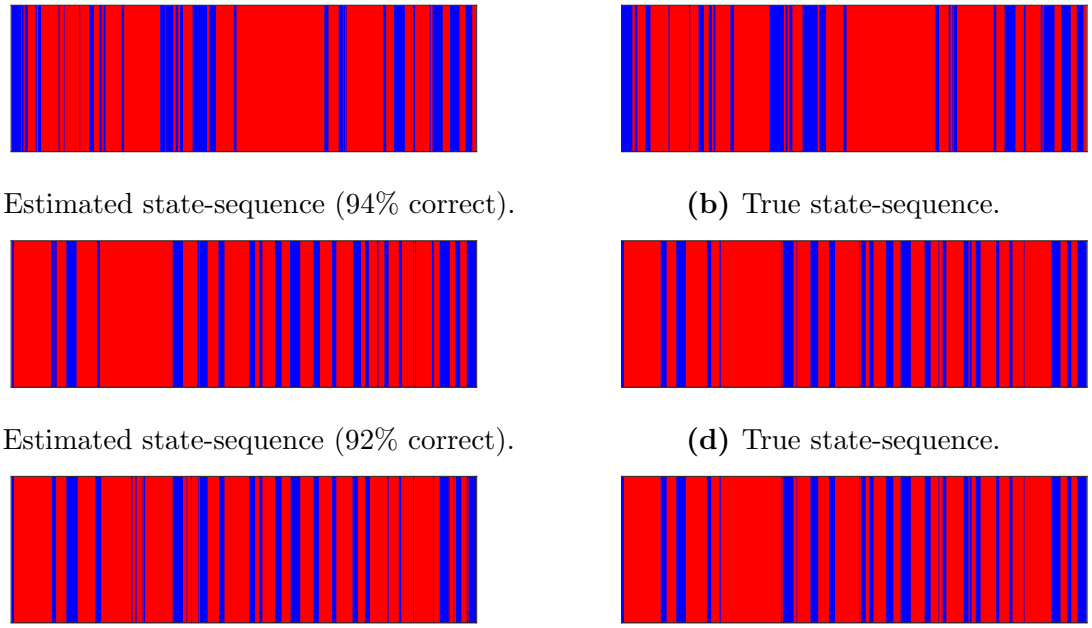


(c) A1 estimates of transition probabilities p_AC and p_AC between a free and a bound state.



(d) A2 estimates of the fraction π_C of particles that are bound at a given time.

Figure 11: Analysis of T1 trajectories. 10 tracks were generated directly via a Markov-sequence, as described in Sec. 2.2.1. The parameters used were $D_\mathrm{A} = 1.5\ \frac{\mu\mathrm{m}^2}{s}$, $D_\mathrm{C} = 0.2\ \frac{\mu\mathrm{m}^2}{s}$, $p_\mathrm{AC} = 0.0929$, $p_\mathrm{CA} = 0.0464$. Based on the analysis of these tracks, physical parameters were inferred using a HMM/A1 approach (see Sec. 2.5.2) in (a) and (c), and a step-size distribution/A2 approach (see Sec. 2.5.3) in (b) and (d). These estimations, of diffusion constants and transition probabilities/bound fraction, are plotted as a function of time frames (i.e. track length or number of steps).



(a) Estimated state-sequence (94% correct). **(b)** True state-sequence.

(c) Estimated state-sequence (92% correct). **(d)** True state-sequence.

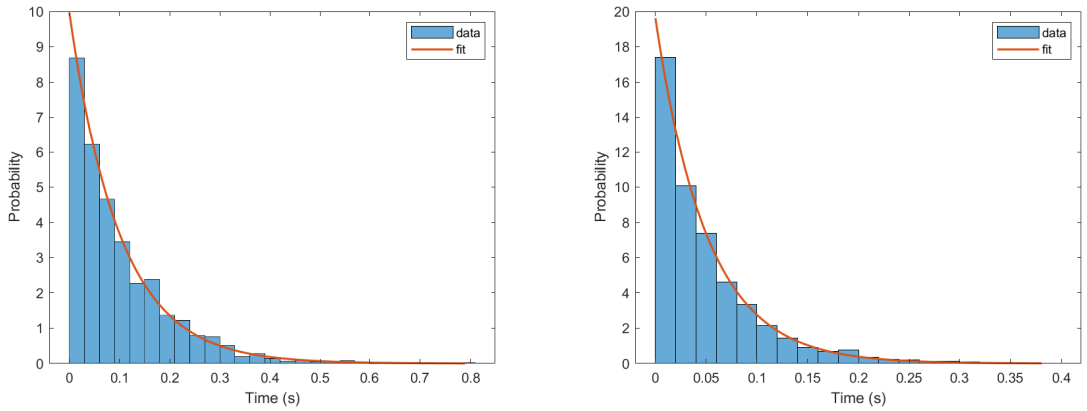
(e) Estimated state-sequence (91% correct). **(f)** True state-sequence.

Figure 12: Track segmentation analysis. Estimated state sequences of a HMM/A1 approach (see Sec. 2.6), are shown on the left hand side and the corresponding true state sequences are shown on the right hand side. The top sequences come from a T1 trajectory (see Sec. 2.2.1), the second ones come from a T2 trajectory (see Sec. 2.2.2) and the third ones come from a T3 trajectory (see Sec. 2.4), which is extracted from a synthetic movie using a tracking software (ImageJ). Note that (d) and (f) are identical, so that (c) and (e) can be compared directly. The T1 simulation parameters are summarised in Table 2 and the T2/T3 simulation parameters are summarised in Table 3.

3.2.2 Analysis of T2 trajectories

Next, we consider motion model simulations, wherein we model a reversible, bi-molecular reaction confined on an area (cell contact region), such that particles diffuse and may react if they "bump into" each other. Thus, we use Gillespie/T2 simulations (see Sec. 2.2.2) to generate trajectories of such a system.

A known property of Markov chains is that they exhibit exponential waiting times. As a simple way of testing whether or not the simulation waiting times are exponential, we store bound and unbound times for all A-particles. In Fig. 13, we see that both the bound times (a) and unbound times (b) are approximately exponentially distributed for our calibrated parameters. (In subfigure (b), the first bar is somewhat high, which is probably due to a rapid rebinding between the same two particles.) The good exponential fits verify that we may indeed view the particle dynamics as the result of a two-state HMM.



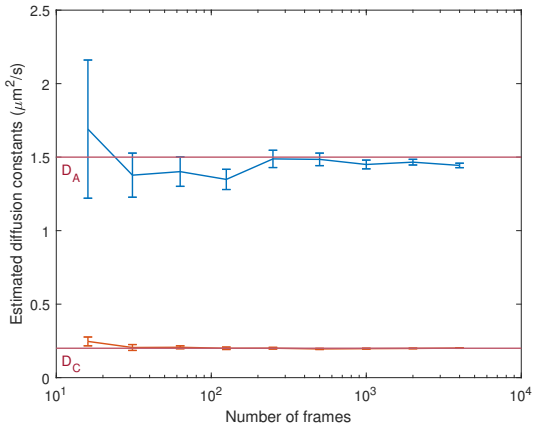
(a) A histogram of binding times from simulations. The bound times were stored for diffusing A-particles.

(b) A histogram of unbound times from simulations. The time between bindings were stored for diffusing A-particles.

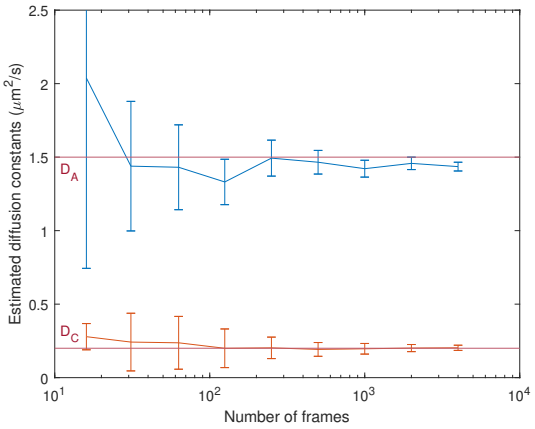
Figure 13: Waiting times from Gillespie simulations. Simulation data exhibit an exponential distribution in both (a) (bound times) and (b) (unbound times). An MLE fit of an exponential is shown together with the data. The parameters used are identical to those in Table 3; the simulations are described in detail in Sec. 2.2.2.

Here, we increase the realism of generated data, by using T2-generated trajectories. The increased realism comes from the fact that a physical system has been simulated. In particular, state transitions may now occur at *any* time, not just in-between time frames. The results from repeating the same analysis as above, is shown in Fig. 14. From Figs. 14a-14b, it is clear that both methods (A1, A2) perform very well. With more than 1000 time frames, there are no gains in estimation accuracy. Thus, the small errors involved in the analysis, e.g. due to state transitioning during a frame, set limits on the maximal obtainable precision.

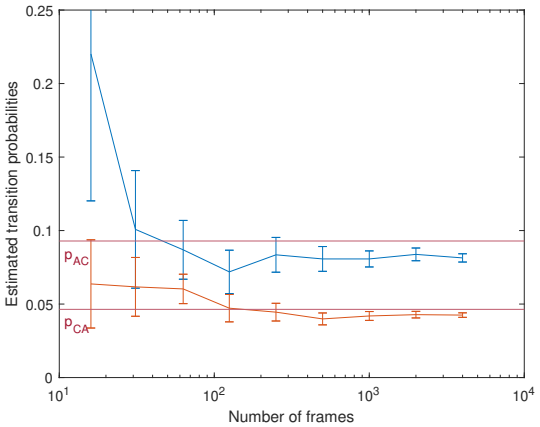
A note for the A2 analysis is that the discreteness of Gillespie simulations became visible



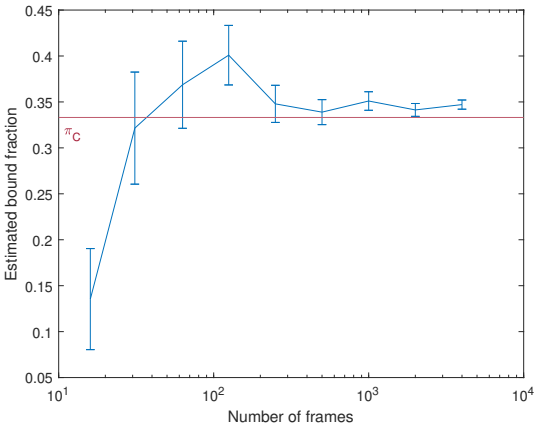
(a) A1 estimates of free (D_A) and bound (D_C) diffusion constants.



(b) A2 estimates of free (D_A) and bound (D_C) diffusion constants.



(c) A1 estimates of transition probabilities p_AC and p_AC between a free and a bound state.



(d) A2 estimates of the fraction π_C of particles that are bound at a given time.

Figure 14: Analysis of T2 trajectories. 10 tracks were generated with Gillespie simulations, as described in Sec. 2.2.2. The parameters used are summarised in Table 3. Based on the analysis of these tracks, physical parameters were inferred using a HMM/A1 approach (see Sec. 2.5.2) in (a) and (c), and a step-size distribution/A2 approach (see Sec. 2.5.3) in (b) and (d). These estimations, of diffusion constants and transition probabilities/bound fraction, are plotted as a function of time frames (i.e. track length or number of steps).

in the Rayleigh histogram, in the form of distinct spikes. The number of combinations of displacements for a typical step length in the bound state, were too few. To solve this artificial problem, we added uniformly distributed random numbers from the interval $(-a, a)$, where a is the lattice constant, to every coordinate. This way, all possible positions within a lattice site were filled in without causing a bias.

Looking at Figs. 14c-14d, the A1 analysis levels out after 250-500 time frames. Again, it seems that there is a limit to the maximal precision that we can obtain. The same holds true for the A2 analysis. Overall, there is a similarity in precision between the two methods for corresponding time frames.

The track segmentation is a bit less accurate than with T1 trajectories, being on average 92% correct. One estimated and true pair of state sequences is shown in Figs. 12c-12d. The main features are shared, but is easy to see some deviations. Presumably, the assumption of immediate state transitions explains the deviations that were not present with T1 tracks. For short tracks, other factors due to the increased realism of T2 simulations could also be of some importance, e.g. temporary crowding effects.

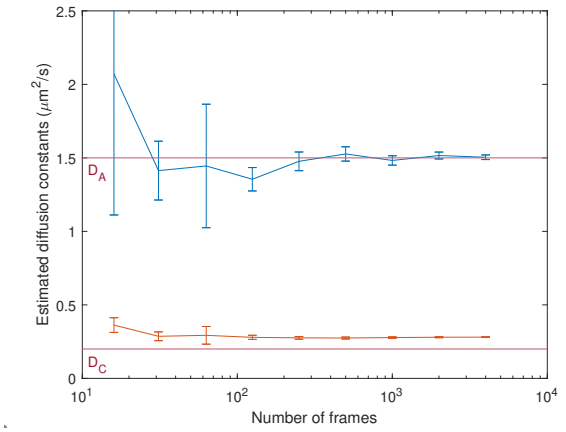
3.2.3 Analysis of T3 trajectories

Finally, we consider Gillespie-tracks in synthetic movies (see Sec. 2.4) with particle tracking done with the TrackMate plug-in in ImageJ. Clearly, we now include tracking noise in the form of dot detection and dot linking. The tracking errors also cause most tracks to be split as two dots cannot be separated. The results are presented in Fig. 15 with emitter intensity $\lambda_S = 200$ and remaining camera-parameters chosen as in Sec. 3.1. Because the trajectories are the same as the T2-trajectories in Sec. 3.2.2, all results in this section are comparable to those.

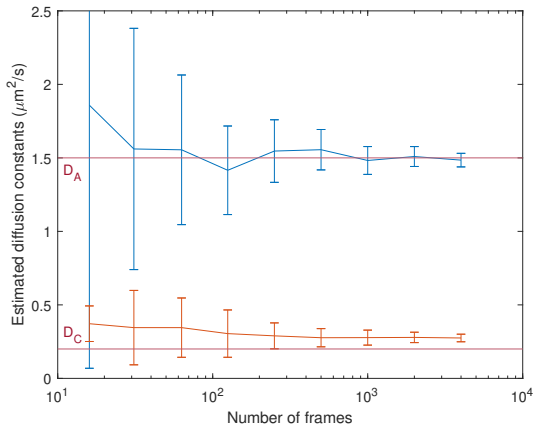
We see in Figs. 15a-15b that there are large similarities with the corresponding analysis of T2 tracks. Yet, a clear difference is that D_C levels out and does not converge to exactly the correct value. This should be due to the localisation error, which is relatively larger for shorter steps lengths. Considering the Figs. 15c - 15d, the A1 analysis actually performs somewhat better than with T2 tracks. One must keep in mind that all four HMM-parameters are linked together, which in this case may have proved beneficial. The A2 analysis on the other hand, performs a bit inferior after 1000 steps compared to the analysis of T2 tracks. Still, it is quite close to the real value.

For the track segmentation analysis, we analysed the only two intact tracks after 1000 steps. The mean correctness of these two trajectories is 90% and thus as accurately segmented as with T2 tracks. Visually, we see the difference by comparing Figs. 12c and 12e.

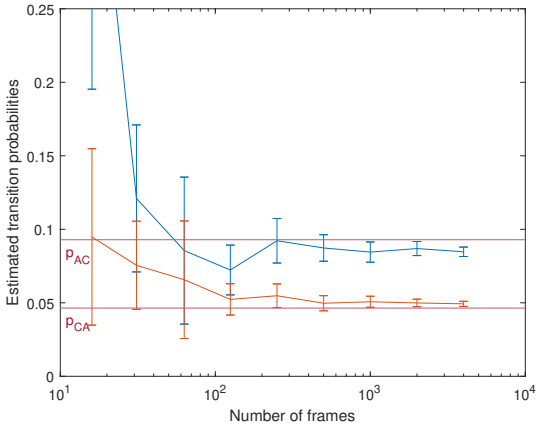
The last analysis in Fig. 15, of T3 tracks, is dependent on the emitter intensity λ_S, holding all other camera-parameters constant. In order to quantify this in some detail, we here display errors in dot detection (emitter localisation). Accordingly, we compare true dot positions to estimated ones, and associate them pairwise based on proximity. In Fig. 16a, we see the distribution of distances between true dot positions and estimated ones through tracking. It is somewhat reminiscent of Rayleigh's distribution (compare to



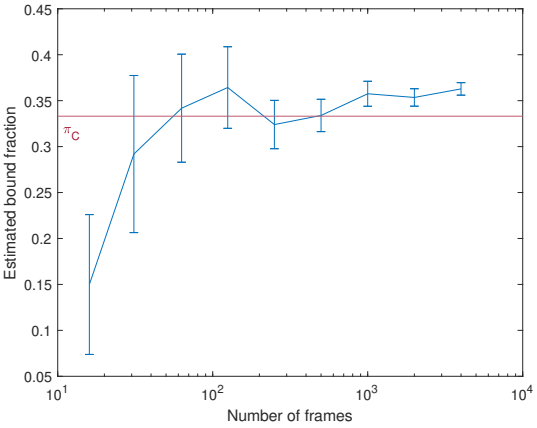
(a) A1 estimates of free (D_A) and bound (D_C) diffusion constants.



(b) A2 estimates of free (D_A) and bound (D_C) diffusion constants.

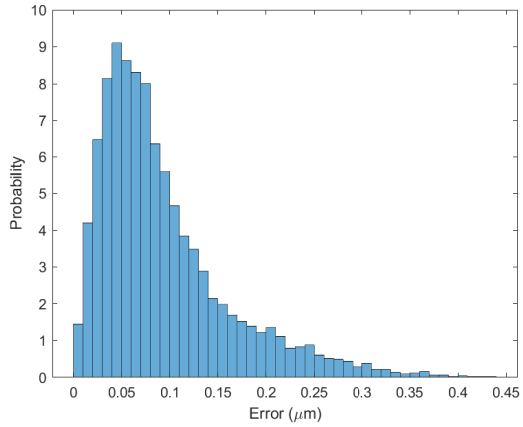


(c) A1 estimates of transition probabilities p_AC and p_AC between a free and a bound state.

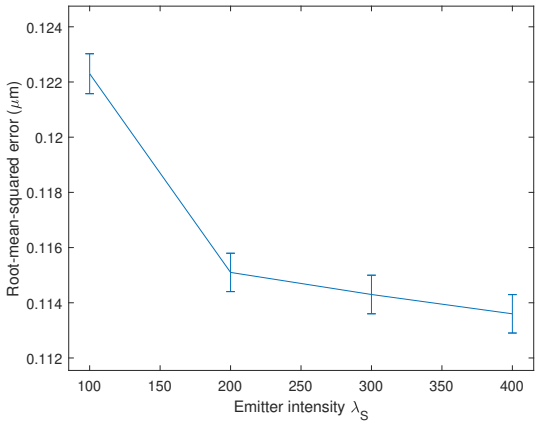


(d) A2 estimates of the fraction π_C of particles that are bound at a given time.

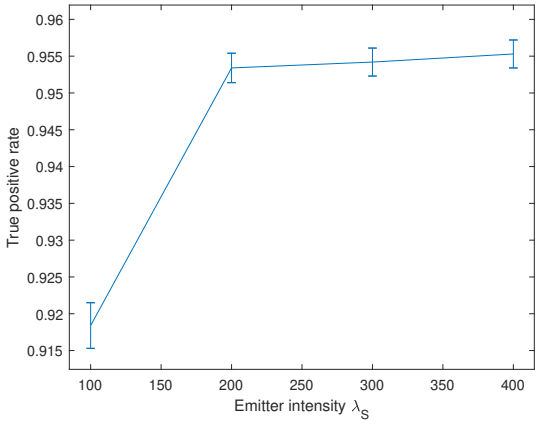
Figure 15: Analysis of T3 trajectories. 10 tracks were extracted from tracking the particles in a synthetic movie, based on Gillespie simulations (Sec. 2.2.2), and a camera response model (Sec. 2.3.3). The Gillespie parameters used are found in Table 3 and as tracking software, the TrackMate plug-in for ImageJ was used. Based on the analysis of these generated tracks, physical parameters were inferred using a HMM/A1 approach (see Sec. 2.5.2) in (a) and (c), and a step-size distribution/A2 approach (see Sec. 2.5.3) in (b) and (d). These estimations, of diffusion constants and transition probabilities/bound fraction, are plotted as a function of time frames (i.e. track length or number of steps).



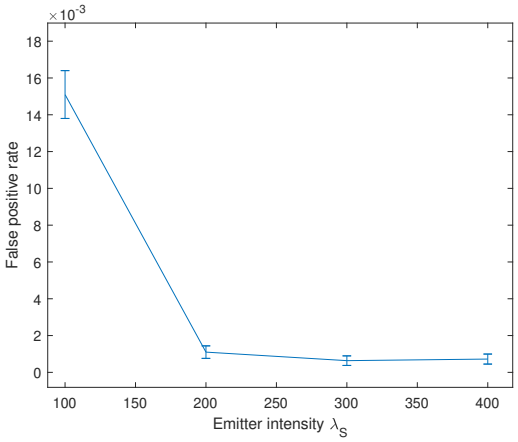
(a) Localisation errors compared to ground truth for $\lambda_S = 200$.



(b) Root-mean-squared as a function of emitter intensity.



(c) True positive rate as a function of emitter intensity.



(d) False positive rate as a function of emitter intensity.

Figure 16: Tracking assessment based on emitter intensity. Tracking localisation errors (a), root-mean-squared error for distances between dots and estimated positions (b), true positive rate (c) and false positive rate (d). The true positive rate is defined as the number of correct identifications divided by the number of particle "dots"; the false positive rate is defined as the number of incorrect identifications divided by the number of estimated positions. By producing synthetic movies which we extract tracks from via ImageJ and compare to ground truth tracks, we assess the tracking performance.

the dotted curve in Fig. 10b), which would indicate Gaussian errors in both coordinate directions. However, a Rayleigh fit turns out not to fit so well, because of outliers (see Fig. 22 in Appendix I). In Fig. 16b, the root-mean-squared (RMS) error of the distances is plotted as a function of λ_S. We see that the most gains are found in the interval $\lambda_S < 200$, where the low emitter intensity aggravates the goal of finding the centre of the dot. As a comparison, 0.11 μm = 1 pixel.

In Figs. 16c - 16d we see the true positive rate (TPR) - the number of correct identifications divided by the number of dots - and false positive rate (FPR) - the number of incorrect identifications divided by the number of estimated positions. For the lowest emitter intensity, $\lambda_S = 100$, the TPR is rather low due to the difficulty of separating dot from background. Above $\lambda_S = 200$, we reach a TPR $> 95\%$ and with minor further gains. An analogous trend is seen for the FPR, which overall is very close to zero. This is reasonable, since the probability of losing a dot should be much higher than for the background to be bright enough to be mistaken as a dot. To put emitter intensities into context, it has experimental room for improvement up to at least $\lambda_S = 250$. The highest value tested, $\lambda_S = 400$, is idealised.

4 Discussion

4.1 A1 analysis assumptions

In order for a HMM-analysis to work optimally, a few constraints need to be fulfilled. These are discussed below, one by one.

First, state switching in T2 simulations and reality may occur at any time. Still, the A1 analysis presumes that state switching only occurs in-between frames, such that there is only one state per frame [10]. Therefore, we require that

$$\max(p_{\mathrm{AC}}, p_{\mathrm{CA}}) \ll 1. \tag{4.27}$$

Recall that, by construction, switching only occurs in-between frames for T1 tracks. Consulting Eq. (2.12), we see that we need to have Q_{on} and Δt small in order to fulfil Eq. (4.27) above for T2 simulations.

Secondly, the A1 analysis presumes exponential waiting times. This is fulfilled whenever there is an excess of binding sites [10]. In turn, this should be satisfied as long as the perfect mixing assumption is fulfilled and with a relatively high substrate concentration.

Thirdly, the separation of diffusion constant magnitudes should be notable [10], with at least a factor of two difference.

4.2 A hidden diffusion constant

In a usual type of tracking experiment, only one type of particle (the A-particles) is labelled. Hence, there is no way of knowing the diffusion constant D_{B} of B-particles. This is then

a "hidden variable", that must be guessed based on properties such as particle size and crowding effects from collisions with other surface molecules.

4.3 Choice of tracking parameters

All tracking software demands some input from its user. This point is seldom highlighted, despite its important role in repeatability. This thesis is no exception, but we do know our ground truth positions. We have chosen the parameters to be reasonable based on the diffusion coefficients, and tested that it yields good results. The chosen parameters in the TrackMate plug-in were: estimated object diameter = 0.5 μm, linking max distance = 0.7 μm = gap-closing max distance, gap-closing max frame gap = 1.

4.4 Quantisation noise and quantum efficiency

In the camera model, we made two simplifications, which could be important for other cameras: disregarding a camera noise source and approximating the QE.

At the end of the camera response chain, there is a rounding to integers called "quantisation noise" [18], see Appendix F. We found it to be completely negligible in this thesis.

The QE, we simply set equal to one. In words, we are saying that the photo-diodes work perfectly. On this camera, this is motivated since the peak QE value is larger than 0.95 [24]. Yet, it is strongly dependent on wavelength and can at wavelengths equal to 200 nm or 950 nm be lower than 0.3 [24]. On cameras with inferior photo-diodes however, it may be important to include the QE also at peak values.

4.5 Motion blur

During the exposure time - the time during which the camera collects light for a given frame - incessant diffusive motion occurs. While usually very small, these displacements can be noticeable. Thus, a frame actually contains information of a mean value of scenes from reality. The collective effect of this fact is blurring, called "motion blur".

There are two main ways that we could incorporate motion blur into the synthetic movies. The simplest way is to add it after the movie has been generated. One then convolves each frame with a Gaussian, resulting in increased blurriness. A more correct way, would instead be to generate many images over time-intervals which are much shorter than the exposure time. Then, one superposes these images over the exposure time. In this thesis, we have not included motion blur.

The magnitude of motion blur is dependent on the magnitude of displacements occurring within the sampling time. Thus, for a fixed diffusion constant, effects may be visible for e.g. a 10 ms exposure time, but negligible for a 1 ms exposure time. Because this thesis mainly contains relatively short exposure times, we do not expect it to be significant here.

4.6 Photo-blinking

In this thesis, we did not include photo-blinking, whereby an emitter stops emitting photons during one or several time frames. This "noise" makes tracking, specifically dot linking, more difficult. We then expect the mean track length to decrease and the number of identified tracks to increase. Shorter tracks would likely be a larger concern for the A1 analysis (particularly track segmentation), due to its explicit calculation of transition probabilities.

5 Conclusion and outlook

In this thesis, we have constructed and tried out a testing-system for generating realistic particle tracks. There are three main steps involved in this procedure. First, we introduced a toolbox for producing synthetic fluorescence microscopy movies, by simulating reaction-diffusion processes and emulating the response of an sCMOS camera imaging system. The second step was to extract tracks by applying a standard tracking software (e.g. TrackMate in ImageJ) on these movies. Finally, we introduced another toolbox to analyse the trajectories and extract physical model parameters. For that purpose, we used two methods - HMM/A1 and a step-size distribution/A2 analysis.

Our hope is that the methods presented in this thesis can be utilised to provide insight into real experiments and into the optimal choice of analysis method. For example, one may investigate systematically under what conditions on the transition probabilities an A1 analysis yields good results. Another potential use is as training data for a neural network, to handle particle tracking.

Moreover, we hope that the overall analysis presented here will be developed further. For instance, one could include motion blur and photo-blinking into the synthetic movies. On the analysis-side, one could try to include a dot detection error into the A1 and A2 methods. We assess the realism already to be quite high and there is no obvious limit as to how realistic, and thus useful, this can be developed in the future.

References

[1] L. Sompayrac. *How The Immune System Works*. 5th edition. Wiley Blackwell, 2016.

[2] T. Dam et al. "Supported Lipid Bilayers and the Study of Two-Dimensional Binding Kinetics". In: *Frontiers in molecular biosciences* 9.833123 (2022).

[3] M. Lever et al. "Phenotypic models of T cell activation". In: *Nature reviews. Immunology* 14 (2014), pp. 619–629.

[4] J. J. Y. Lin et al. "Mapping the stochastic sequence of individual ligand-receptor binding events to cellular activation: T cells act on the rare events". In: *Science signaling* 12.564 (2019).

[5] M. Chouliara et al. "Single-cell measurements of two-dimensional binding affinity across cell contacts". In: *Biophysical journal* 120.22 (2021), pp. 5032–5040.

[6] N. Chenouard et al. "Objective comparison of particle tracking methods". In: *Nature methods* 13.3 (2019), pp. 281–290.

[7] S. Wieser et al. "Versatile Analysis of Single-Molecule Tracking Data by Comprehensive Testing against Monte Carlo Simulations". In: *Biophysical journal* 95 (2008), pp. 5988–6001.

[8] M. Axmann et al. "Determination of Interaction Kinetics between the T Cell Receptor and Peptide-Loaded MHC Class II via Single-Molecule Diffusion Measurements". In: *Biophysical journal* 103 (2012), pp. L17–L19.

[9] G. P. O'Donoghue et al. "Direct single molecule measurement of TCR triggering by agonist pMHC in living primary T cells". In: *eLife* 2 (2013).

[10] R. Das et al. "A Hidden Markov Model for Single Particle Tracks Quantifies Dynamic Interactions between LFA-1 and the Actin Cytoskeleton". In: *PLoS computational biology* 5.11 (2009).

[11] D. T. Gillespie. "Approximate accelerated stochastic simulation of chemically reacting systems". In: *Journal of chemical physics* 115.4 (2001), pp. 1716–1733.

[12] T. Ambjörnsson et al. "Single-file dynamics with different diffusion constants". In: *Journal of chemical physics* 129 (2008), p. 185106.

[13] D. T. Gillespie. "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions". In: *Journal of computational physics* 22.4 (1976), pp. 403–434.

[14] T. Aquino et al. "Chemical Continuous Time Random Walks". In: *Physical review letters* 119.23 (2017).

[15] A. C. Bebbington. "A Simple Method of Drawing a Sample Without Replacement". In: *Journal of the royal statistical society series C* 24.1 (1975).

[16] V. Junghans et al. "Effects of a local auxiliary protein on the two-dimensional affinity of a TCR–peptide MHC interaction". In: *Journal of cell science* 133.15 (2020).

[17] R. Phillips et al. *Physical Biology of the Cell*. 2nd ed. Taylor & Francis, 2012.

[18] K. Wei et al. "Physics-based Noise Modeling for Extreme Low-light Photography". In: *IEEE transactions on pattern analysis and machine intelligence* (2021).

[19] F. Huang et al. "Video-rate nanoscopy using sCMOS camera–specific single-molecule localization algorithms". In: *Nature methods* 10.7 (2013), pp. 653–658.

[20] National institute of science and technology. *Probability Plot*. Accessed: 2022-07-01. URL: `https://itl.nist.gov/div898/handbook/eda/section3/probplot.htm`.

[21] W.H. Press et al. *Numerical Recipes: The Art of Scientific Computing*. 3rd edition. Cambridge University Press, 2007.

[22] *Prime 95B Manual*. Teledyne Photometrics. Arizona, USA.

[23] National institute of science and technology. *Probability Plot Correlation Coefficient Plot*. Accessed: 2022-07-09. URL: `https://itl.nist.gov/div898/handbook/eda/section3/ppccplot.htm`.

[24] *Prime 95B Scientific CMOS Camera Datasheet*. Arizona, USA: Teledyne Photometrics.

[25] R.M. Mazo. *Brownian Motion - Fluctuations, Dynamics, and Applications*. Oxford Science Publications, 2008.

[26] K.F. Riley et al. *Mathematical Methods For Physics And Engineering*. 3rd edition. United Kingdom: Cambridge University Press, 2006.

[27] F. Huang et al. "Simultaneous multiple-emitter fitting for single molecule super-resolution imaging". In: *Biomedical optics express* 2.5 (2011), pp. 1377–1393.

[28] *Prime 95B Customer Applications Handbook*. Teledyne Photometrics. Arizona, USA.

[29] M. Hirsch et al. "A Stochastic Model for Electron Multiplication Charge- Coupled Devices – From Theory to Practice". In: *PLoS one* 8.1 (2013).

[30] B. L. Joiner et al. "Some Properties of the Range in Samples from Tukey's Symmetric Lambda Distributions". In: *Journal of the American statistical association* 66.334 (1971), pp. 394–399.

[31] J. Gil-Pelaez. "Note on the inversion theorem". In: *Biometrika* 38.3/4 (1951), pp. 481–482.

[32] V. Witkovský. "Numerical inversion of a characteristic function: An alternative tool to form the probability distribution of output quantity in linear measurement models". In: *Acta IMEKO* 5.3 (2016), pp. 32–44.

A The diffusion equation

Let us use a path integral approach to microscopically derive the two-dimensional joint probability density for a particle displacement during a time interval. Bearing that in mind, note that the observed motion of a diffusing particle is the result of a tremendous number of microscopic collisions with the surrounding fluid molecules (first suggested by Smoluchowski [25]). We may therefore think of diffusion as the result of independent, random collisions. Suppose the particle starts at position $\mathbf{r}_0$ at time t_0 and ends at a nearby position $\mathbf{r}_s$ at time t_s just a little later. Suppose further that the probability density for this small displacement is

$$u(\mathbf{r}_s, t_s|\mathbf{r}_0, t_0) = \frac{1}{4\pi D(t_s - t_0)} \exp\big(-\frac{|\mathbf{r}_s - \mathbf{r}_0|^2}{4D(t_s - t_0)}\big). \tag{A.28}$$

Now, we wish to find the corresponding PDF of observing a displacement $\mathbf{r}$ after at a time $\Delta t > t_s - t_0$. Let $t_0 = 0$ for simplicity. Then, divide up the time into n steps (n large)

$$t_k = k\epsilon, \quad k = 0, 1, 2, ..., n-1, \tag{A.29}$$

so that $\Delta t = t_n$ and $\mathbf{r} = \mathbf{r}_n$. Between every time step, the particle trajectory may be any path connecting the latest position $\mathbf{r}_k$ to the next position $\mathbf{r}_{k+1}$. These positions are considered to be fixed, so that a path is a sequence of fixed positions. Assuming that all the incremental steps are independent, the probability density of a path is

$$\omega(\mathbf{r}_0, \mathbf{r}_1, ..., \mathbf{r}_n|t_n) = \prod_{k=0}^{n-1} u(\mathbf{r}_{k+1}, t_{k+1}|\mathbf{r}_k, t_k) \tag{A.30}$$

$$= \prod_{k=0}^{n-1} \frac{1}{4\pi D\epsilon} \exp\big(-\frac{|\mathbf{r}_{k+1} - \mathbf{r}_k|^2}{4D\epsilon}\big) \tag{A.31}$$

$$= \frac{1}{(4\pi D\epsilon)^n} \exp\big(-\sum_{k=0}^{n-1} \frac{|\mathbf{r}_{k+1} - \mathbf{r}_k|^2}{4D\epsilon}\big). \tag{A.32}$$

Note that Eq. (A.31) is central to the HMM/A1 parameters estimation method. In the language of HMMs, every factor in the product is an "emission probability"; with access to track displacements we can draw probabilistic conclusions about the motion model parameters.

The probability density of $\mathbf{r}_n$ at time t_n, is obtained by integrating over all possible paths (integrating out all intermediates):

$$u(\mathbf{r}_n, \Delta t|\mathbf{r}_0) = \int_{-\infty}^{\infty} d^2 r_1 ... \int_{-\infty}^{\infty} d^2 r_{n-1} \frac{1}{4\pi D\epsilon} \exp\big(-\frac{|\mathbf{r}_1 - \mathbf{r}_0|^2 + ... + |\mathbf{r}_n - \mathbf{r}_{n-1}|^2}{4D\epsilon}\big), \tag{A.33}$$

where we use the notation $d^2r = dxdy$. For every integral to be evaluated, there will be two exponential terms involved. All these can be evaluated by using the identity

$$\int_{-\infty}^{\infty} d^2r_k \exp\left(-a|\mathbf{r}_k - \mathbf{r}_{k-1}|^2 + b|\mathbf{r}_{k-1} - \mathbf{r}_{k-2}|^2\right) = \frac{\pi}{a+b} \exp\big(-\frac{ab}{a+b}|\mathbf{r}_k - \mathbf{r}_{k-2}|^2\big), \tag{A.34}$$

where a, b are constants. Thus, for the first integral, we have that

$$\int_{-\infty}^{\infty} d^2r_{n-1} \exp\big(-\frac{|\mathbf{r}_k - \mathbf{r}_{k-1}|^2 + b|\mathbf{r}_{k-1} - \mathbf{r}_{k-2}|^2}{4D\epsilon}\big) = \frac{1}{2} 4\pi\epsilon D \exp\big(-\frac{|\mathbf{r}_n - \mathbf{r}_{n-2}|^2}{2 \cdot 4D\epsilon}\big) \tag{A.35}$$

and therefore Eq. (A.33) becomes

$$\begin{aligned} u(\mathbf{r}_n, \Delta t|\mathbf{r}_0) = \frac{1}{2} \frac{1}{(4\pi D\epsilon)^{n-1}} \int_{-\infty}^{\infty} d^2r_1 ... \\ \int_{-\infty}^{\infty} d^2r_{n-3} \frac{1}{4\pi D\epsilon} \exp\big(-\frac{|\mathbf{r}_1 - \mathbf{r}_0|^2 + ... + |\mathbf{r}_{n-3} - \mathbf{r}_{n-4}|^2}{4D\epsilon}\big) \\ \int_{-\infty}^{\infty} d^2r_{n-2} \exp\big(-\frac{|\mathbf{r}_{n-2} - \mathbf{r}_{n-3}|^2}{4D\epsilon} - \frac{|\mathbf{r}_n - \mathbf{r}_{n-2}|^2}{2 \cdot 4D\epsilon}\big). \end{aligned} \tag{A.36}$$

By utilising Eq. (A.34) again, we have

$$\int_{-\infty}^{\infty} d^2r_{n-2} \exp\big(-\frac{|\mathbf{r}_{n-2} - \mathbf{r}_{n-3}|^2}{4D\epsilon} - \frac{|\mathbf{r}_n - \mathbf{r}_{n-2}|^2}{2 \cdot 4D\epsilon}\big) = \frac{1}{3} 2 \cdot 4\pi\epsilon D \exp\big(-\frac{|\mathbf{r}_n - \mathbf{r}_{n-3}|^2}{3 \cdot 4D\epsilon}\big), \tag{A.37}$$

so that Eq. (A.36) simplifies to

$$\begin{aligned} u(\mathbf{r}_n, \Delta t|\mathbf{r}_0) = \frac{1}{3} \frac{1}{(4\pi D\epsilon)^{n-2}} \int_{-\infty}^{\infty} d\mathbf{r}_1 ... \\ \int_{-\infty}^{\infty} d^2r_{n-4} \frac{1}{4\pi D\epsilon} \exp\big(-\frac{|\mathbf{r}_1 - \mathbf{r}_0|^2 + ... + |\mathbf{r}_{n-5} - \mathbf{r}_{n-4}|^2}{4D\epsilon}\big) \\ \int_{-\infty}^{\infty} d^2r_{n-3} \exp\big(-\frac{|\mathbf{r}_{n-3} - \mathbf{r}_{n-4}|^2}{4D\epsilon} - \frac{|\mathbf{r}_n - \mathbf{r}_{n-3}|^2}{3 \cdot 4D\epsilon}\big). \end{aligned} \tag{A.38}$$

By continuing these integral evaluations, we end up with

$$u(\mathbf{r}_n, \Delta t|\mathbf{r}_0) = \frac{1}{n \cdot 4\pi D\epsilon} \exp\big(-\frac{|\mathbf{r}_n - \mathbf{r}_0|^2}{n \cdot D\epsilon}\big) \tag{A.39}$$

and by reintroducing $\Delta t = t_n = n\epsilon$ and $\mathbf{r} = \mathbf{r}_n$, we have the final result

$$u(\mathbf{r}, \Delta t|\mathbf{r}_0) = \frac{1}{4\pi D\Delta t} \exp\big(-\frac{|\mathbf{r}-\mathbf{r}_0|^2}{4D\Delta t}\big). \tag{A.40}$$

Some remarks about the solution in Eq. (A.40) can be mentioned. Importantly, it can be shown to satisfy the diffusion equation (also known as the heat equation):

$$\frac{\partial u(\mathbf{r}, t)}{\partial t} = D\nabla^2 u(\mathbf{r}, t). \tag{A.41}$$

By adopting a statistical viewpoint, we can regard solution (A.40) as a (Gaussian) distribution of particles. As such, we can calculate (or directly read off) its mean and variance. The results are

$$\langle|\mathbf{r}-\mathbf{r}_0|\rangle = 0 \tag{A.42}$$

and

$$\langle|\mathbf{r}-\mathbf{r}_0|^2\rangle = 4Dt. \tag{A.43}$$

The mean centred on zero is expected; the variance increasing linearly with t is a characteristic of diffusion processes. It implies that a particle's typical displacement in a time interval t is proportional to $\sqrt{t}$.

B Gillespie simulations

We here give an overview of Gillespie simulations, with information collected mostly from [13]. In short, a Gillespie simulation is a stochastic simulation method, which was originally designed for modelling chemical reactions. Thanks to its stochastic nature, it naturally incorporates fluctuations that are not present in deterministic models (which are usually formulated as a set of coupled ordinary differential equations for the concentrations of the molecular species at each instant of time, c.f. Eqs. (C.1) in Appendix C). The Gillespie algorithm is equivalent to the solution of a spatially homogeneous probabilistic master equation, of which its solution describes the probability of finding molecular populations at each instant in time. In a probabilistic framework, reaction constants are interpreted as reaction probabilities per unit time. Needless to say, most realistic systems lack an analytic solution to the master equation. So apart from an assumption of homogeneity, it is a very general method with far-reaching applicability.

The centrepiece quantity in a Gillespie simulation is the "reaction probability density" (rPDF) P. The probability that at time t, the next reaction will happen in differential time interval $(t+\tau, t+\tau+d\tau)$ and be the particular reaction R_μ is $Pd\tau$. In the Gillespie method, one assumes the following form of the rPDF:

$$P(\tau, \mu) = k_\mu \exp\,(-\sum_{\nu=1}^{M} k_\nu \tau). \tag{B.1}$$

τ is the waiting time between events, M is the number of reactions and k_μ is the average probability per time that an R_μ reaction will occur somewhere inside the volume (or area in our case). Note that in contrast to the original intention of the Gillespie method, "a reaction" here is any possible event (i.e. hop diffusion or binding/unbinding) that has a rate associated with it.

$P(\tau, \mu)$ above can be conditioned into a probability P_1 for a reaction happening in the time interval $(t+\tau, t+\tau+d\tau)$, and a probability P_2 that the next reaction will be of type R_μ, given that it will happen during $(t+\tau, t+\tau+d\tau)$:

$$P(\tau, \mu) = P_1(\tau)P_2(\mu|\tau). \tag{B.2}$$

Then, marginalising over μ,

$$P_1(\tau) = \sum_{\mu=1}^{M} P(\tau, \mu) \tag{B.3}$$

and combining Eqs. (B.2) and (B.3),

$$P_2(\mu|\tau) = \frac{P(\tau, \mu)}{\sum_{\nu=1}^{M} P(\tau, \nu)}. \tag{B.4}$$

Rewriting P_1 and P_2 in terms of $k = \sum_{\nu=1}^{M} k_\nu$, we now have

$$P_1(\tau) = ke^{-k\tau}, \quad 0 \leq \tau \leq \infty \tag{B.5}$$

$$P_2(\mu|\tau) = \frac{k_\mu}{k}, \quad \mu = 1, 2, ..., M. \tag{B.6}$$

The time between reaction events is exponentially distributed with mean $1/k$ (Eq. (B.5)), and the probability for a given reaction is proportional to its reaction rate (Eq. (B.6)).

The Gillespie algorithm simulates the stochastic process described by the reaction probability density. Doing a single Gillespie simulation amounts to a single realisation of Eq. (B.1). By carrying out several independent realisations with the same initial conditions, we obtain a statistically correct temporal evolution of the system [13].

C Reaction simulations

As en extra method of verifying the results of several runs of stochastic Gillespie simulations, we numerically solve the corresponding set of deterministic rate equations for concentrations. These assume perfect mixing at all times, which implies that the system is in a reaction-limited regime. Let c_{AB} denote the concentration of formed complexes, and c_{A} (c_{B}) denote the concentrations of A- (B-) particles. The system of equations that describe the evolution of each concentration is then

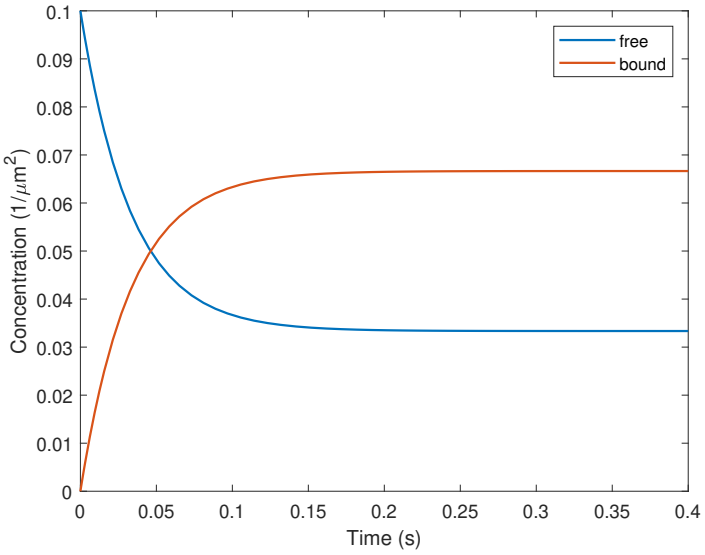


Figure 17: Time evolution of reacting molecular concentrations. Free and bound A-particle concentrations are plotted as a function of time with parameters $k_{\text{on}} = 0.20\ \frac{(\mu\text{m})^2}{s}$, $k_{\text{off}} = 0.10/\text{s}$, $c_{\text{A}} = 0.1\ \frac{\text{molecules}}{(\mu\text{m})^2}$, $c_{\text{B}} = 100\ \frac{\text{molecules}}{(\mu\text{m})^2}$ and $c_{\text{AB}} = 0.0\ \frac{\text{molecules}}{(\mu\text{m})^2}$. Eqs. (C.1) were solved numerically with a Runge-Kutta integration scheme. The B-particle concentration is suppressed for clarity.

$$\begin{cases} \dfrac{\mathrm{d}c_{\text{AB}}(t)}{\mathrm{d}t} = -k_{\text{off}}c_{\text{AB}}(t) + k_{\text{on}}c_A(t)c_B(t) \\ \dfrac{\mathrm{d}c_A(t)}{\mathrm{d}t} = k_{\text{off}}c_{\text{AB}}(t) - k_{\text{on}}c_A(t)c_B(t) \\ \dfrac{\mathrm{d}c_B(t)}{\mathrm{d}t} = k_{\text{off}}c_{\text{AB}}(t) - k_{\text{on}}c_A(t)c_B(t). \end{cases} \tag{C.1}$$

We see that the rate of change of each concentration is dependent on the current concentrations of all molecular species. On the RHS of each equation above, the first term considers the dissociation of an AB-complex; the second term considers the formation of an AB-complex. The system of equations, Eqs. (C.1), lets us test dependence of later concentrations on initial concentrations and assess the time it takes to reach equilibrium. In Fig. 17 for example, we see that equilibrium occurs after about 0.3 s.

D Derivation of Rayleigh's distribution

Let us derive the distribution for the step length of a 2-dimensional random walk. This distribution is put to use for the A1 analysis in Sec. 2.5.3 in the main text. Let ΔX be a random variable with probability density function (PDF)

$$f(\Delta x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\big(-\frac{1}{2}\big(\frac{\Delta x - \mu}{\sigma}\big)^2\big)$$
$$\mu = 0 \text{ and } \sigma = \sqrt{2D\Delta t} \implies$$
$$= f(\Delta x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\big(-\frac{(\Delta x)^2}{2\sigma^2}\big)$$

and similarly ΔY be another, independent, random variable with PDF

$$f(\Delta y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\big(-\frac{(\Delta y)^2}{2\sigma^2}\big).$$

Form a new stochastic variable, $S(\Delta X, \Delta Y)$, with values

$$s = \sqrt{(\Delta x)^2 + (\Delta y)^2}.$$

Its PDF, call it $\phi(s)$, can be found from those of ΔX and ΔY. By independence, their joint PDF is

$$p(\Delta x, \Delta y) = f(\Delta x) f(\Delta y)$$
$$= \frac{1}{2\pi\sigma^2} \exp\big(-\frac{(\Delta x)^2 + (\Delta y)^2}{2\sigma^2}\big).$$

Therefore,

$$\phi(s) = \int_{-\infty}^{\infty} d\Delta x \int_{-\infty}^{\infty} d\Delta y \; \delta(s - \sqrt{(\Delta x)^2 + (\Delta y)^2}) p(\Delta x, \Delta y).$$

Here, we have introduced the delta-function, so that for every possible value of p, we pick out the corresponding step length s. Now, transform to plane polar coordinates

$$r = \sqrt{(\Delta x)^2 + (\Delta y)^2}, \quad \theta = \arctan(\frac{\Delta y}{\Delta x})$$
$$d\Delta x d\Delta y = r dr d\theta$$

so that

$$\phi(s) = \int_0^{\infty} r dr \int_0^{2\pi} d\theta \; \delta(s - r) \frac{1}{2\pi\sigma^2} \exp\big(-\frac{r^2}{2\sigma^2}\big)$$
$$= \frac{s}{\sigma^2} \exp\big(-\frac{s^2}{2\sigma^2}\big),$$

i.e.

$$\phi(s) = \frac{s}{2D\Delta t} \exp\big(-\frac{s^2}{4D\Delta t}\big). \tag{D.1}$$

Eq. (D.1) is called "Rayleigh's distribution". It shares some similarities with the Gaussian distributions from which it is "built", e.g. that it goes to zero in both limits of s. An important difference, however, is that is is not symmetric, but has a long tail.

E Photon counts at a pixel

E.1 Point spread function

No apparatus has an infinite resolution. As a consequence, the distribution which we want to measure (the true distribution) will be distorted and thus limited by the apparatus' resolution function. In mathematical terms, such a resolution function is therefore not a δ-function, but has a finite width. The observed distribution $h(z)$ can be shown to be the convolution of the true function $f(x)$ and the resolution function $g(y)$:

$$h(z) = \int_{-\infty}^{\infty} f(x)g(z-x)\ dx \tag{E.1}$$

as demonstrated in e.g. [26]. This can also be seen from the fact that the PDF of a sum of two independent random variables is given by their convolution.

For a microscope, the resolution function is called the "point spread function" (PSF). It is the impulse response function of a microscope to a point source of light [27], of a certain wavelength. A point source emitter will then not look like a point source, but as a "smeared out" blob. In 2D, this smearing effect is approximately characterised by the Gaussian distribution [27]:

$$\mathrm{PSF}(x,y) = \frac{1}{2\pi\sigma_{\mathrm{PSF}}^2} \exp\left(-\frac{(x-x_0)^2 + (y-y_0)^2}{2\sigma_{\mathrm{PSF}}^2} \right), \tag{E.2}$$

where (x, y) is the evaluation point, (x_0, y_0) is the emitter position and σ_{PSF} is its standard deviation. $\mathrm{PSF}(x, y)$ is then a two-dimensional version of $g(y)$. Like most resolution functions, it is centred and symmetric around the true value. Its key property is its width, determined by σ_{PSF}. Note also, that σ_{PSF} is dependent on the wavelength of light and therefore, so is the PSF.

E.2 Photon count probability

Let us calculate the probability that the photons from an emitter hits a given pixel. To that end, consider a photon emitted from a given point source, (x_0, y_0), in the camera focus. The probability that the photon hits a certain pixel can be calculated by integrating the PSF over that pixel. Suppose furthermore that the pixels are square-shaped with side

length 1 pixel. Let pixel i have centre (x_m, y_m) and denote its corresponding probability by $P(i)$. Then, we find that

$$\begin{aligned} P(i) &= \int_{x_m-\frac{1}{2}}^{x_m+\frac{1}{2}} dx \int_{y_m-\frac{1}{2}}^{y_m+\frac{1}{2}} dy \ \mathrm{PSF}(x,y) \\ &= \int_{x_m-\frac{1}{2}}^{x_m+\frac{1}{2}} dx \exp\bigg(-\frac{(x-x_0)^2}{2\sigma_{\mathrm{PSF}}^2}\bigg) \int_{y_m-\frac{1}{2}}^{y_m+\frac{1}{2}} dy \exp\bigg(-\frac{(y-y_0)^2}{2\sigma_{\mathrm{PSF}}^2}\bigg) \end{aligned}$$

making a change of variables $t^2 = \frac{(x-x_0)^2}{2\sigma^2}$ yields

$$\begin{aligned} &\int_{x_m-\frac{1}{2}}^{x_m+\frac{1}{2}} dx \exp\bigg(-\frac{(x-x_0)^2}{2\sigma_{\mathrm{PSF}}^2}\bigg) = \sqrt{2}\sigma \int_{\frac{x_m-x_0-\frac{1}{2}}{\sqrt{2}\sigma}}^{\frac{x_m-x_0+\frac{1}{2}}{\sqrt{2}\sigma}} dt \exp\left(-t^2\right) \\ &= \sqrt{2}\sigma\frac{\sqrt{\pi}}{2}\bigg(\frac{2}{\sqrt{\pi}}\int_0^{\frac{x_m-x_0+\frac{1}{2}}{\sqrt{2}\sigma}} dt \exp\left(-t^2\right) - \frac{2}{\sqrt{\pi}}\int_0^{\frac{x_m-x_0-\frac{1}{2}}{\sqrt{2}\sigma}} dt \exp\left(-t^2\right)\bigg) \\ &= \frac{\sigma\sqrt{\pi}}{\sqrt{2}}\bigg(\mathrm{erf}\bigg(\frac{x_m-x_0+\frac{1}{2}}{\sqrt{2}\sigma}\bigg) - \mathrm{erf}\bigg(\frac{x_m-x_0-\frac{1}{2}}{\sqrt{2}\sigma}\bigg)\bigg) \end{aligned}$$

By treating the y-integral in a similar manner, we obtain

$$P(i) = \frac{1}{4}\bigg(\mathrm{erf}\bigg(\frac{x_m-x_0+\frac{1}{2}}{\sqrt{2}\sigma}\bigg) - \mathrm{erf}\bigg(\frac{x_m-x_0-\frac{1}{2}}{\sqrt{2}\sigma}\bigg)\bigg)\bigg(\mathrm{erf}\bigg(\frac{y_m-y_0+\frac{1}{2}}{\sqrt{2}\sigma}\bigg) - \mathrm{erf}\bigg(\frac{y_m-y_0-\frac{1}{2}}{\sqrt{2}\sigma}\bigg)\bigg) \tag{E.3}$$

which is our final result and agrees with [27]. erf(x) here is the "error function" defined by

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \exp\left(-t^2\right)\, dt. \tag{E.4}$$

F Step-by-step image modelling

In this appendix, we describe a means of generating synthetic images, based on an sCMOS (scientific complementary metal–oxide–semiconductor) camera. CMOS is a newer camera architecture then EMCCD (electron-multiplying charge-coupled device), in particular the subtype sCMOS. This subtype is suitable for low-light conditions such as fluorescence microscopy. The output will be an image count, based on the sum $gX + gS + Y + Z + o$, partly like in [18]. Here, the new variables S and Z denote the emitter photon counts and rounding errors, respectively.

F.1 Photon counts

We begin by considering "background" photons, i.e. light from outside our experiment of interest. Due to quantum uncertainty, there is a varying photon flux. Therefore, an image

sensor pixel will receive photons at random during a given time interval. Note that this noise is camera-unrelated and ineluctable.

The resulting distribution of photon counts during exposure time is known to be Poissonian, see Eq. (H.1). This makes sense, given that sufficient assumptions for a Poisson distribution are fulfilled: the events occur independently of one another and the mean event rate is independent of any occurrences (and two events cannot occur at the exact same instant). Therefore, we set the background counts to

$$X \sim \mathcal{P}(\lambda_{\text{Bg}}), \tag{F.1}$$

where $\mathcal{P}(\lambda_{\text{Bg}})$ denotes the Poissonian distribution with parameter λ_{Bg} - the expected number of background photons hitting a given pixel during the exposure time. By the symbol $\sim$ we mean "is a random number drawn from the distribution"

Now, let us consider photon counts due to fluorophores. We realise that the same assumptions for a Poisson distribution still holds. Therefore, just as for background counts, the fluorophore-emitted photons will be Poisson distributed. But in contrast to background counts, these "signal" counts are based upon the location of the fluorophores. In order to relate this location to that of a given pixel, we make use of the results in Sec. E.2. Combining these two facts, we obtain

$$S \sim \mathcal{P}(\lambda_{\text{S}} P(i)), \tag{F.2}$$

where λ_{S} denotes the expected number of fluorophore photons hitting a given pixel i during exposure time and $P(i)$ is the probability for a given photon to hit pixel i.

In practice, when generating the number of incoming photons onto a pixel, we consider one pixel at a time. All, if any, emitters that are close enough to the current pixel will generate a photon count. There is also always a background count for each pixel. However, the number of photons at a pixel cannot in reality be recorded directly as photon counts. Rather, the camera response to these photons introduces further noise and effects as discussed in the next in sections.

F.2 Photons to electrons

In an ideal camera, each incident photon generates a fixed number of electrons, called "photo-electrons". The physical mechanism utilised in doing so is the photoelectric effect. Camera photo-diodes convert photons into photo-electrons; the conversion factor is the quantum efficiency, QE. It accounts for the imperfect work done by the photodiodes in converting photons to electrons and is defined as the ratio of the average number of electrons generated in a pixel, to the number of incident photons on that pixel during exposure time.

Because the wavelength of light is proportional to its energy ($E = hc/\lambda$), QE is wavelength-dependent. This dependence is camera-specific and should be provided by the manufacturer - in our case it can be found in [24]. However, for simplicity, we here assume monochromatic light. This should be a very good approximation, since the fluorophores are identical and subject to the same environment.

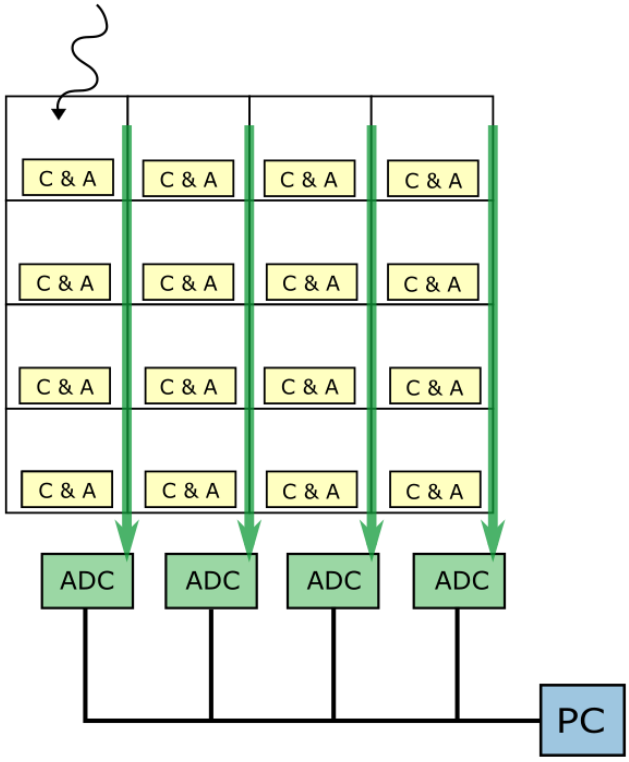


Figure 18: Schematic of CMOS camera architecture. Light hits a pixel and is converted to a voltage, that passes down that column were it is converted into digital counts and read out into a PC. Each pixel has its own condensator and amplifier (C&A) and every column has its own analog-to-digital converter (ADC). Image inspired from [28].

Simply by multiplying the expected photon counts with the QE, we obtain the number of photo-electrons directly from Eqs. (F.1)-(F.2):

$$X_e \sim \mathcal{P}(\mathrm{QE} \cdot \lambda_{\mathrm{Bg}}) \tag{F.3}$$
$$S_e \sim \mathcal{P}(\mathrm{QE} \cdot \lambda_{\mathrm{S}} \cdot P(i)). \tag{F.4}$$

F.3 Electrons to voltage

The CMOS camera architecture has a specific read-out structure. In contrast to a CCD-camera, each pixel has its own capacitor and amplifier, that handle the conversion of charge to voltage [28]. Moreover, every column of pixels share the same analog-to-digital converter (ADC), see Fig. 18. These ADCs are in turn connected to a computer. It is thanks to this design that the CMOS read-out speed of is faster than that of CCD.

Another consequence of the architectural differences, is that the the read noise Y of

CMOS-cameras will not become normally distributed as for CCD-cameras [29]. Rather, it will feature longer tails. We will approximate this distribution with a distribution from the Tukey-Lambda ($TL()$) family of distributions, as done in [18]. This family of distributions can approximate a wide range of symmetrical distributions.

To generate TL-random numbers, we regard its definition. Since it is a family of wildly different distributions, there is no single PDF in its definition. We have that the random numbers drawn fulfil

$$TL(\lambda_{\mathrm{TL}}; \mu_{\mathrm{TL}}, \sigma_{\mathrm{TL}}) = \mu_{\mathrm{TL}} + \sigma_{\mathrm{TL}} \frac{R^{\lambda_{\mathrm{TL}}} - (1-R)^{\lambda_{\mathrm{TL}}}}{\lambda_{\mathrm{TL}}}, \tag{F.5}$$

where λ_{TL} is the shape parameter that determines the specific distribution and R is a uniformly generated random number from $[0, 1]$ [30]. We have here also introduced a location parameter μ_{TL} and a scale parameter σ_{TL}.

Rectangular (pseudo)-random numbers are available directly in computational programmes such as MATLAB, and so we can use Eq. (F.5) to directly generate TL random numbers. We assume that this distributions is centred on zero, i.e. $\mu_{\mathrm{TL}} = 0$. To summarise,

$$Y \sim TL(\lambda_{\mathrm{TL}}; 0, \sigma_{\mathrm{TL}}). \tag{F.6}$$

Alternatively, the heavy tails can be modelled by a logistic distribution:

$$Y \sim L(0, \sigma_{\mathrm{TL}}), \tag{F.7}$$

see Eq. (H.3) for its PDF. This distribution is contained within the Tukey-lambda family and can be described as "a heavy-tailed Gaussian" distribution. Whether this specific distribution is a good approximation depends on the specific camera used; for the Teledyne Prime 95 B considered in this thesis, it is an excellent approximation.

F.4 Voltage to digital numbers

There are round-off errors involved in converting voltage to digital numbers in a camera. We call this the "quantisation noise" and write

$$Z \sim U(-\frac{1}{2}, \frac{1}{2}), \tag{F.8}$$

where U denotes the uniform distribution. If an image count has a value of 80.4, the rounding subtracts 0.4. We assume that the first decimal of every non-rounded count should be almost equally likely. Therefore, modelling the rounding in this way makes sense, at least for a camera that rounds its counts to the nearest integer. Because this rounding error is so small however, we neglect it in this thesis. In other words, we use a "continuum approximation" for the final image counts.

G Characteristic function

The characteristic function (CF) of a distribution is a useful construct when dealing with sums of random variables. For a continuous random variable, the CF is the Fourier transformation of its PDF:

$$\Phi(k) = \mathcal{F}[p(x)](k) = \int_{-\infty}^{\infty} dx\ p(x)e^{ikx} = \langle e^{ikx} \rangle, \tag{G.1}$$

where $p(x)$ is the PDF of a random variable X, $< X >$ denotes the average of X and k is the Fourier variable. The last equality is taken to be the definition of a CF, and also holds for a discrete distribution.

A key theorem, the "inversion theorem", states that the CF of a given PDF is unique [31]. Hence, we may first calculate the CF and then Fourier-invert it back to its corresponding PDF. In the general case $Z = f(X, Y)$, we have that

$$\Phi_Z(k) = \langle e^{ikf(X,Y)} \rangle. \tag{G.2}$$

If furthermore $f(X, Y) = X + Y$, and X and Y are independent, we directly obtain

$$\Phi_Z(k) = \langle e^{ik(X+Y)} \rangle = \langle e^{ikX} e^{ikY} \rangle = \langle e^{ikX} \rangle \langle e^{ikY} \rangle = \Phi_X(k)\Phi_Y(k). \tag{G.3}$$

Like so, the CF of a sum of independent random variables factorise into the CFs of the individual random variables. In the end, we calculate the inverse Fourier transform

$$p(z) = \mathcal{F}^{-1}[\Phi_Z(k)](z) = \frac{1}{2\pi} \int_{-\infty}^{\infty} dk\ \Phi_Z(k)e^{-ikz} \tag{G.4}$$

to obtain the final PDF of our random variable Z. If we now use Eq. (G.3), the final result is

$$p(z) = \mathcal{F}^{-1}[\Phi_Z(k)](z) = \frac{1}{2\pi} \int_{-\infty}^{\infty} dk\ \Phi_X(k)\Phi_Y(k)e^{-ikz} \tag{G.5}$$

and is thus given directly in terms of the individual CFs. The expression for $p(z)$ can be reformulated [32], into a form known as Gil-Pelaez inversion formula:

$$p(z) = \frac{1}{\pi} \int_{0}^{\infty} dk\ \mathfrak{R}(\Phi(k)e^{-ikz}), \tag{G.6}$$

where $\mathfrak{R}$ denotes the real part of the expression inside brackets. The merit of this formula is that it is well formulated for numerical quadrature.

H Relevant distributions

The CFs corresponding to the relevant noise distributions (see Appendix F) are well-known and can be calculated analytically with Eq. (G.1) in Appendix G.

H.0.1 Poissonian CF

The CF of a Poissonian distribution

$$f(x) = e^{-\lambda}\frac{\lambda^x}{x!}, \quad x = 0, 1, 2, 3... \tag{H.1}$$

is

$$\Phi(k) = \exp\left(\lambda(e^{ik} - 1)\right). \tag{H.2}$$

H.0.2 Logistic CF

The CF of a logistic distribution

$$f(x) = \frac{e^{-\frac{x-\mu}{\sigma}}}{\sigma(1 + e^{-\frac{x-\mu}{\sigma}})^2} \tag{H.3}$$

is

$$\Phi(k) = e^{ik\mu}\frac{\pi\sigma k}{\sinh(\pi\sigma k)}. \tag{H.4}$$

I Supplementary figures

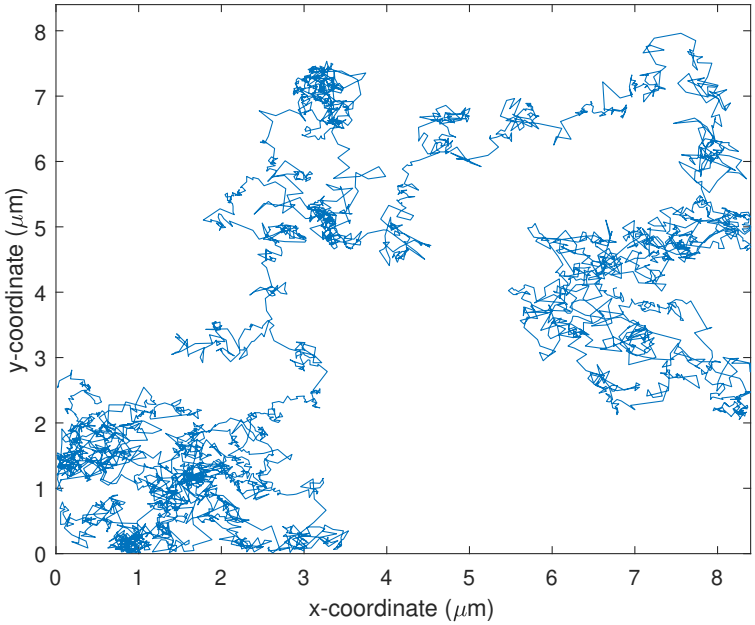
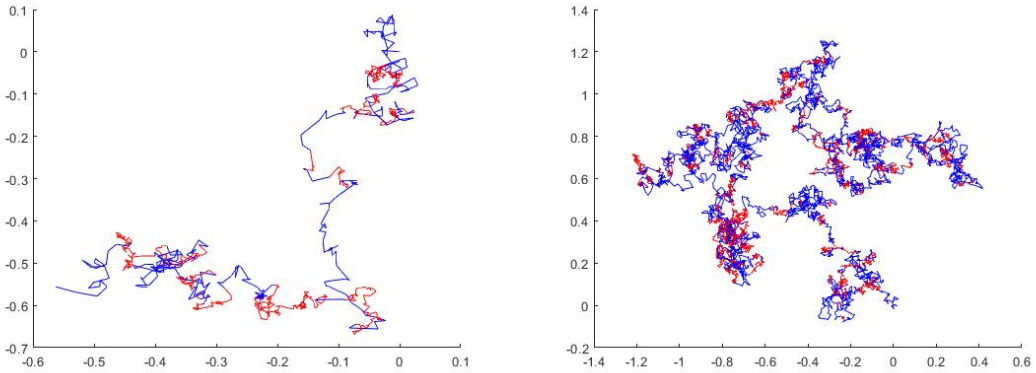


Figure 19: Example trajectory from a Gillespie simulation. The simulation parameters are summarised in Table 3. Here, 4000 steps are plotted with a sampling time $\Delta t = 5$ ms.



(a) A simulated trajectory of 10^3 steps. **(b)** A simulated trajectory of 10^4 steps.

Figure 20: Segmented state sequence trajectories. All displacements were generated from the probability density Eq. (A.40) for diffusion in two dimensions, with diffusion constant depending on the current state. The state sequence was determined using the algorithm in Fig. 1. Parameters used are $D_{\mathrm{A}} = 0.1$ $(\mu\mathrm{m})^2/\mathrm{s}$, $D_{\mathrm{C}} = 0.01$ $(\mu\mathrm{m})^2/\mathrm{s}$, $p_{\mathrm{AC}} = 0.05$, $p_{\mathrm{CA}} = 0.025$ in both cases. The sampling time was $\Delta t = 15$ ms. Blue indicates state A and red indicates state C for each step.

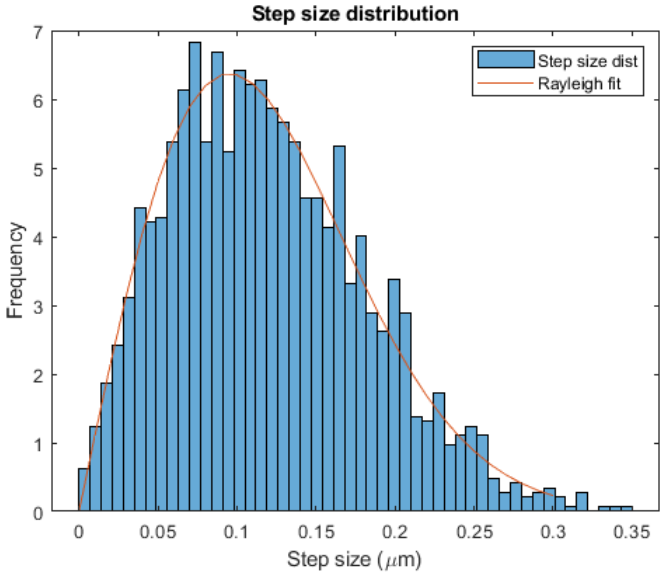
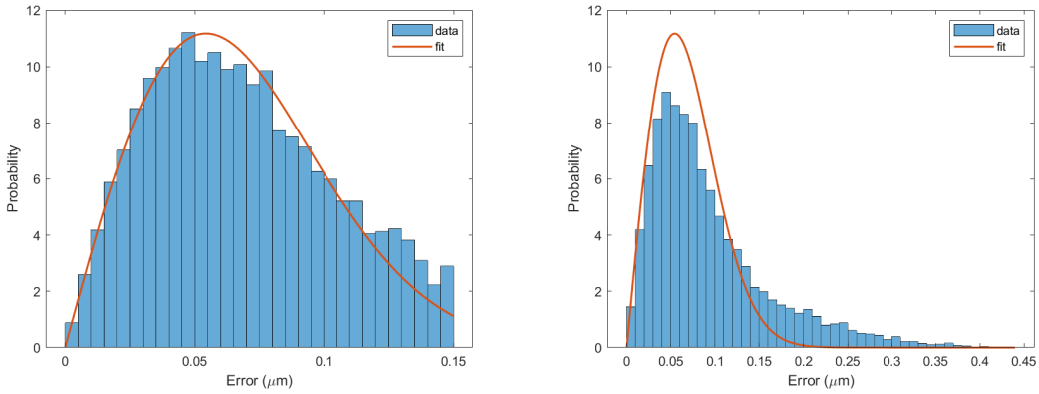


Figure 21: Step-size distribution for experimental data. A maximum-likelihood-estimation fit of Rayleigh's distribution (line) to experimental data (bars) of individual displacements. Particle tracking was done with the TrackMate plug-in in ImageJ.



(a) Rayleigh fit to localisation errors up to 0.15 μm.

(b) The same Rayleigh fit from (a), extended to all tracking errors.

Figure 22: Rayleigh fitting to localisation errors. A fitting to the distribution in Fig. 16a in the main text.