Biophysical*Journal*
Article

Biophysical Society

Resolving Fast, Confined Diffusion in Bacteria with Image Correlation Spectroscopy

David J. Rowland,[1] Hannah H. Tuson,[1] and Julie S. Biteen[1,*]
[1]Department of Chemistry, University of Michigan, Ann Arbor, Michigan

ABSTRACT By following single fluorescent molecules in a microscope, single-particle tracking (SPT) can measure diffusion and binding on the nanometer and millisecond scales. Still, although SPT can at its limits characterize the fastest biomolecules as they interact with subcellular environments, this measurement may require advanced illumination techniques such as stroboscopic illumination. Here, we address the challenge of measuring fast subcellular motion by instead analyzing single-molecule data with spatiotemporal image correlation spectroscopy (STICS) with a focus on measurements of confined motion. Our SPT and STICS analysis of simulations of the fast diffusion of confined molecules shows that image blur affects both STICS and SPT, and we find biased diffusion rate measurements for STICS analysis in the limits of fast diffusion and tight confinement due to fitting STICS correlation functions to a Gaussian approximation. However, we determine that with STICS, it is possible to correctly interpret the motion that blurs single-molecule images without advanced illumination techniques or fast cameras. In particular, we present a method to overcome the bias due to image blur by properly estimating the width of the correlation function by directly calculating the correlation function variance instead of using the typical Gaussian fitting procedure. Our simulation results are validated by applying the STICS method to experimental measurements of fast, confined motion: we measure the diffusion of cytosolic mMaple3 in living *Escherichia coli* cells at 25 frames/s under continuous illumination to illustrate the utility of STICS in an experimental parameter regime for which in-frame motion prevents SPT and tight confinement of fast diffusion precludes stroboscopic illumination. Overall, our application of STICS to freely diffusing cytosolic protein in small cells extends the utility of single-molecule experiments to the regime of fast confined diffusion without requiring advanced microscopy techniques.

INTRODUCTION

Microscopy has long been a staple technique in biological research. In particular, based on the development of techniques for selectively labeling specific cellular components, fluorescence microscopy has enormous value for elucidating intracellular biology (1). More recently, the ability to visualize a single molecule at a time has improved the localization precision below the standard diffraction limit of light (2–4). In live cells, the function of a protein in a biological process can be inferred from its rate of diffusion under different chemical or genetic conditions (5–9). Traditionally, optical measurements of subcellular diffusion have been done using fluorescence recovery after photobleaching (FRAP) (10–12), but single-molecule imaging techniques like single-particle tracking (SPT) are being increasingly used to precisely evaluate the motion of a diffusing biomolecule (13–15).

SPT is achieved by connecting a series of single-molecule positions over time. In an optimal SPT experiment, the camera integration time is fast enough that within one frame, the target diffusing molecule does not produce blur by moving far compared to the diffraction limit of light. Acquiring long trajectories improves the statistical significance, but unfortunately, there is a tradeoff due to the finite fluorescence yield of a single-molecule probe: increasing the illumination intensity to enable single-molecule detection in a shorter imaging frame time will shorten the trajectory lengths. This is a particular problem for the most common fluorescent labels in live-cell intracellular imaging, fluorescent proteins (1,16). These two conflicting requirements of fast imaging and long tracks limit the total range of measurable diffusion rates; this range limit is an important issue when heterogeneities yield a range of diffusion coefficients that are measured simultaneously. If a pulsed illumination source is available, stroboscopic illumination can be applied to decrease the amount of blur without increasing the frame rate or decreasing the trajectory lengths (17). However, confining diffusion to a small volume—for instance, within a cell or organelle—introduces a further constraint: the data acquisition rate must be faster than the time it takes for the diffusing molecule to explore the entire confinement volume.

Submitted December 15, 2015, and accepted for publication April 18, 2016.
*Correspondence: jsbiteen@umich.edu
Editor: Paul Wiseman.
http://dx.doi.org/10.1016/j.bpj.2016.04.023

Stroboscopic illumination does not increase the data acquisition rate, and so the maximum measureable diffusion coefficient for a molecule in a confined volume is still limited by the maximum camera frame rate. Additionally, in all cases, because SPT relies on a tracking algorithm to construct trajectories for single diffusing molecules, these trajectories should not overlap. Overall, SPT is best suited for characterizing a collection of sparse and homogeneously diffusing molecules in unconfined environments.

Alternatives that can overcome some limitations of the localization-based SPT analysis include spatiotemporal image correlation spectroscopy (STICS) (18–21), which has previously been used both in vitro (21) and in live cells (22), and the related methods of *k*-space image correlation spectroscopy (23,24) and superresolution optical fluctuation imaging (25). All of these methods compute the correlation function of an entire fluorescence imaging movie instead of relying on localization and tracking, and like fluorescence correlation spectroscopy, these powerful image correlation approaches can be extended to measure 2D maps of heterogeneous diffusion (18,26) and very fast diffusion (27). Particle image correlation spectroscopy measures diffusion dynamics by computing a similar correlation function from data that have already been processed with the Gaussian localization used in SPT and other methods (28). In STICS, the spatial cross-correlation at some time lag, τ, is the average of the spatial cross-correlations of all pairs of images separated by this τ and has a width (termed the image-mean-squared displacement (iMSD)) that increases with τ. The iMSD-versus-τ function can be fit to calculate the diffusive characteristics of a collection of fluorescent molecules.

In this article, we use STICS to resolve fast, confined motion in a widefield imaging microscope. If a confocal microscope is available, extremely fast diffusion ($>$100 μm^2/s) can be measured with raster image correlation spectroscopy, which uses a scanning mirror to eliminate image blur (27,29). Here, we instead focus on developing STICS to extend the capabilities of a single-molecule imaging microscope beyond the limits of SPT (maximum measurable diffusion coefficient, ~10 μm^2/s (30)) to an order-of-magnitude-faster motion. Unlike SPT, STICS does not require accurate position determinations for delocalized, fast-moving molecules. Rather, we show here that STICS analysis can explicitly account for in-frame motion blur based on a modification to the mathematical formulation. Conventionally, STICS assumes that molecular diffusion is described by a Gaussian distribution of step sizes, but both in-frame motion and tight confinement will alter the shape of this distribution. In-frame motion corrupts the distribution because the time lag between two frames becomes poorly defined when the integration time or the diffusion rate is large. Tight confinement further distorts the distribution, because this boundary condition excludes the largest step sizes. The artifacts introduced by confinement and in-frame motion were not encountered in previous applications of image correlation spectroscopy, because those experiments focused on slow motion (26,31–34), or else minimized pixel dwell time, as in raster image correlation spectroscopy (29). On the other hand, here in the regime of fast, confined motion, the two independent effects of blur and confinement manifest themselves as two independent diffusion-coefficient measurement biases.

Here, we use STICS to measure the rapid subcellular dynamics of a freely diffusing protein in the highly confined interior of a bacterial cell for the first time, to our knowledge. We describe the measurement biases that arise from confinement and fast motion by simulating unconfined diffusion and diffusion confined to bacteria-sized cylinders, both with and without in-frame blur. We develop a method to correct these diffusion-coefficient measurement biases when molecules diffusing up to 15 μm$^2 \cdot$s^{-1} are confined inside bacterial cells as small as 1 μm in length. We find that in-frame motion adds a positive bias to the diffusion-coefficient estimation and that confinement adds an independent, nonmonotonically varying bias. We show how both biases can be removed by directly computing the variances of the STICS correlation function. Finally, we apply STICS to experimental fluorescence microscopy movies of freely diffusing mMaple3 fluorescent protein in the *Escherichia coli* cytoplasm, and we find a diffusion coefficient, $D = 9.6 \pm 1.0$ μm$^2 \times$ s^{-1}, for these molecules. This value agrees well with previous fluorescence recovery after photobleaching measurements of fluorescent protein mobility in the *E. coli* cytoplasm (D = 6.1–14.1 μm$^2 \times$ s^{-1} (14)), as well as with SPT measurements with very short camera integration times (0.75–4 ms, $D = 7.3$ μm$^2 \times$ s^{-1} (30)). In this last case, a shortened camera integration time minimized the data loss due to in-frame motion (30). In this work, we instead analyze data taken with typical camera imaging rates (25–100 frames/s) by modifying the STICS protocol to explicitly account for confinement and in-frame motion. Overall, this relaxation of the experimental constraints enables direct, single-molecule-based diffusion measurements in a conventional wide-field single-molecule imaging setup (35) of samples that necessarily contain densely packed fluorophores, systems that contain multiple interacting diffusive populations, and systems that exhibit fast confined diffusion in cellular and subcellular compartments and nanomaterials (36).

MATERIALS AND METHODS

Simulations

Three-dimensional diffusion was simulated by generating step sizes from a zero-mean normal distribution with variance of 200 nm^2 in each of the three dimensions independently. This variance produces a root mean-square (RMS) displacement that is $<$10% of the waist of the smallest microscope point-spread function (σ_{PSF}) considered, enabling us to simulate smooth motion blurring. For simulations of diffusion confined to a cylinder, candidate steps that fell outside of the cylinder boundary were elastically reflected. The cell boundary is expected to act as an impermeable wall that does not interact with cytoplasmic proteins like the mMaple3 fluorescent protein used here. These 3D trajectories were projected onto the *xy* plane to simulate two-dimensional (2D) imaging.

The pixel brightness value in each subframe was calculated using a symmetric Gaussian function centered at the position of the diffusing molecule. For one-dimensional (1D) motion, the step-size distribution variance is $2Dt_{\text{frame}}$, where D is the diffusion coefficient and the camera integration time is t_{frame}. Motion with any desired D was therefore simulated by combining the appropriate number of simulation subframes; i.e., any D and t_{frame} can be calculated by taking the average of $n = 2Dt_{\text{frame}}/(200\ \text{nm}^2)$ subframes. The pixel width was set to 49 nm for all simulations for consistency with experiments. The 2D probability distribution of step sizes from one frame to the next is a normal distribution with zero mean and a variance of $4Dt_{\text{frame}}$:

$$p = N\left(0, 4Dt_{\text{frame}}\right). \tag{1}$$

The squared step sizes and correlation functions were calculated for the SPT and STICS methods, respectively, as described below. For the bias estimation simulations (see Figs. 2 and 3), incremental iterations were used to more precisely compute the STICS correlation function (see STICS analysis, below, for details). Once ~50,000 total image frames were simulated, D was estimated with STICS and we computed the bias on this value, $(\theta/\theta_0) - 1$, from the ratio of the estimated value, θ, to the true value, θ_0. For the error estimation simulations (see Fig. 4), both STICS and SPT were performed on identical simulated raw data to which zero-mean white noise had been added with the Matlab built-in function *randn* to achieve a signal/noise ratio (SNR) of 6 for immobile molecules. The effective SNR, however, is reduced by motion blur and will be <6. The simulated data set used in this study consisted of 1200 frames of contiguous motion.

SPT analysis

Candidate molecular locations were found in simulated single-molecule movies by following a band-pass filter in the Fourier domain with a watershed algorithm. These positions were refined to the local center of mass within a 15 pixel × 15 pixel window. The refined locations were then used as the initial guess for a least-squares fit of the raw image of the molecule by a seven-parameter generalized bivariate Gaussian function using Matlab's built-in function *lsqcurvefit*. The center of this fit was taken to be the position of the diffusing molecule. Because only a single molecule was present in our simulations at any time, the time series of localizations was used as the single-particle trajectory, with no need for a more advanced tracking algorithm. Infrequently, the analysis of a single-molecule image incorrectly indicated more than one molecule, in which case all putative molecules from that image were ignored. The MSDs for the first 15 possible time lags, τ, were then computed and these results were fit to a model for 1D confined diffusion inside an infinite square well (13):

$$\text{MSD}(\tau) = \frac{L^2}{6}\left(1 - \frac{96}{\pi^4}\sum_{\text{n odd}}\frac{1}{n^2}\exp\left[-\left(\frac{n\pi}{L^2}\right)^2 D\tau\right]\right) + B. \tag{2A}$$

Here, L is the confinement length and B is a constant offset equal to the measurement variance of the position of an immobile molecule. Only the MSD curve computed in the direction of the cylinder long axis was considered, because with the diffusion coefficients considered here, the diffusing molecule may explore the entire short axis of the cell in a single frame. 1D motion was recovered from 2D position data by projection onto the long axis of the cell.

STICS analysis

For both simulated and experimental data, the full time-space correlation functions were computed using the Fourier transform method for convolutions, as described previously (17,33). The width of the correlation function increases with time lag, and Gaussian fitting can estimate these widths. The time lag series of widths is called the iMSD, which for our model of square-confined diffusion (13) has the functional form

$$\text{iMSD}(\tau) = \frac{L^2}{6}\left(1 - \frac{96}{\pi^4}\sum_{\text{n odd}}\frac{1}{n^2}\exp\left[-\left(\frac{n\pi}{L^2}\right)^2 D\tau\right]\right) + C. \tag{2B}$$

Here, C is a constant offset proportional to the diameter of the image of the diffusing molecule. The sum was truncated after the change in the function value dropped below 10^{-10}, a condition which was always satisfied with fewer than 10 terms. Only the iMSD curve computed from the cell long axis was considered, because for the fast diffusion coefficients considered here, the diffusing molecule explores the entire short axis of the cell in a single image frame. For the experimental bacterial cell data analysis, one iMSD curve was calculated for each of 87 movies, which had ~4000 imaging frames each. The function iMSD(τ) was fit to Eq. 2B for the first 15 values of τ (with the Matlab function *lsqcurvefit*) to estimate the diffusion coefficient, D.

Each frame of the simulated movies was symmetrically padded to double the original width with a padding value of zero because the data were generated with zero intensity offset. For the experimental data, the phase contrast image of the bacterial cell was used to automatically compute, with a valley filter, a selection mask that describes the position of each bacterium. Pixels that fell outside of this mask were replaced with the mean intensity value inside the selection mask. Because the absolute offset of the real bacterial cell image is ill-defined or weakly measureable, the experimental bacterial cell data were padded with the average intensity value inside the cell.

Bacterial strains and growth conditions

DH5α *E. coli* cells containing a pBAD plasmid encoding the fluorescent protein mMaple3 under the arabinose promoter (37) were struck out from a freezer stock on a 1.5% Luria Bertani (LB)/agar plate containing 50 μg × mL^{-1} ampicillin and incubated overnight at 37°C. A single colony from the plate was used to inoculate 2 mL LB containing 50 μg × mL^{-1} ampicillin. This culture was incubated overnight (~16 h) at 37°C with shaking. Following incubation, the culture was diluted 1:50 into M9 minimal medium containing 0.4% glycerol as the carbon source. The culture was incubated at 37°C with shaking to an optical density at 595 nm of ~0.35, at which time mMaple3 expression was induced by adding arabinose to a final concentration of 0.01%. After induction, the culture was incubated at 37°C with shaking for an additional 2 h. Cells were pipetted onto a 2% agarose/M9/arabinose pad and inverted onto a plasma-etched coverslip for imaging, as previously described (38).

Alternatively, longer cells were grown by using cephalexin, a β-lactam antibiotic that blocks cell division but allows cell growth (39). Here, the overnight culture was diluted 1:100 into LB and incubated at 37°C with shaking to an optical density at 595 nm of ~0.5. The culture was then centrifuged for 5 min at 5000 × g, the supernatant was decanted, and the cell pellet was resuspended in an equal volume of M9 medium containing 0.4% glycerol and 0.01% arabinose. Cephalexin was added to a final concentration of 60 μg × mL^{-1}. The culture was then incubated at 37°C with shaking for an additional 30 min. Cells were pipetted onto a 2% agarose/M9/arabinose pad (which also contained 60 μg × mL^{-1} cephalexin) and inverted onto a plasma-etched coverslip for imaging.

Imaging conditions

Bacterial samples were imaged at room temperature using wide-field epifluorescence microscopy in an Olympus IX71 inverted microscope with a 100×, 1.40 NA oil immersion objective (in Immersol 518F immersion oil, Carl Zeiss, Oberkochen, Germany) and appropriate excitation, emission, and dichroic filters (LL01-561, BLP01-561, and Di01-R561, respectively, Semrock, Rochester, NY). After a 3× beam expander, a Photometrics (Tucson, AZ) Evolve EMCCD camera with >90% quantum efficiency captured the images at 25–100 frames/s. Each camera pixel corresponds to a 49 nm × 49 nm area of the sample. Fluorescence of mMaple3 in the cells was photoswitched from green to red photon emission using a 406 nm laser (Cube 406-100, Coherent, Santa Clara, CA), co-aligned with the 561 nm fluorescence excitation laser (Sapphire 560-50, Coherent). The samples were illuminated with low laser power densities (0.07 μW × μm^{-2} and 0.01 mW × μm^{-2} for the 406 nm illumination and the 561 nm illumination, respectively). Both laser beams were circularly polarized with a quarter waveplate (AO15Z 1/4 556 and AO15Z 1/4 408, respectively, Tower Optical, Boynton Beach, FL). During imaging, the cells were given a 1- to 10-ms dose of 406 nm light every ~30 s. Movie acquisitions lasted 2–5 min each.

Due to the small (~1 μm) cell thickness, a low background noise is maintained even when single-molecule imaging in bacteria cells is done with a wide-field microscopy configuration (40). Thus, this work describes the analysis of wide-field microscopy, though these methods could be readily applied to total internal fluorescence microscopy.

Theory

Characteristic motion, M$_C$: a unit-free parameter describing the amount of in-frame motion and the extent of confinement

Because of the high dimensionality of the simulation parameter space, which includes, e.g., camera integration time, pixel size, and magnification, it is desirable to have an invariant description of the degree of in-frame motion. To this end, we renormalize the diffusion rate, D, to produce a unit-free characteristic motion parameter, M_C. This M_C simplifies the presentation of the simulation results and can highlight consistencies among parameter sets. M_C is proportional to the squared distance, $D \times t_\mathrm{frame}$, traveled in each captured image and inversely proportional to the variance of the point-spread function of the diffusing molecule (σ^2_PSF). For consistency among dimensionalities, the factor 2/d is applied, where $d = 2$ for 2D image data:

$$M_{C,\mathrm{unconfined}} = \frac{2Dt_\mathrm{frame}}{d\sigma^2_\mathrm{PSF}}. \tag{3A}$$

For confined diffusion, doubling the confinement length, L, is equivalent to halving the RMS in-frame displacement value. In the case of confined diffusion, the unconfined characteristic motion, $M_\mathrm{C,unconfined}$, is therefore modified by a scaling factor that describes the degree of confinement, the unit-free factor L_0^2/L^2:

$$M_{C,\mathrm{confined}} = \frac{2Dt_\mathrm{frame}}{d\sigma^2_\mathrm{PSF}}\frac{L_0^2}{L^2}, \tag{3B}$$

where L_0 is a reference confinement length chosen here to be the average cell length, 3 μm.

In Figs. 2, 3, and 6, we rescale D based on the simulation parameters (Table S1 in the Supporting Material) according to these definitions of M_C. The overlap seen in these figures demonstrates the validity of Eqs. 3A and 3B.

The effect of in-frame motion on STICS

In STICS, the correlation function, G, is the convolution of the step-size probability distribution, p, with W_PSF, the approximately Gaussian microscope point-spread function (19):

$$G(\xi,\chi,\tau) = \frac{\gamma}{N}p(\xi,\chi,\tau)\otimes W_\mathrm{PSF}(\xi,\chi). \tag{4}$$

Here, ξ and χ are displacements in x and y, respectively, for each time lag τ. N is the average number of molecules present in each frame and γ is a geometric factor that accounts for an underrepresentation of intensity fluctuations due to the illumination beam shape. The 2D step-size distribution, p, of unconfined Brownian diffusion is traditionally assumed to be normally distributed:

$$\mathrm{Var}(p) = 2dD\tau, \tag{5}$$

where d is the dimensionality and D is the diffusion coefficient. When Eq. 5 applies, G in Eq. 4 will be a Gaussian distribution, because it is the convolution of two Gaussians. However, when there is significant in-frame motion, the diffusing molecule will have moved a nonnegligible distance during the imaging integration time, t_frame. In this case of significant in-frame motion, the step-size distribution, p, is not only a function of the average time lag, τ_0; p also depends on a range of effective time lags, τ_eff, in the range of $\tau_0 - t_\mathrm{frame} < \tau_\mathrm{eff} < \tau_0 + t_\mathrm{frame}$. This mixture of step sizes from nonidentical time lags causes p to become the sum of nonidentical normal distributions, and importantly, this p is not a normal distribution. When such in-frame motion is present, the correlation function, G_blur, is the convolution of the point-spread function with a weighted average of Gaussians:

$$G_\mathrm{blur} = \frac{\gamma}{N}\left[\left(\int_{\tau_0-t_\mathrm{frame}}^{\tau_0+t_\mathrm{frame}} g\left(\tau_\mathrm{eff}\right)\,d\tau_\mathrm{eff}\right)^{-1} \times \int_{\tau_0-t_\mathrm{frame}}^{\tau_0+t_\mathrm{frame}} g\left(\tau_\mathrm{eff}\right)N\left(0,4D\tau_\mathrm{eff}\right)\,d\tau_\mathrm{eff}\right]\otimes W_\mathrm{PSF}(\xi,\chi). \tag{6}$$

Here, $N(0,4D\tau_\mathrm{eff})$ is a zero-mean normal distribution with variance $4D\tau_\mathrm{eff}$ and weighting factor $g(\tau_\mathrm{eff})$. The point-spread function, W_PSF, is an Airy Disk, which we approximate here by a Gaussian function (41).

The effect of confinement on STICS

To simulate confinement, we imposed reflective boundary conditions on the step-size distribution, p. In the limit of fast diffusion and small confinement, p approaches a uniform distribution. In general, if zero-flux boundary conditions are imposed on ξ and χ in the distribution, $p(\xi,\chi,\tau)$, which is a symmetric function centered on zero, then the correlation function, G, is modified:

$$G_\mathrm{conf} = \frac{\gamma}{N}\left[2\sum_{n,\mathrm{int.}} p(\xi+nL,\chi+nL,\tau) - p(\xi,\chi,\tau)\right]\otimes W(\xi,\chi), \quad -\frac{L}{2}\leq\xi,\chi\leq\frac{L}{2}. \tag{7}$$

Overall, the effect of Eqs. 6 and 7 is a deviation of the Gaussian function in Eq. 4 to a degree that depends directly on the amount of in-frame motion or confinement.

Estimation of the diffusion coefficient by direct calculation of the variance

According to Eq. 5, D is proportional to the variance of the step-size distribution, p. If p describes ideal free diffusion, then p is Gaussian-shaped. The correlation function, G, computed by STICS for this type of data would then also be Gaussian. However, G is not generally Gaussian due to in-frame motion and confinement, as described above. In its traditional implementation, STICS estimates the correlation-function variance, iMSD, as being the variance parameter of a least-squares fit of G to a Gaussian function. If G is not Gaussian, then fitting G to a Gaussian will not accurately measure the variance. In this case, the inaccuracy can be removed by directly computing the variance according to its definition:

$$\mathrm{iMSD}_x = \mathrm{Var}(G_\xi) = \sum_i \xi_i^2 G_\xi \tag{8A}$$

$$\mathrm{iMSD}_y = \mathrm{Var}(G_\chi) = \sum_i \chi_i^2 G_\chi. \tag{8B}$$

Here, we consider in-plane diffusion in each of the two dimensions, x and y, separately. The marginal distributions G_ξ and G_χ for displacements in x and y, respectively, are computed from the 2D correlation function, G, by discrete summation:

$$G_\xi = \sum_\xi G(\xi, \chi, \tau) \tag{9A}$$

$$G_\chi = \sum_\chi G(\xi, \chi, \tau). \tag{9B}$$

This separation allows diffusion along the long and short axes of bacterial cells to be treated independently. We fixed the orientation of the cylindrical boundary in the simulations to lie parallel to the x axis and directly calculated the variance for the simulated data. This direct variance calculation method was not used to estimate the iMSD of the experimental data, because the measurement precision was too low to be useful for this analysis, especially after the interpolation involved in image rotation.

It should be noted that this method for finding the variance requires the distribution, G, to be normalized at every value of τ. In general, however, the integral of G with respect to ξ and χ is not unity for each time lag. In principle, this integral can be normalized by dividing it by the discrete integral of G or by dividing it by the known prefactor γ/N in Eq. 4. In our simulations, we ensured a priori that G was normalized at every τ by fixing $\gamma/N = 1$.

RESULTS AND DISCUSSION

Effect of in-frame motion on SPT

We simulated diffusion with 10 different values of D in 3-μm-long cylinders with reflective boundary conditions. These movies were processed by SPT, and the diffusion coefficient, D, for each trajectory was estimated from the MSDs. Each measured D was scaled to the corresponding unit-free characteristic motion, M_C, according to Eq. 3B. Because not all single-molecule images could be fit in our regime of finite SNR, trajectories were truncated. We computed for each M_C the percent recovered displacements, $n_1/(n_T - 1)$, where n_1 is the number of measured displacements that have a time lag of one frame and n_T is the total number of frames in the simulated movie. The percent recovered displacements decreases as M_C increases, and the number of recovered displacements approaches zero for large M_C (Fig. 1).

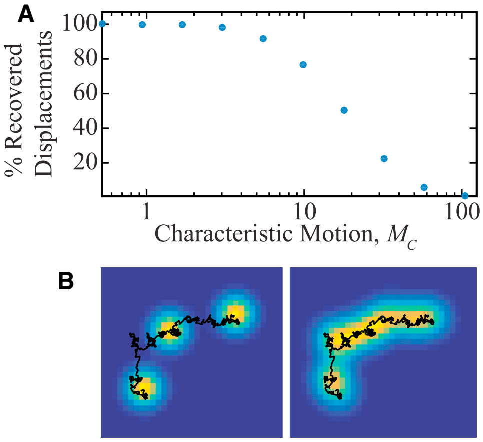


FIGURE 1 Motion blur degrades the single-particle tracking (SPT) signal as characteristic motion, M_C, increases. Diffusion confined in a cylinder was simulated and analyzed with SPT. M_C is defined according to Eq. 3B. (*A*) The percent of recovered data decreases with increasing M_C. The pixel width is 49 nm, confinement length is $L = 3$ μm, integration time is $t_{\mathrm{frame}} = 50$ ms, and $\sigma_{\mathrm{PSF}} = 98$ nm. (*B*) Representative simulated images of a diffusing molecule with three of the subframes (*left*) or all of the subframes (*right*) included in the image. The actual trajectory (*black curve*) is the same in both panels. To see this figure in color, go online.

In general, M_C is proportional to D and t_{frame}, and increasing either of these parameters increases the degree of in-frame motion of the diffusing molecule. Accordingly, Fig. 1 *A* indicates that increasing in-frame motion leads to decreased data recovery. Fig. 1 *B* compares simulations of the same trajectory (*black line*) with different amounts of in-frame motion. On the left side, the trajectory is sampled instantaneously at three distinct time points, producing three ideal point-spread functions (*punctate yellow spots*); the molecular positions at those time points can be determined from a Gaussian fitting algorithm. On the other hand, the right side shows the case when the trajectory is sampled continuously. Such in-frame motion can blur and distort the molecule image. Thus, when in-frame motion is dominant, this distortion prevents a Gaussian fitting algorithm from identifying molecular positions. Because SPT creates a trajectory from a series of single-molecule fits, such a failed fit decreases the percent recovered displacements. SPT, then, is a method that tolerates only subtle in-frame motion. In Fig. 1 *A*, <50% of the one-frame displacements are recoverable when M_C is >20, even though 20 is a reasonable value for subcellular diffusion in a typical single-molecule microscope (40). For instance, $M_C = 20$ could correspond to a molecule diffusing with a rate of 2.88 μm^2/s measured with a microscope that has a point-spread function standard deviation of 98 nm, a camera integration time of 50 ms, and a pixel width of 49 nm in the object plane. Experimentally, in-frame motion can be minimized for even extremely fast diffusers with stroboscopic illumination (e.g., pulsed sample illumination with pulse widths that are shorter than t_{frame}; Fig. 1 *B*, *left panel*) (17) or very high imaging frame rates (30).

Effect of in-frame motion on STICS

STICS has the potential to estimate the diffusion coefficient in the regime of large M_C (i.e., significant image blurring) that precludes SPT analysis. However, a description of the accuracy and precision of this method are required to validate its use in this regime. To investigate the accuracy of STICS in different diffusion regimes, we first simulated unconfined, single-molecule diffusion with 10 diffusion coefficients between 0.1 $\mu m^2 \times s^{-1}$ and 20 $\mu m^2 \times s^{-1}$ (Fig. 2). Each diffusion coefficient was considered for $t_{frame} = 10$ ms and 50 ms and for point-spread function standard deviations of $\sigma_{PSF} = 49$ nm, 98 nm, and 147 nm. No noise was included in this set of simulations. Per Eq. 3A, different combinations of these parameters can lead to the same degree of in-frame motion, so each measured diffusion coefficient, D, was converted to the corresponding value of the unit-free parameter M_C. Simulated movies were analyzed by STICS, and the diffusion-coefficient measurement bias, $(D_{measured} - D_0)/D_0$, is plotted as a function of M_C in Fig. 2 *A*. According to Fig. 2 *A*, STICS very accurately estimates D at low M_C, but overestimates the diffusion coefficient as M_C increases. This positive measurement bias increases monotonically with increasing in-frame motion, with a bias of ~10% when $M_C = 100$. The measurement bias for each set of parameters (t_{frame}, σ_{PSF}) was assigned a unique color in Fig. 2 *A* (Table S1). The different color sets overlap and show the same trend, demonstrating that the characteristic motion, M_C in Eq. 3A, is indeed the right variable for considering diffusion at different rates under different experimental conditions.

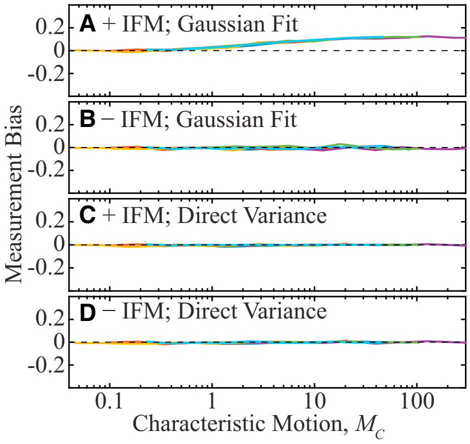


FIGURE 2 Diffusion coefficient measurement bias estimated from simulations of *unconfined* diffusion of a fluorescent molecule with no noise. Colors indicate the set of parameters (t_{frame}, σ_{PSF}) (Table S1), and M_C is defined according to Eq. 3A. (*A*) STICS analysis by Gaussian fitting of simulations that contain in-frame motion (*IFM*). Significant bias is measured for $M_C > 1$. (*B*) STICS analysis by Gaussian fitting of simulated movies that exclude in-frame motion. (*C* and *D*) The results of analyzing the same data as in (*A*) and (*B*), but determining the step-size distribution variance in the STICS protocol by directly calculating the variance rather than by fitting the correlations to Gaussian functions. Both the removal of in-frame motion in (*B*) and the direct calculation of variance in (*C*) and (*D*) eliminate the measurement bias. To see this figure in color, go online.

We generated the images of a diffusing fluorescent molecule with in-frame motion analyzed in Fig. 2 *A* from a series of images of a stationary molecule that diffuses between subframes (Fig. 1 *B*, *right panel*). We hypothesized that the measurement bias in Fig. 2 *A* can be attributed to in-frame motion because this bias disappears when M_C is small. To examine the role of in-frame motion, we excluded this motion from the simulation by excluding all subframes but the first in each image (Fig. 1 *B*, *left panel*). We repeated STICS analysis on this stroboscopic data set, which is free of in-frame motion (Fig. 2 *B*), and found that the measurement bias is removed when no in-frame motion is allowed. Thus, the bias in Fig. 2 *A* is attributed to in-frame motion.

For the conventional STICS analysis in Fig. 2, *A* and *B*, the correlation function variance is estimated as the variance parameter of a fit to a Gaussian function (18). Although curve fitting to the correlation function is a powerful method of increasing the diffusion coefficient measurement precision, as described in the Theory section, in the case of in-frame motion, the step sizes will not be normally distributed. Therefore, we hypothesized that the diffusion coefficient measurement bias in Fig. 2 *A* comes from fitting a non-Gaussian correlation function to a Gaussian fitting function. We therefore calculated the correlation function variances directly instead of by a fit to a Gaussian (Eq. 8). The measurement biases from this method are shown in Fig. 2, *C* and *D*, for simulated data with and without in-frame motion, respectively. This direct variance calculation allows for bias-free measurements of the diffusion coefficient even when in-frame motion is included in the simulated movies (compare Fig. 2, *A* and *C*). The bias due to in-frame motion in Fig. 2 *A* is therefore the result of fitting STICS correlation functions to a fit function that incorrectly assumes an absence of in-frame motion, and the discrepancy in shape between the actual correlation function and this assumed Gaussian function increases with increasing in-frame motion.

Effect of confinement on STICS

In addition to the artifacts that arise from fitting fast diffusion to the wrong function in the STICS method (Fig. 2 *A*), we hypothesized that confinement also gives rise to a diffusion-coefficient measurement bias in the case of fast-diffusing molecules (high M_C). We considered the diffusion of a fluorescent molecule in the cytoplasm of a bacterial cell by simulating diffusion of a molecule in cylinders of diameter 1 μm and lengths $L = 2$, 3, and 4 μm with 10 diffusion coefficients between 0.1 $\mu m^2 \times s^{-1}$ and 20 $\mu m^2 \times s^{-1}$. We considered that $t_{frame} = 10$ and 50 ms, and σ_{PSF} was 98 nm. No noise was included in this set of simulations. In Fig. 3, each set of parameters is scaled to the unit-free parameter M_C as described in Eq. 3B.

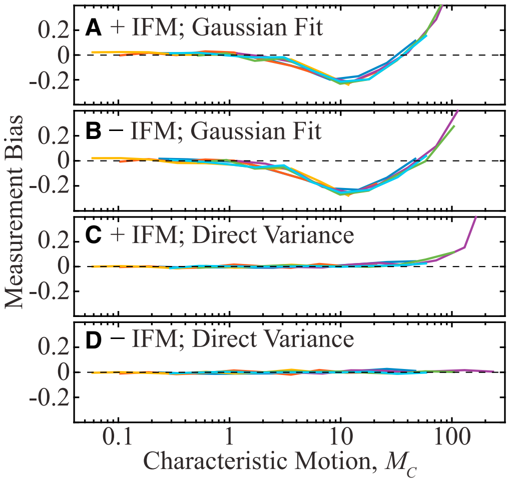


FIGURE 3 Diffusion coefficient measurement bias estimated from simulations without noise of a fluorescent molecule confined to a cylinder. Colors indicate the set of parameters (t_{frame}, σ_{PSF}) (Table S1), and M_C is defined according to Eq. 3B. (*A*) STICS analysis by Gaussian fitting of simulations that contain in-frame motion (*IFM*). Significant, nonmonotonically varying bias occurs for fast characteristic motion, $M_C > 1$. (*B*) STICS analysis by Gaussian fitting of simulated movies that exclude in-frame motion. The nonmonotonically varying bias from (*A*) is still present at all values of M_C considered, although the bias due to in-frame motion is removed. (*C* and *D*) The results of analyzing the same data as in (*A*) and (*B*), but determining the step size distribution variance in the STICS protocol by directly calculating the variance rather than by fitting the correlations to Gaussian functions. The combination of direct variance calculation and removal of in-frame motion in (*D*) removes the bias completely. To see this figure in color, go online.

Conventional STICS analysis of these simulations of confined motion (Fig. 3 *A*) revealed a large, nonmonotonically varying bias when $M_C > 1$, with biases as large as 50% for $M_C = 100$. This effect persists even when in-frame motion is eliminated from the simulations (Fig. 3 *B*) by excluding all subframes but the first in each image, as described in Fig. 2 *B*. Comparing Fig. 3, *A* and *B*, (see Fig. S1 *A*) indicates that in-frame motion does contribute slightly to the bias in Fig. 3 *A*, but the persistence of a significant bias in Fig. 3 *B* indicates that the effect of in-frame motion is minimal relative to the effect of confinement on the measurement bias.

We hypothesized that this confinement artifact arose from using an incorrect fit function to determine the correlation-function variance. To circumvent this fit function, the correlation-function variances were calculated directly, as for Fig. 2, *C* and *D*. Again, the direct variance calculation allows a bias-free measurement of *D*: directly calculating the variance in this manner removed nearly all of the measurement bias for simulated data even with in-frame motion (Fig. 3 *C*). The fact that direct variance calculation can remove the large biases in estimation of *D* indicates that, as was true for in-frame motion (Fig. 2 *A*), confinement leads to a large discrepancy between the assumed normal step size distribution (Eq. 1) and the actual observed distribution.

As confinement (and therefore M_C) increases, a Gaussian fitting function no longer correctly describes the STICS correlation function, and the resulting bias increases with increasing M_C (Fig. S1 *B*). There remains in Fig. 3 *C* a large positive bias at large M_C that is caused by in-frame motion and is removed upon excluding in-frame motion (Fig. 3 *D*).

Comparing the precision of STICS and SPT

Given the protocol established above to remove measurement *bias* artifacts from STICS analysis through direct calculation of the variance, we next considered the measurement *errors* incurred in estimating single-molecule diffusion coefficients by either STICS or SPT in the presence of realistic noise and experimental parameters. We examined the ability of STICS to precisely determine *D* in simulated 1200-frame movies of a fluorescent molecule diffusing inside a cylinder of diameter 1 μm and $L = 3$ μm, with SNR $= 6$, $\sigma_{\text{PSF}} = 98$ nm, $t_{\text{frame}} = 50$ ms, and 10 diffusion coefficients between 0.1 $\mu\text{m}^2 \times \text{s}^{-1}$ and 20 $\mu\text{m}^2 \times \text{s}^{-1}$. In Fig. 4, the percent error in the determination of the diffusion coefficient is plotted as a function of M_C (as defined in Eq. 3B). STICS and SPT analysis are performed on two identical data sets: movies simulated with and without in-frame motion. For minimal in-frame motion and minimal confinement (small M_C), the precision of SPT is higher than that of STICS. Yet, when $M_C > 30$ (Fig. 4, *arrow*), in-frame motion causes SPT (*yellow curve*) to be significantly less precise than STICS (*blue curve*). This is consistent with the data loss during tracking (Fig. 1 *A*). However, in the absence of

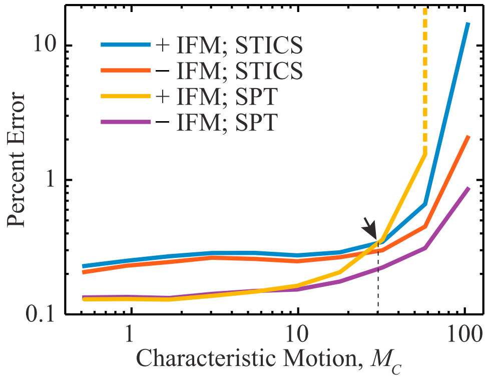


FIGURE 4 Simulated movies (see text for parameters) of confined diffusion with and without in-frame motion were analyzed with the SPT and STICS methods to compare the precision of the two methods. The diffusion coefficient determination by STICS was done by direct calculation of the variance. SPT has higher precision at slow speeds, but becomes less precise for fast characteristic motion. The inclusion of in-frame motion causes the precision of SPT to become worse than that of STICS at $M_C \sim 30$ (*arrow*), which, for this parameter set, corresponds to a diffusion coefficient of 5.76 $\mu\text{m}^2 \times \text{s}^{-1}$. The tenth data point of the yellow curve (SPT with in-frame motion) was infinite due to the data loss described in Fig. 1 *A*; this singularity is indicated by a dashed vertical line beginning at the ninth data point. To see this figure in color, go online.

in-frame motion, SPT outperforms STICS under all experimental conditions considered in this work. The error in determining D for both methods increases with increasing M_C, because the effective SNR of each simulated image decreases with increasing M_C, as is the case when the number of photons emitted by a molecule is kept constant, but these are spread over more pixels due to image blur.

By directly considering the correlation-function variance rather than assuming a Gaussian functional form for this distribution, the diffusion of even fast-moving, highly confined molecules can be estimated with STICS. Indeed, because of in-frame motion, when $M_C > 50$, one can no longer track diffusing molecules at all with SPT (Fig. 4, *yellow curve*), whereas STICS can still determine the diffusion coefficient with <1% error. Overall, Fig. 4 can indicate the appropriate analytical method for a given single-molecule diffusion measurement based on the experimental parameters.

Cytosolic mMaple3 diffusion in *E. coli*

Although single-molecule imaging has successfully answered a host of questions in cell biology (40,42–44), one current challenge is a limited ability to characterize the dynamics of the fastest molecules—especially when confinement precludes stroboscopic illumination—for instance, freely diffusing proteins in the bacterial cytoplasm (5). We imaged *E. coli* cells expressing the photoswitchable fluorescent protein mMaple3, which is switched from a green state to a red state upon activation with 406-nm light (37). Before activation, the cells were nonfluorescent (Fig. S3 *B*). We exposed the cells to 406-nm laser pulses to photoswitch one mMaple3 molecule at a time (Fig. S3 *C*), and we sequentially imaged dozens of mMaple3 molecules in each of 87 different cells. The cells were imaged in the red channel (imaging wavelength, 561 nm) with $t_{\text{frame}} = 40$ ms, $\sigma_{\text{PSF}} = 98$ nm, and pixel width = 49 nm (Fig. S3; Movie S1). We measured the cell lengths, which varied from 1 to 10 μm. Single-molecule photoswitching was evident as the average fluorescence intensity of the cell increased after each photoswitching pulse and decreased a short time later due to photobleaching, but the large degree of in-frame motion due to the fast diffusion prevented single molecules from being visualized as punctate fluorescent spots and localized with Gaussian fitting in the majority of the images. This is consistent with the low data recovery rate for simulations of molecules with high M_C in Fig. 1 *A*.The diffusion of free mMaple3 under these conditions could therefore not be analyzed by SPT. Although the method of direct variance determination removes the systematic error (compare Figs. 2 *C* and 3 *C* to Figs. 2 *A* and 3 *A*) and explains the source of the bias, this method has extremely low precision (Fig. S2, *light blue* and *green curves*). This approach is therefore not used to analyze the experimental results that follow.

Fig. 5 illustrates our implementation of Gaussian-fitting STICS for one representative cell ($L = 3.1$ μm). A phase-contrast image of the cell (Fig. 5 *A*) provided the boundaries for a cell mask (Fig. 5 *B*), and a single iMSD curve was compiled for each cell's axial dimension based on Gaussian fits to the correlation function within this mask (Fig. 5 *C*). Each iMSD was fit to the square-confined diffusion model in Eq. 2B to estimate the coefficient of diffusion of mMaple3 in that cell (Fig. 5 *D*). Gaussian fitting was used to calculate the iMSD with increased precision in our experimental regime of relatively noisy data. Also, because transverse and longitudinal diffusion must be considered independently, here, due to the different confinement lengths in the two directions, the correlation-function variances cannot be calculated directly without interpolating the raw data onto a rotated pixel array, which would impose additional errors.

We analyzed 87 unique bacterial cells and used the small molecule cephalexin to generate cells with $L = 1$–10 μm. The effect of cephalexin on the diffusion of mMaple3 in the cytoplasm of *E. coli* is expected to be minimal (12). The simulations described in Fig. 3 *A* indicate the biases that are expected, and the expected bias from Fig. 3 *A* can be used to convert biased experimental diffusion

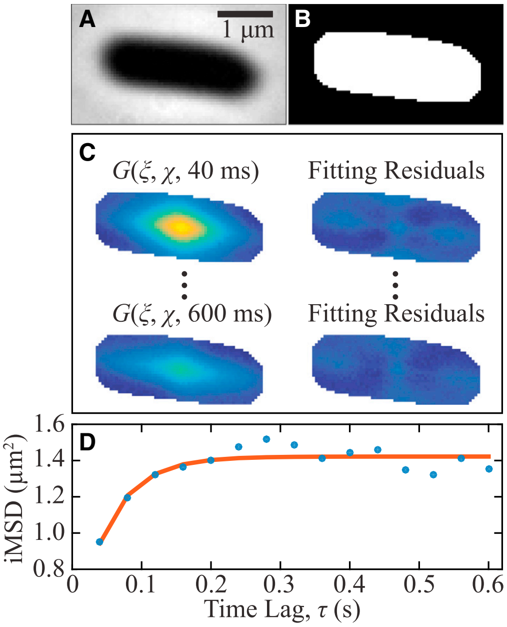


FIGURE 5 Analysis procedure for estimating by STICS the diffusion coefficient of free cytosolic mMaple3 inside a single *E. coli* cell. (*A*) A phase contrast image of the *E. coli* cell provides (*B*) a mask of the cell location and orientation. (*C*) The STICS correlation function is computed and then fit to a Gaussian function, *G*. The phase mask in (*B*) includes only the correlation amplitudes that correspond to displacements inside the cell. (*D*) The long-axis variances of the Gaussian fits to the correlation functions (*iMSD*) were plotted as a function of time lag, τ (*dots*), and this iMSD curve was fit to a model for square-confined diffusion (Eq. 2B; *red curve*) to obtain a single diffusion coefficient measurement for each cell. To see this figure in color, go online.

measurement made with Gaussian-fitting STICS to the corresponding unbiased value (Fig. 6 *A*). For instance, in the example of the representative cell in Fig. 5, Gaussian-fitting STICS measured $D = 5.20\ \mu\text{m}^2 \times \text{s}^{-1}$ in a cell with $L = 2.41\ \mu\text{m}$. This measurement was taken with $\sigma_{\text{PSF}} = 98$ nm, $t_{\text{frame}} = 40$ ms, and pixel width = 49 nm, so Eq. 3B gives $M_{\text{C}} = 33.6$, for which value Fig. 3 *A* predicts a bias of -0.01 (Fig. 6 *A*, *arrow*). Interestingly, for the representative cell in Fig. 5 ($M_{\text{C}} = 33.6$), the measured *D* (5.20 $\mu\text{m}^2 \times \text{s}^{-1}$) is converted to an unbiased *D* of 5.15 $\mu\text{m}^2 \times \text{s}^{-1}$; i.e., Gaussian-fitting STICS produces a nearly unbiased estimate of *D* for the amount of in-frame motion and confinement in this regime.

Based on the interpolated curve in Fig. 6 *A*, all *D* measurements were converted to unbiased *D*, and Fig. 6 *B* shows a histogram of the average diffusion-coefficient estimates for single mMaple3 molecules in each of the 87 different *E. coli* cells. The unbiased mean diffusion coefficient of 9.6 ± 1.0 $\mu\text{m}^2 \times \text{s}^{-1}$ agrees with the range of free protein diffusion inside the cytoplasm of *E. coli* reported elsewhere (30,45), indicating that Gaussian-fitting STICS is an appropriate analysis method for obtaining the average *D* of a cytoplasmic fluorescent protein in *E. coli*. We found no correlation between the measured *D* and *L*, indicating that the degree of confinement did not strongly affect our measurements (Fig. S4). Overall, these experiments demonstrate that the diffusion of fluorescent molecules in small volumes with a significant amount of in-frame motion that precludes SPT can be characterized by the STICS method.

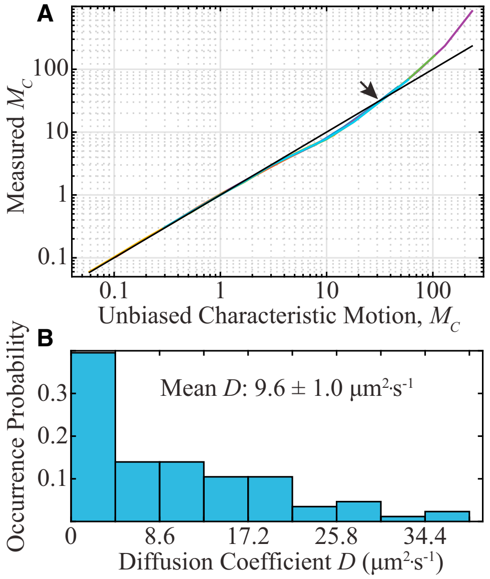


FIGURE 6 Analysis of the diffusion coefficients of mMaple3 in 87 *E. coli* cells measured with STICS. (*A*) The measured M_{C} may be converted to unbiased M_{C} based on the simulations in Fig. 3 *A*, which are used here to create a lookup table. The overlapping colored lines represent the same experimental parameter sets as in Fig. 3 *A* (Table S1). The black line indicates 1:1 correspondence (no bias). (*B*) Histogram of the unbiased average mMaple3 diffusion coefficient measured in each of 87 different *E. coli* cells as estimated by the STICS method. Between 5 and 15 mMaple3 fluorescent proteins are photoswitched one at a time in each cell, and the biased measurements are corrected by cubic interpolation of the curve in (*A*). To see this figure in color, go online.

CONCLUSIONS

In this study, we have extended STICS to the regime of rapidly moving molecules in highly confined environments by considering the motion of freely diffusing fluorescent proteins inside living *E. coli* bacterial cells. In this regime, such fast, confined motion can still be characterized with negligible bias, although we have also identified regimes where fast, confined motion results in a significant bias in diffusion-coefficient measurement by STICS. Simulations of this bias indicate that it stems from a STICS correlation function that is not well approximated by a Gaussian function (Figs. 2 *A* and 3 *A*); this bias increases with increasing diffusion coefficient, *D*, and increasing confinement, L^{-2}, as described here by the unit-free parameter characteristic motion (M_{C}). Thus, when M_{C} becomes large due to in-frame motion or confinement, Gaussian-fitting STICS is a biased method for estimating the diffusion coefficient. Interestingly, this bias can be removed by directly calculating the variances of the correlation function rather than estimating the variance by Gaussian fitting (Figs. 2 *C* and 3 *C*), and we provide an analytical description of this process. Furthermore, because this unbiased direct calculation of the correlation function variances is less precise than Gaussian fitting and therefore not desirable for the treatment of noisy experimental data, the simulations described in this article provide a way to remove the bias when Gaussian-fitting STICS is used. In particular, the plot of measured M_{C} versus unbiased M_{C} in Fig. 6 *A*, together with the definition of M_{C} in Eq. 3B, provides a look-up table for all experimental parameter sets in future experiments where in-frame motion and confinement causes Gaussian-fitting STICS to measure a biased value of M_{C}. In this way, some of the precision of Gaussian-fitting STICS is preserved while the measurement bias is eliminated. Still, in the future, it would be desirable to solve Eqs. 6 and 7 and replace the Gaussian fit function entirely with a more accurate description of a correlation function computed from a movie of a highly confined diffusing fluorescent molecule that exhibits in-frame motion; that is beyond the scope of this work.

Overall, Fig. 4 illustrates one of the primary strengths of STICS: when $M_{\text{C}} > 50$, in-frame motion dominates and SPT fails entirely to estimate the diffusion coefficient because of data loss, as shown in Fig. 1 *A*. Thus, although the experimental data in Figs. 5 and 6 were not analyzable by SPT, diffusion coefficients were still recovered by STICS. Based on this article, independent researchers in single-molecule fields should be able to determine the value of characteristic motion in their experiments and decide which analysis method (STICS or SPT) produces the most precise or least biased estimate of the diffusion coefficient for the relevant experimental regime. This decision can be made based on performance or pragmatism: for instance, the timing and optics of stroboscopic illumination can be difficult to implement, especially in commercial single-molecule instruments, and current EMCCD camera technology requires a very small imaging area for the highest accessible frame rates. The relative precision of STICS compared to SPT at large M_C makes STICS a promising and facile method for the estimation of diffusion coefficients in experimental measurements of fast diffusion that is confined within small cells and organelles, within nanomaterials, and in any other highly confined system where the imaging camera frame rate is longer than the time that the molecule takes to explore the confinement volume.

SUPPORTING MATERIAL

Four figures, one table, Matlab codes, and one movie are available at http://www.biophysj.org/biophysj/supplemental/S0006-3495(16)30218-1.

AUTHOR CONTRIBUTIONS

D.J.R. and J.S.B. designed the research. D.J.R. performed the simulations. H.H.T. performed the experiments. D.J.R. analyzed the data. D.J.R., H.H.T., and J.S.B. discussed all the results. D.J.R. wrote the article, which was edited by all authors. All authors read and approved the final manuscript.

ACKNOWLEDGMENTS

Thanks to Xiaowei Zhuang for generously providing the mMaple3 plasmid. Thanks also to Sarah Veatch and Matthew Stone for insightful discussions.

The authors were supported by funding from a Burroughs Wellcome Career Award at the Scientific Interface (to J.S.B.) and by National Science Foundation grant DB-1306434 (to H.H.T.).

REFERENCES

1. Nienhaus, K., and G. U. Nienhaus. 2014. Fluorescent proteins for live-cell imaging with super-resolution. *Chem. Soc. Rev.* 43:1088–1106.
2. Rust, M. J., M. Bates, and X. Zhuang. 2006. Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM). *Nat. Methods.* 3:793–795.
3. Betzig, E., G. H. Patterson, …, H. F. Hess. 2006. Imaging intracellular fluorescent proteins at nanometer resolution. *Science.* 313:1642–1645.
4. Hess, S. T., T. P. Girirajan, and M. D. Mason. 2006. Ultra-high resolution imaging by fluorescence photoactivation localization microscopy. *Biophys. J.* 91:4258–4272.
5. Haas, B. L., J. S. Matson, …, J. S. Biteen. 2015. Single-molecule tracking in live *Vibrio cholerae* reveals that ToxR recruits the membrane-bound virulence regulator TcpP to the *toxT* promoter. *Mol. Microbiol.* 96:4–13.
6. Uphoff, S., R. Reyes-Lamothe, …, A. N. Kapanidis. 2013. Single-molecule DNA repair in live bacteria. *Proc. Natl. Acad. Sci. USA.* 110:8063–8068.
7. Hammar, P., M. Walldén, …, J. Elf. 2014. Direct measurement of transcription factor dissociation excludes a simple operator occupancy model for gene regulation. *Nat. Genet.* 46:405–408.
8. Sanamrad, A., F. Persson, …, J. Elf. 2014. Single-particle tracking reveals that free ribosomal subunits are not excluded from the *Escherichia coli* nucleoid. *Proc. Natl. Acad. Sci. USA.* 111:11413–11418.
9. Karunatilaka, K. S., E. A. Cameron, …, J. S. Biteen. 2014. Superresolution imaging captures carbohydrate utilization dynamics in human gut symbionts. *mBio.* 5:e02172-14.
10. Axelrod, D., D. E. Koppel, …, W. W. Webb. 1976. Mobility measurement by analysis of fluorescence photobleaching recovery kinetics. *Biophys. J.* 16:1055–1069.
11. Sochacki, K. A., I. A. Shkel, …, J. C. Weisshaar. 2011. Protein diffusion in the periplasm of *E. coli* under osmotic stress. *Biophys. J.* 100:22–31.
12. Elowitz, M. B., M. G. Surette, …, S. Leibler. 1999. Protein mobility in the cytoplasm of *Escherichia coli*. *J. Bacteriol.* 181:197–203.
13. Kusumi, A., Y. Sako, and M. Yamamoto. 1993. Confined lateral diffusion of membrane receptors as studied by single particle tracking (nanovid microscopy). Effects of calcium-induced differentiation in cultured epithelial cells. *Biophys. J.* 65:2021–2040.
14. Lill, Y., W. A. Kaserer, …, K. Ritchie. 2012. Single-molecule study of molecular mobility in the cytoplasm of *Escherichia coli*. *Phys. Rev. E Stat. Nonlin. Soft Matter Phys.* 86:021907.
15. Saxton, M. J., and K. Jacobson. 1997. Single-particle tracking: applications to membrane dynamics. *Annu. Rev. Biophys. Biomol. Struct.* 26:373–399.
16. Donehue, J. E., E. Wertz, …, J. S. Biteen. 2014. Plasmon-enhanced brightness and photostability from single fluorescent proteins coupled to gold nanorods. *J. Phys. Chem. C Nanomater. Interfaces.* 118:15027–15035.
17. Elf, J., G.-W. Li, and X. S. Xie. 2007. Probing transcription factor dynamics at the single-molecule level in a living cell. *Science.* 316:1191–1194.
18. Hebert, B., S. Costantino, and P. W. Wiseman. 2005. Spatiotemporal image correlation spectroscopy (STICS) theory, verification, and application to protein velocity mapping in living CHO cells. *Biophys. J.* 88:3601–3614.
19. Di Rienzo, C., E. Gratton, …, F. Cardarelli. 2013. Fast spatiotemporal correlation spectroscopy to determine protein lateral diffusion laws in live cell membranes. *Proc. Natl. Acad. Sci. USA.* 110:12307–12312.
20. Petersen, N. O., P. L. Höddelius, …, K.-E. Magnusson. 1993. Quantitation of membrane receptor distributions by image correlation spectroscopy: concept and application. *Biophys. J.* 65:1135–1146.
21. Pozzi, D., C. Marchini, …, G. Caracciolo. 2013. Mechanistic understanding of gene delivery mediated by highly efficient multicomponent envelope-type nanoparticle systems. *Mol. Pharm.* 10:4654–4665.
22. Brown, C. M., B. Hebert, …, P. W. Wiseman. 2006. Probing the integrin-actin linkage using high-resolution protein velocity mapping. *J. Cell Sci.* 119:5204–5214.
23. Kolin, D. L., D. Ronis, and P. W. Wiseman. 2006. k-Space image correlation spectroscopy: a method for accurate transport measurements independent of fluorophore photophysics. *Biophys. J.* 91:3061–3075.
24. Abu-Arish, A., E. Pandzic, …, P. W. Wiseman. 2015. Cholesterol modulates CFTR confinement in the plasma membrane of primary epithelial cells. *Biophys. J.* 109:85–94.
25. Kisley, L., R. Brunetti, …, C. F. Landes. 2015. Characterization of porous materials by fluorescence correlation spectroscopy super-resolution optical fluctuation imaging. *ACS Nano.* 9:9158–9166.
26. Hendrix, J., V. Baumgärtel, …, D. C. Lamb. 2015. Live-cell observation of cytosolic HIV-1 assembly onset reveals RNA-interacting Gag oligomers. *J. Cell Biol.* 210:629–646.
27. Digman, M. A., P. Sengupta, …, E. Gratton. 2005. Fluctuation correlation spectroscopy with a laser-scanning microscope: exploiting the hidden time structure. *Biophys. J.* 88:L33–L36.

28. Semrau, S., and T. Schmidt. 2007. Particle image correlation spectroscopy (PICS): retrieving nanometer-scale correlations from high-density single-molecule position data. *Biophys. J.* 92:613–621.

29. Digman, M. A., C. M. Brown, …, E. Gratton. 2005. Measuring fast dynamics in solutions and cells with a laser scanning microscope. *Biophys. J.* 89:1317–1327.

30. Bakshi, S., B. P. Bratton, and J. C. Weisshaar. 2011. Subdiffraction-limit study of Kaede diffusion and spatial distribution in live *Escherichia coli*. *Biophys. J.* 101:2535–2544.

31. Deville, S., R. Penjweini, …, M. Ameloot. 2015. Intracellular dynamics and fate of polystyrene nanoparticles in A549 Lung epithelial cells monitored by image (cross-) correlation spectroscopy and single particle tracking. *Biochim. Biophys. Acta.* 1853:2411–2419.

32. Chiu, C. L., J. S. Aguilar, …, M. A. Digman. 2014. Nanoimaging of focal adhesion dynamics in 3D. *PLoS One.* 9:e99896.

33. Storti, B., C. Di Rienzo, …, F. Beltram. 2015. Unveiling TRPV1 spatio-temporal organization in live cell membranes. *PLoS One.* 10:e0116900.

34. Meucci, S., M. Travagliati, …, M. Cecchini. 2014. Tubeless biochip for chemical stimulation of cells in closed-bioreactors: anti-cancer activity of the catechin-dextran conjugate. *RSC Adv.* 4:35017–35026.

35. Moerner, W. E., and D. P. Fromm. 2003. Methods of single-molecule fluorescence spectroscopy and microscopy. *Rev. Sci. Instrum.* 74:3597–3619.

36. Liao, Y., S. K. Yang, …, J. S. Biteen. 2012. Heterogeneous single-molecule diffusion in one-, two-, and three-dimensional microporous coordination polymers: directional, trapped, and immobile guests. *Nano Lett.* 12:3080–3085.

37. Wang, S., J. R. Moffitt, …, X. Zhuang. 2014. Characterization and development of photoactivatable fluorescent proteins for single-molecule-based superresolution imaging. *Proc. Natl. Acad. Sci. USA.* 111:8452–8457.

38. Haas, B. L., J. S. Matson, …, J. S. Biteen. 2014. Imaging live cells at the nanometer-scale with single-molecule microscopy: obstacles and achievements in experiment optimization for microbiology. *Molecules.* 19:12116–12149.

39. Greenwood, D., and F. O'Grady. 1974. The comparative performance of β-lactam antibiotics against ampicillin sensitive *Escherichia coli* in conditions simulating those of the infected urinary bladder. *Br. J. Exp. Pathol.* 55:245–250.

40. Tuson, H. H., and J. S. Biteen. 2015. Unveiling the inner workings of live bacteria using super-resolution microscopy. *Anal. Chem.* 87:42–63.

41. Thompson, R. E., D. R. Larson, and W. W. Webb. 2002. Precise nanometer localization analysis for individual fluorescent probes. *Biophys. J.* 82:2775–2783.

42. Persson, F., I. Barkefors, and J. Elf. 2013. Single molecule methods with applications in living cells. *Curr. Opin. Biotechnol.* 24:737–744.

43. Cattoni, D. I., J. B. Fiche, and M. Nöllmann. 2012. Single-molecule super-resolution imaging in bacteria. *Curr. Opin. Microbiol.* 15:758–763.

44. Xia, T., N. Li, and X. Fang. 2013. Single-molecule fluorescence imaging in living cells. *Annu. Rev. Phys. Chem.* 64:459–480.

45. Lill, Y., W. A. Kaserer, …, K. Ritchie. 2012. Single-molecule study of molecular mobility in the cytoplasm of *Escherichia coli*. *Phys. Rev. E.* 86:021907.

Biophysical Journal, Volume *110*

Supplemental Information

Resolving Fast, Confined Diffusion in Bacteria with Image Correlation Spectroscopy

David J. Rowland, Hannah H. Tuson, and Julie S. Biteen

Contents

- SI Figures S1 – S4
- SI Table S1
- SI Movie S1 Caption

Matlab codes for generating simulation data (dataGen.m) and for the single-particle tracking (SPT) and spatio-temporal image correlation spectroscopy (STICS) analyses (dataParse.m) and accessory code (bpassDJR.m, gaussFit.m)

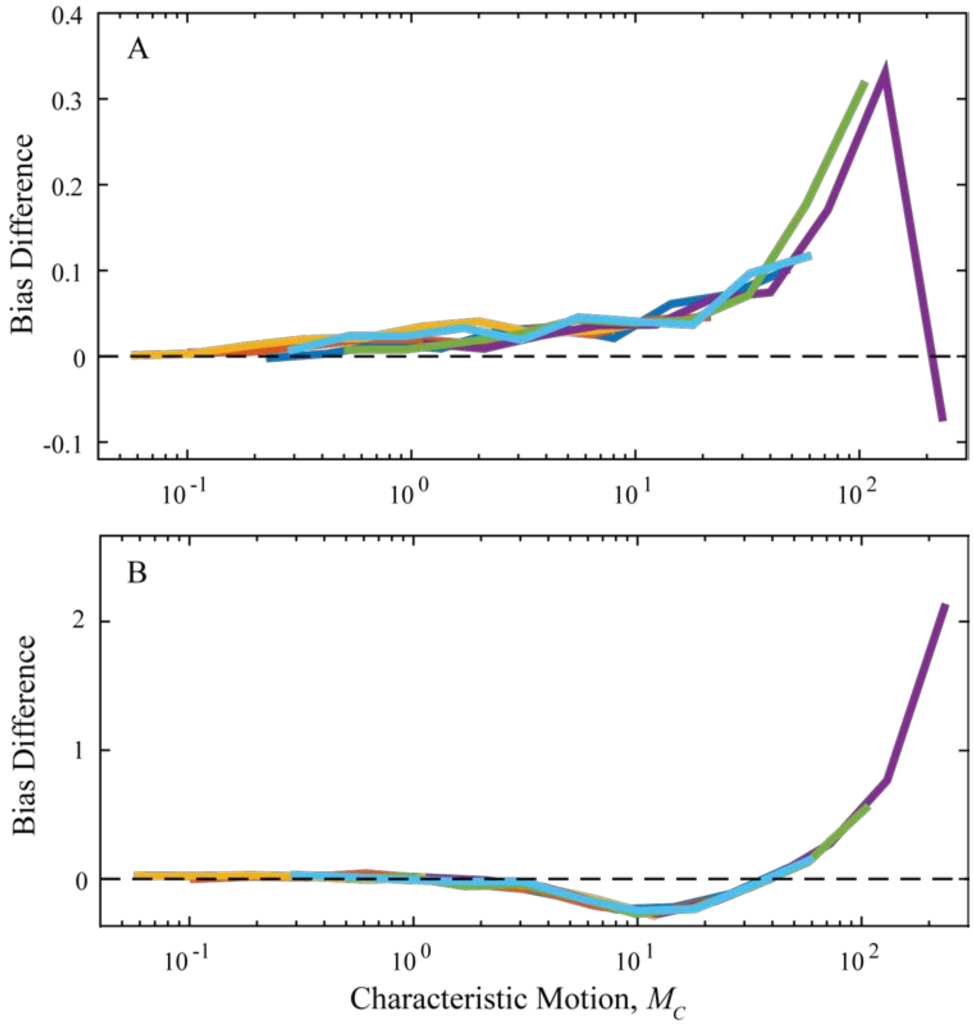


Figure S1. (A) Figures 3A and 3B showed the bias in determination by Gaussian fitting STICS of the diffusion coefficient, *D*, in the presence and absence of in-frame motion, respectively. Here, the difference between these two biases shows that removing in-frame motion does decrease the bias, especially at $M_C > 10$. (B) Figures 3A and 3C showed the bias in determination of the diffusion coefficient, *D*, by Gaussian fitting STICS and by STICS with direct computation of the variance, respectively. Here, the difference between these two biases highlights the large magnitude of the confinement artifact in Gaussian fitting STICS, which overwhelms the in-frame motion artifact in magnitude. Colors indicate the set of parameters (t_{frame}, σ_{PSF}) (SI Table S1) and M_C is defined according to Eq. 3B.

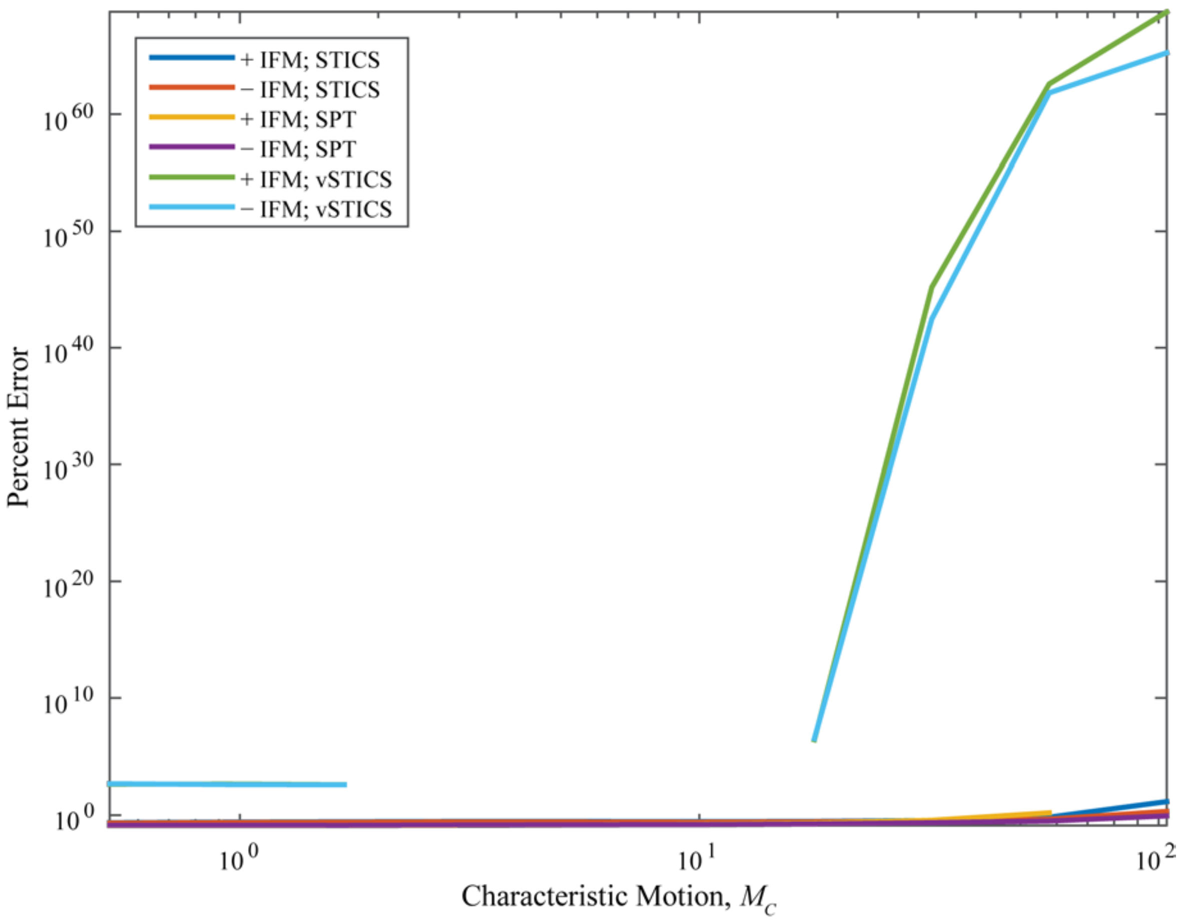


Figure S2. Measurement variance of STICS with the two methods of variance computation. Dark blue, red, yellow, and purple curves as in Figure 4. The green and light blue lines give the percent error when STICS is used to analyze data with and without in-frame motion (IFM), when the Gaussian-fitting step is replaced by direct variance computation. The missing points in the light blue and green curves (direct variance method, vSTICS) represent a complete failure to estimate the value, and even when the analysis algorithm had sufficient data to succeed the percent error was never below 1000%. The M_C is defined according to Eq. 3B.

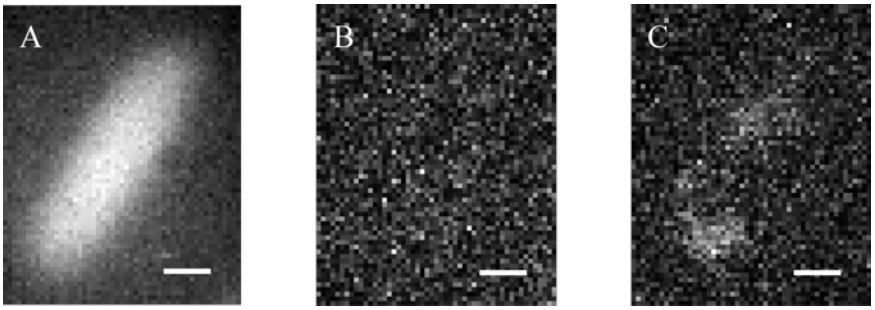


Figure S3. (A) Time-averaged fluorescence image obtained from summing a time series of fluorescence images of mMaple3 diffusion inside an *E. coli* cell (full time series in Movie S1). (B) A single image (t_{frame} = 40 ms) from that time series shows a cell after the mMaple3 molecule has bleached. (C) An image of the same *E. coli* cell in 'B' two frames earlier, showing a typical mMaple3 molecule diffusing so rapidly that it is diffuse over nearly the entire bacterium. Scale bars: 0.5 µm.

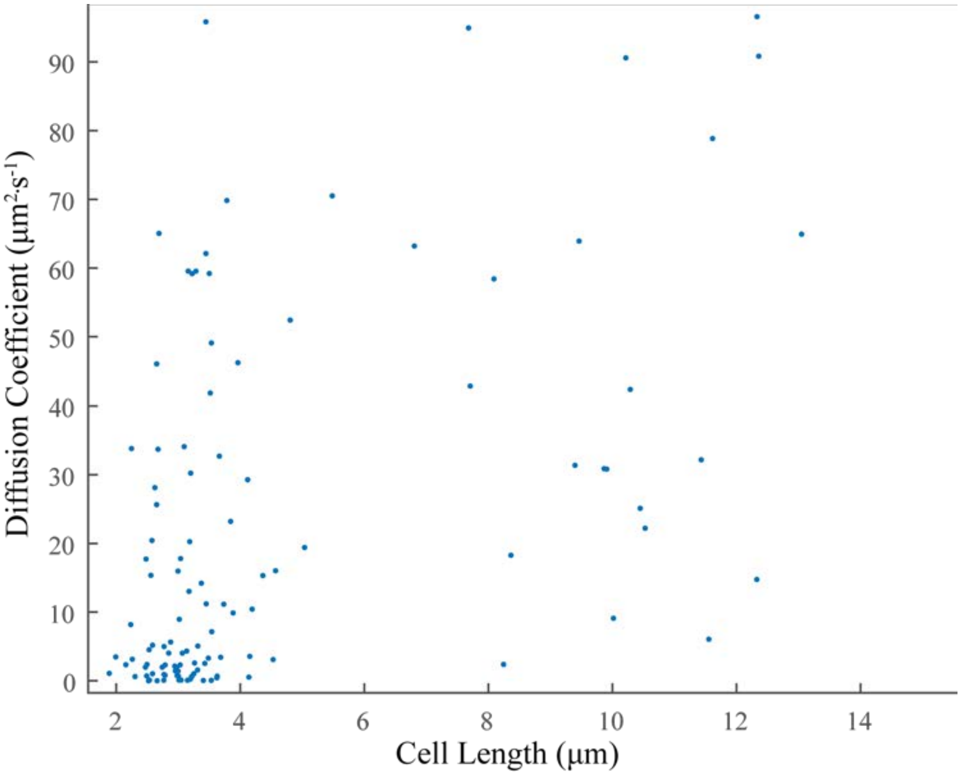
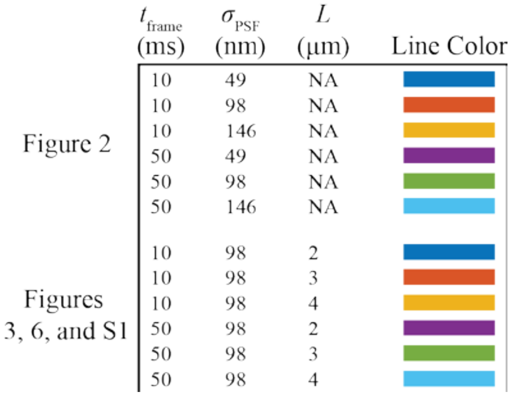


Figure S4. The measured diffusion coefficient for mMaple3 molecules in each *E. coli* cell did not strongly depend on the cell length calculated from the phase contrast image of that cell.

Table S1. Line colors for the simulated data in Figures 2, 3, 6 and S1.



	t_{frame} (ms)	σ_{PSF} (nm)	L (µm)	Line Color
Figure 2	10	49	NA	🟦
	10	98	NA	🟧
	10	146	NA	🟨
	50	49	NA	🟪
	50	98	NA	🟩
	50	146	NA	🟦
Figures 3, 6, and S1	10	98	2	🟦
	10	98	3	🟧
	10	98	4	🟨
	50	98	2	🟪
	50	98	3	🟩
	50	98	4	🟦

Movie S1. Photoswitching and imaging of mMaple3 in an *E. coli* cell. To visualize single molecules, a low-power 406-nm laser (shown in the movie as a flash in the background) was used to photoswitch 0 – 1 copies of mMaple3 per cell at a time. Two frames from this movie are presented as Figures S3B and S3C. Imaging rate: 25 frames per second. The movie area is 2.50 µm × 3.14 µm.

MATLAB CODE: bpassDJR.m

```
function res = bpassDJR(image_array,lnoise,lobject,threshold,lzero)
%
% NAME:
%         bpassDJR
% PURPOSE:
%         Implements a real-space bandpass filter that suppresses
%         pixel noise and long-wavelength image variations while
%         retaining information of a characteristic size.
%
% CATEGORY:
%         Image Processing
% CALLING SEQUENCE:
%         res = bpass( image_array, lnoise, lobject, threshold, lzero )
% INPUTS:
%         image:  The two-dimensional array to be filtered.
%         lnoise: Characteristic lengthscale of noise in pixels.
%              Additive noise averaged over this length should
%              vanish. May assume any positive floating value.
%              May be set to 0 or false, in which case only the
%              highpass "background subtraction" operation is
%              performed.
%         lobject: (optional) Integer length in pixels somewhat
%              larger than a typical object. Can also be set to
%              0 or false, in which case only the lowpass
%              "blurring" operation defined by lnoise is done,
%              without the background subtraction defined by
%              lobject.  Defaults to false.
%         threshold: (optional) By default, after the convolution,
%              any negative pixels are reset to 0.  Threshold
%              changes the threshhold for setting pixels to
%              0.  Positive values may be useful for removing
%              stray noise or small particles.  Alternatively, can
%              be set to -Inf so that no threshholding is
%              performed at all.
%
% OUTPUTS:
%         res:   filtered image.
% PROCEDURE:
%         simple convolution yields spatial bandpass filtering.
% NOTES:
% Performs a bandpass by convolving with an appropriate kernel.  You can
% think of this as a two part process.  First, a lowpassed image is
% produced by convolving the original with a gaussian.  Next, a second
% lowpassed image is produced by convolving the original with a boxcar
% function. By subtracting the boxcar version from the gaussian version, we
% are using the boxcar version to perform a highpass.
%
% original - lowpassed version of original => highpassed version of the
% original
%
% Performing a lowpass and a highpass results in a bandpassed image.
%
% Converts input to double.  Be advised that commands like 'image' display
% double precision arrays differently from UINT8 arrays.
% MODIFICATION HISTORY:
%           Written by David G. Grier, The University of Chicago, 2/93.
%
%           Greatly revised version DGG 5/95.
%
%           Added /field keyword JCC 12/95.
%
%           Memory optimizations and fixed normalization, DGG 8/99.
%           Converted to Matlab by D.Blair 4/2004-ish
%
%           Fixed some bugs with conv2 to make sure the edges are
%           removed D.B. 6/05
%
%           Removed inadvertent image shift ERD 6/05
%
%           Added threshold to output.  Now sets all pixels with
%           negative values equal to zero.  Gets rid of ringing which
%           was destroying sub-pixel accuracy, unless window size in
%           cntrd was picked perfectly.  Now centrd gets sub-pixel
%           accuracy much more robustly ERD 8/24/05
%
%           Refactored for clarity and converted all convolutions to
%           use column vector kernels for speed.  Running on my
%           macbook, the old version took ~1.3 seconds to do
%           bpass(image_array,1,19) on a 1024 x 1024 image; this
%           version takes roughly half that. JWM 6/07
%
%       This code 'bpass.pro' is copyright 1997, John C. Crocker and
%       David G. Grier.  It should be considered 'freeware'- and may be
%       distributed freely in its original form when properly attributed.
%
%
% 2016 edits by David J. Rowland, The University of Michigan:
% added lzero to the input list instead of just declaring it inside the function.

if nargin < 3, lobject = false; end
if nargin < 4, threshold = 0; end

normalize = @(x) x/sum(x);

image_array = double(image_array);

if lnoise == 0
  gaussian_kernel = 1;
else
  gaussian_kernel = normalize(...
    exp(-((-ceil(5*lnoise):ceil(5*lnoise))/(2*lnoise)).^2));
end

if lobject
  boxcar_kernel = normalize(...
      ones(1,length(-round(lobject):round(lobject))));
end

% JWM: Do a 2D convolution with the kernels in two steps each.  It is
% possible to do the convolution in only one step per kernel with
```

```
%
  % gconv = conv2(gaussian_kernel',gaussian_kernel,image_array,'same');
  % bconv = conv2(boxcar_kernel', boxcar_kernel,image_array,'same');
%
% but for some reason, this is slow.  The whole operation could be reduced
% to a single step using the associative and distributive properties of
% convolution:
%
  % filtered = conv2(image_array,...
  %   gaussian_kernel'*gaussian_kernel - boxcar_kernel'*boxcar_kernel,...
  %   'same');
%
% But this is also comparatively slow (though inexplicably faster than the
% above).  It turns out that convolving with a column vector is faster than
% convolving with a row vector, so instead of transposing the kernel, the
% image is transposed twice.

gconv = conv2(image_array',gaussian_kernel','same');
gconv = conv2(gconv',gaussian_kernel','same');

if lobject
  bconv = conv2(image_array',boxcar_kernel','same');
  bconv = conv2(bconv',boxcar_kernel','same');

  filtered = gconv - bconv;
else
  filtered = gconv;
end

% commented out because why do it in the first place?
% % Zero out the values on the edges to signal that they're not useful.
% lzero = max(lobject,ceil(5*lnoise));
% lzero=0;

filtered(1:(round(lzero)),:) = 0;
filtered((end - lzero + 1):end,:) = 0;
filtered(:,1:(round(lzero))) = 0;
filtered(:,(end - lzero + 1):end) = 0;

% JWM: I question the value of zeroing out negative pixels.  It's a
% nonlinear operation which could potentially mess up our expectations
% about statistics.  Is there data on 'Now centroid gets subpixel accuracy
% much more robustly'?  To choose which approach to take, uncomment one of
% the following two lines.
% ERD: The negative values shift the peak if the center of the cntrd mask
% is not centered on the particle.

% res = filtered;
filtered(filtered < threshold) = 0;
res = filtered;
end
```

MATLAB CODE: dataGen.m

```
function [v, simProps] = dataGen(varargin)
%
% NAME:
%       dataGen
% PURPOSE:
%       Generates space and time resolved single-molecule imaging data of a
%                               single diffuser.
% CATEGORY:
%       Data Simulation
% CALLING SEQUENCE:
%       [v, simProps] = dataGen(boundaryCondition, blurFlag);
% INPUTS:
%       varargin:  use paired inputs to set the property (input 1) to the
%          value (input 2) desired.
%
%       Properties:             Descriptions:
%
%       D                       diffusion coefficient in microns^2/s
%
%       tFrame                  image frame integration time in seconds
%
%       pixSize                 Pixel size in micrometers
%
%       psfSize                 standard deviation (width) of the microscope's
%                           point spread function in micrometers. i.e.
%                           FWHM = sqrt(2*log(2)) * psfSize
%
%       celSize                 1x2 vector: [width (diameter), height
%                           (length)] of the bounding box (cylinder)
%                           depending on the confinement condition
%
%       nFrames                  number of frames to be simulated
%
%       SNR                     signal to noise ratio (ratio of maximum
%                           signal amplitude to standard deviation of
%                           background noise)
%
%       confBool                  1 for confined to the interior of a
%                           cylinder, and 0 for free diffusion.
%
%       blurFlag                 1 for blurry motion, 0 for 'stroboscopic
%                           illumination'
%
% OUTPUTS:
%       v:                          simulated image time sequence
%       simProps:                   properties of the simulation in the Matlab structure format
% PROCEDURE:
%       1. Simulate molecular trajectory
%       2. Evaluate pixel intensities
%                       3. Add noise
% MODIFICATION HISTORY:
%       Written by David J. Rowland, The University of Michigan, 3/16.
% NOTES:
%       This code 'dataGen.m' should be considered 'freeware'- and may be
%       distributed freely in its original form when properly attributed.
```

```
%
%       For testing purposes, this line makes a movie of confined, blurry
%       diffusion.
%
%       v=dataGen('confined', 1)


%% Default simulation parameters
D = .01;                % diffusion coefficient in microns^2/s
tFrame = .05;                           % frame integration time in seconds
pixSize = .049;                         % width of pixels in microns
psfSize = .098;                         % s.d. of the psf in microns
celSize = [1,3];        % [width, length] of confinement cylinder in microns
nFrames = 1e3;                          % number of frames in the simulated movie
SNR = 20;                                        % signal to noise ratio for added white noise (0:inf)
confBool = 1;           % 'confined' or 'unconfined'
blurFlag = 1;           % include blur subframes or not

% algorithmically-determined image size designed to disallow edge effects
imSize = ceil(celSize/pixSize+4*ceil(psfSize/pixSize));

% check variable size
if prod([imSize(1:2),nFrames])*8/1e9>1
    warning('video is over a GB. manually pass this block if you wish to continue')
end

% minimum increment of speed in units of microns^2/subframe
dtRef = 0.0001;

% initialize simulation properties structure
simProps.D = D;
simProps.tFrame = tFrame;
simProps.pixSize = pixSize;
simProps.psfSize = psfSize;
simProps.celSize = celSize;
simProps.nFrames = nFrames;
simProps.SNR = SNR;
simProps.blurFlag = blurFlag;
simProps.nSubs = [];
simProps.dtRef = dtRef;
simProps.confBool = confBool;

% if any sim parameters are included as inputs, change the simulation
% parameters mentioned
if ~rem(nargin,2)
    fNames=fieldnames(simProps);
    for ii=1:2:nargin
        whichField = strcmp(fNames,varargin{ii});

        if all(~whichField)
            warning('Check spelling. Parameter change may have not occurred')
        end

        eval([fNames{whichField} ' = varargin{ii+1}'])
        eval(['simProps.' fNames{whichField} ' = ' fNames{whichField},';'])
    end
```

```
elseif ~rem(nargin,1)
    warning('use paired inputs')
    v=[];
    return
end

% number of subframes required
nSubs = ceil(D*tFrame/dtRef);

% update the value of the diffusion coefficient since rounding may change it.
D = nSubs*dtRef/tFrame;

%% Trajectory generation
if confBool
    % confined particle trajectory
    mLocs = zeros(3, nFrames);
    for ii = 1 : nFrames*nSubs-1    % this loop can be compiled to mex64 to increase its speed
        % three 1d steps pulled from normal distribution with variance 2*dtRef
        step = sqrt(2*dtRef) * randn(3,1);

        candPos = mLocs(:,ii) + step;
        prevPos = mLocs(:,ii);

        % the ordering in the celSize vector matters because of this line:
        r = celSize(1)/2;

        % if the candidate position is outside of the cylinder in the x/z dimensions, reflect the step
        % against the inside of the cylinder. path length is preserved.
        if sqrt(sum(candPos([1,3]).^2)) > celSize(1)/2
            m = (candPos(3)-prevPos(3)) / (candPos(1)-prevPos(1));
            b = candPos(3)-m*candPos(1);

            xi=[(-m*b+sqrt(-b^2+r^2+m^2*r^2))/(1+m^2),...
                (-m*b-sqrt(-b^2+r^2+m^2*r^2))/(1+m^2)];
            yi=m*xi+b;

            % there are two solutions. the one closest to the candidate position is chosen. the farther
            % one is on the other side of the cell.
            whichone = (xi-candPos(1)).^2 + (yi-candPos(3)).^2 + (xi-prevPos(1)).^2 + (yi-prevPos(3)).^2;
            xip = [xi(find(whichone == min(whichone))),yi(find(whichone == min(whichone)))];

            normv = -xip/sqrt(sum(xip.^2));
            l = sqrt(sum((candPos([1,3])'-xip).^2));
            pf = 2*sum((prevPos([1,3])'-xip).*normv)*normv-(prevPos([1,3])'-xip);
            pf = pf/sqrt(sum(pf.^2))*l+xip;

            % replace x/z components of the position with the reflected x/z components
            out=candPos;
            out([1,3])=pf;

            candPos=out;
        end

        % if the candidate position is outside of the cylinder in the y dimension (cell's long axis)
        if candPos(2) < -celSize(2)/2
```

```
            candPos(2) = 2*-celSize(2)/2 - candPos(2);
        end
        if candPos(2) > celSize(2)/2
            candPos(2) = 2*celSize(2)/2 - candPos(2);
        end

        mLocs(:, ii+1) = candPos;
    end
else

    % unconfined particle trajectory
    mLocs=cumsum(sqrt(2*dtRef) * randn(3,nFrames*nSubs),2);
end

%% movie generation

if blurFlag
    % arrange tracks for subframe averaging
    tr_x = zeros(nSubs, nFrames);
    tr_y = zeros(nSubs, nFrames);
    for ii = 1:nFrames
        tr_x(:, ii) = mLocs(1, 1+(ii-1)*nSubs : ii*nSubs);
        tr_y(:, ii) = mLocs(2, 1+(ii-1)*nSubs : ii*nSubs);
    end
else
    % just use the first subframe from each frame
    tr_x = mLocs(1, 1:nSubs:end);
    tr_y = mLocs(2, 1:nSubs:end);
end

% shift to positive values
tr_x=tr_x+celSize(1)/2;
tr_y=tr_y+celSize(2)/2;

% noiseless pixel intensities
v=zeros([imSize(1:2),nFrames]); cx = 0; cy = 0;
for ii=-2*psfSize:pixSize:celSize(1)+2*psfSize        % x pixel locations with padding
    cx = cx+1;
    for jj=-2*psfSize:pixSize:celSize(2)+2*psfSize    % y pixel locations with padding
        cy = cy+1;

        % symmeteric gaussian function approximation of Airy Disk
        v(cx,cy,:) = mean(exp(-((ii-tr_x).^2+(jj-tr_y).^2)/2/psfSize^2),1);
    end
    cy = 0;
end

% add white noise
v = v + 1/SNR*randn(size(v));

simProps.D = D;
simProps.tFrame = tFrame;
simProps.pixSize = pixSize;
simProps.psfSize = psfSize;
simProps.celSize = celSize;
simProps.nFrames = nFrames;
simProps.SNR = SNR;
simProps.blurFlag = blurFlag;
simProps.nSubs = nSubs;
simProps.dtRef = dtRef;
simProps.confBool = confBool;
end
```

MATLAB CODE: dataParse.m

```
function resStruct = dataParse(data, mask)
%
% NAME:
%       dataParse
% PURPOSE:
%       Analyze single-molecule imaging data with tracking or STICS
% CATEGORY:
%       Image Processing
% CALLING SEQUENCE:
%       resStruct = dataParse(data,flag);
% INPUTS:
%       data:           x by y by t movie of fluorescence data
%       mask:            (optional) 2d binary roi selection mask
% OUTPUTS:
%       resStruct:       fitting result 'structure'
% PROCEDURE:
%       1. Peak fitting, then MSD calculation a la single molecule analysis
%       2. Correlation function calculation and width estimation
%       3. Diffusion coefficient estimation by MSD fitting
% MODIFICATION HISTORY:
%       Written by David J. Rowland, The University of Michigan, 3/16.
% NOTES:
%       This code 'dataParse.m' should be considered 'freeware'- and may be
%       distributed freely in its original form when properly attributed.
%
%       For testing purposes, run this script:
%
%       [v,sP]=dataGen('D',.1,'tFrame',.05,'nFrames',500);
%       r = dataParse(v)
%       plot(r.iMSDs); hold all
%       plot(r.MSDs)
%       plot(r.MSDd); hold off


%% analysis parameters
tFrame = 0.05;          % camera integration time in seconds
nTau = 5;               % number of smallest time lag values to use (excluding 0)
pixelSize = .049;       % pixel size in microns
nFrames = size(data,3); % number of frames in the data
yesOverlap = 1;         % use overlapping time lags?
isConfined = 1;         % use confined MSD curve fit?
padStyle = 1;           % padding style for STICS analysis
                        % 1 : pad zeros, 2: pad mean, 3: pad mean outside mask

%% single molecule tracking analysis

% spot fitting.
fitP = zeros(nFrames, 6);
parfor ii = 1:nFrames
    % this loop may be parallelized by simply replaceing 'for' with 'parfor'
    % and starting a parallel pool before running the code.
    fitP(ii,:) = gaussFit(data(:,:,ii));
end

% 'tracking'. missed spots will register as nans.
```

```
tr = fitP(:,1:2) * pixelSize;

% calculate mean squared displacements, one for each time lag value
MSDs=zeros(nTau,2);
for ii=1:nTau
  if yesOverlap                 % overlapping frame pairs
    indvec1=ii+1:nFrames;
    indvec2=1:nFrames-ii;
  elseif ~yesOverlap            % nonoverlapping frame pairs
    indvec2=1:ii:nFrames;
    indvec1=indvec2(1:end-1);
    indvec2=indvec2(2:end);
  end

  % mean squared displacements vs time lag
  MSDs(ii,:)=nanmean((tr(indvec2,:)-tr(indvec1,:)).^2,1);
end

%% STICS analysis

% pad images
switch padStyle
  case 1
    data = padZeros(data);
  case 2
    data = padMean(data);
  case 3
    data = padMask(data,mask);
end

% time-space correlation function calculation
if yesOverlap                   % overlapping frame pairs
  famps=abs(fft(fft(fft(data,[],1),[],2),[],3)).^2;
  STCorr = fftshift(fftshift(real(ifft(ifft(ifft(famps...
    ,[],1),[],2),[],3)),1),2)/numel(famps)/mean(data(:))^2-1;
  STCorr = STCorr(:,:,2:nTau+1);

elseif ~yesOverlap              % nonoverlapping frame pairs
  vFft=fft2(data);
  STCorr=zeros(size(vFft,1),size(vFft,2),nTau+1);

  STCorr(:,:,1)=mean(bsxfun(@times,real(fftshift(fftshift(...
    ifft2(vFft.*conj(vFft)),1),2)),1./(mean(mean(data)).^2)),3);

  for kk=1:nTau
    ind1 = 1:kk:vidsize(3);
    ind2 = ind1(2:end);
    ind1 = ind1(1:end-1);
    STCorr(:,:,kk+1) = mean(bsxfun(@times,real(fftshift(fftshift( ...
      ifft2(vFft(:,:,ind2).*conj(vFft(:,:,ind1))),1),2)), ...
      1./mean(mean(data(:,:,ind1)))./mean(mean(data(:,:,ind2)))),3);
  end
  STCorr=STCorr/numel(vFft(:,:,1))-1;
end

% estimate the widths of the correlation function
```

```
iMSDs=zeros(nTau,2); MSDd=iMSDs;
[x,y]=ndgrid(1:size(STCorr,1),1:size(STCorr,2));
for ii = 1:nTau
    fitP = gaussFit(STCorr(:,:,ii),'widthGuess',5,'nPixels',...
        min(size(STCorr(:,:,1))),'findTheSpot',0);
    iMSDs(ii,:) = fitP(3:4).^2 * pixelSize^2;

    % discrete variance calculation for x dimension
    pmf=sum(STCorr(:,:,ii)/sum(sum(STCorr(:,:,ii))),2);
    MSDd(ii,1) = sum(pmf.*(x(:,1)-mean(x(:))).^2) * pixelSize^2;

    % discrete variance calculation for x dimension
    pmf=sum(STCorr(:,:,ii)/sum(sum(STCorr(:,:,ii))),1);
    MSDd(ii,2) = sum(pmf.*(y(1,:)-mean(y(:))).^2) * pixelSize^2;
end

%% MSD Fitting

% choose fitting function
if isConfined
    f=@(p,X)sqconfMSD1D(p,X);
    pStart = [.1, .1, 0];
    lb = [0, 0, -inf];
    ub = [inf, inf, inf];
else
    f=@(p,X) 2*p(2)*X+p(1);
    pStart = [0, .1];
    lb = [-inf, 0];
    ub = [inf, inf];
end

% time lag domain vector
tau = (1:nTau)'*tFrame;

% D from tracking
pT(:,1)=lsqcurvefit(f,pStart,tau,MSDs(:,1),lb,ub);
pT(:,2)=lsqcurvefit(f,pStart,tau,MSDs(:,2),lb,ub);

% D from STICS
pS(:,1)=lsqcurvefit(f,pStart,tau,iMSDs(:,1),lb,ub);
pS(:,2)=lsqcurvefit(f,pStart,tau,iMSDs(:,2),lb,ub);

% D from 'discrete variance'
pD(:,1)=lsqcurvefit(f,pStart,tau,MSDd(:,1),lb,ub);
pD(:,2)=lsqcurvefit(f,pStart,tau,MSDd(:,2),lb,ub);

resStruct.Dtracking = pT(2,:);
resStruct.Dstics =  pS(2,:);
resStruct.Dvar =  pD(2,:);
resStruct.MSDs = MSDs;
resStruct.iMSDs = iMSDs;
resStruct.MSDd = MSDd;

end

function zi=sqconfMSD1D(p,X)
```

```
l=p(1);
d=exp(p(2));
ns=p(3);
tau=X;

summedTerm=@(t,d,l,n)1/n^4*exp(-(n*pi/l).^2*d*t);

temp=eps*ones(size(tau));
for ii=1:2:2*400-1
    s=summedTerm(tau,d,l,ii);
    if sum(s./temp)<1e-10
        break
    end
    temp=temp+s;
end
zi=l^2/6*(1-96/pi^4*temp)+ns;
zi(isnan(zi))=eps;
zi(isinf(zi))=eps;
end

function imStack=padMask(imStack,mask)
% replace pixels outside the mask with the average value inside the mask in
% each frame

imsize=size(imStack);
mMean=mean(reshape(imStack(mask(:,:,ones(1,imsize(3)))),[],imsize(3)));
imStack(~mask(:,:,ones(1,imsize(3))))=mMean(ones(1,sum(~mask(:))),:);
end

function imstack=padMean(imstack)
% pad the first two dimensions to double size with the mean of each image
imstack=mat2cell(imstack,size(imstack,1),size(imstack,2),ones(1,size(imstack,3)));
imm=cellfun(@(x)mean(mean(x(mask))),imstack,'uniformoutput',false);

imstack=cellfun(@(x,y)padarray(x,floor(size(x)/2),y),imstack,imm,...
    'uniformoutput',false);
imstack=cat(3,imstack{:});
end

function imstack=padZeros(imstack)
% pad the first two dimensions to double size with zeros
imstack=mat2cell(imstack,size(imstack,1),size(imstack,2),ones(1,size(imstack,3)));

imstack=cellfun(@(x,y)padarray(x,floor(size(x)/2),y),...
    imstack,repmat({0},1,1,size(imstack,3)),'uniformoutput',false);
imstack=cat(3,imstack{:});
end
```

MATLAB CODE: gaussFit.m

```
function [fitPars, conf95]=gaussFit(img, varargin) %findTheSpot, plottingFlag)
%
% NAME:
%       gaussFit
% PURPOSE:
%       Fits a generalized gaussian function to 2d imaging data. This code
%       produces results in units of pixels for the center position and
%       widths.
% CATEGORY:
%       Image Processing
% CALLING SEQUENCE:
%       [fitPars, conf95] = gaussFit(img,findTheSpot);
% INPUTS:
%       img:            The two-dimensional array to be fit to a gaussian
%
%       varargin:       use paired inputs to set the property (input 1) to the
%          value (input 2) desired.
%
%       Properties:     Descriptions:
%
%       findTheSpot:    1 or 0. Default behavior is to fit an
%          ROI in the center of the image. If the spot is not near the
%          center or the image is very large, findTheSpot enables the code
%          to first roughly locate the spot and then use that location as
%          the ROI center.
%
%       plottingFlag:   1 or 0. show plotting output. default is 0.
%
%       widthGuess:     set the starting value for the width of the
%                   Gaussian in units of pixels.
%
%       nPixels         pixel width of ROI to be selected from img. default
%                   is 11. the value should be odd.
%
% OUTPUTS:
%       fitPars:        fitting coefficient vector, all units are pixels.
%       fitCI:          95% confidence interval of fitting coefficients at
%                   end of fitting
% PROCEDURE:
%       1. Peak guessing and/or data ROI selection of local area inside img
%       2. Non-linear least squares minimization for 7 (or 6) - parameter
%          Gaussian function on the ROI selected.
% MODIFICATION HISTORY:
%       Written by David J. Rowland, The University of Michigan, 3/16.
% NOTES:
%       This code 'gaussFit.m' should be considered 'freeware'- and may be
%       distributed freely in its original form when properly attributed.
%
%       For testing purposes, run this script:
%
%       img = exp(-x.^2/2/2^2-y.^2/2/3^2)+.02*randn(size(x));
%       p = gaussFit(img,'widthGuess',2);

% if any sim parameters are included as inputs, change the simulation
% parameters mentioned
```

```
if nargin>1
    fNames={'findTheSpot', 'plottingFlag', 'widthGuess', 'nPixels'};
    for ii=1:2:nargin-1
        whichField = strcmp(fNames,varargin{ii});

        if all(~whichField)
            warning('Check spelling. Parameter change may have not occurred.')
        end

        eval([fNames{whichField} ' = varargin{ii+1};'])
    end

elseif rem(nargin,1)
    warning('use paired inputs if using varargin.')

    % empty output. size must change if the gaussian fitting function is changed.
    fitPars = nan(1,6);
    conf95 = nan(1,6);
    return
end

%% declaring fitting predicates

if ~exist('findTheSpot','var')
    findTheSpot = 1;
end
if ~exist('plottingFlag','var')
    plottingFlag = 0;
end
if ~exist('widthGuess','var')
    widthGuess = 2;
end
if ~exist('nPixels','var')
    nPixels = 11;
end

% freely rotating bivariate gaussian function for least squares minimization
% parameters: [xCenter, yCenter, angle, xSD, ySD, amplitude, offset]
% xR=@(x,y,xc,yc,th)(x-xc)*cos(th)-(y-yc)*sin(th);
% yR=@(x,y,xc,yc,th)(x-xc)*sin(th)+(y-yc)*cos(th);
% f=@(p,X) exp( -xR(X(:,1), X(:,2), p(1), p(2), p(3)).^2/2/p(4)^2 + ...
%     -yR( X(:,1), X(:,2), p(1), p(2), p(3)).^2/2/p(5)^2 ) *p(6) + p(7);

% fixed angle fit
% parameters: [xCenter, yCenter, xSD, ySD, amplitude, offset]
th=0;
xR=@(x,y,xc,yc)(x-xc)*cos(th)-(y-yc)*sin(th);
yR=@(x,y,xc,yc)(x-xc)*sin(th)+(y-yc)*cos(th);
f=@(p,X) exp( -xR(X(:,1), X(:,2), p(1), p(2)).^2/2/p(3)^2 + ...
    -yR( X(:,1), X(:,2), p(1), p(2)).^2/2/p(4)^2 ) *p(5) + p(6);

% bounds
lb=[-inf, -inf, 0, 0, -inf, -inf];
ub=[inf, inf, inf, inf, inf, inf];
```

```
%% data selection

if findTheSpot
    % select the local area around a bright spot in a larger image

    % bandpass and threshold
    LP=1;           % low pass value
    HP=10;          % high pass value
    intThresh=0.1;  % intensity threshold. set to zero and then check by inspection
    hMax=0.1;       % larger if the dynamic range of your data is larger
    lzero=4;        % this squelches a 5 pixel boundary around the filtered image

    bIm=bpassDJR(img, LP, HP, intThresh, lzero);

    % watershed algorithm
    extImg=imextendedmax(bIm,hMax);

    % failed watershed algorithm can result in all ones
    if all(extImg(:))
        extImg=extImg-1;
    end

    % shrink to a point. this is the estimated location of the spot
    sIm=bwmorph(extImg,'shrink',inf);

    % the index of the one pixel is a good guess for the particle location
    [locInds(:,1),locInds(:,2)]=find(sIm);

    % temporally coincdident guesses are not treated with this code.
    if size(locInds(:,1))~=1
        fitPars=nan(1,6);
        conf95=nan(1,6);
        return
    end

else
    % otherwise, assume the spot is in near the center of the image
    locInds=round(size(img)/2);
end

% pad the img(s) with nans (removed later).
padsize=[nPixels,nPixels];
padVal=nan;
direction='both';
img=padarray(img,padsize,padVal,direction);
locInds=locInds+nPixels;

% find the selection domain
[sDom1,sDom2]=ndgrid(locInds(1)-(nPixels-1)/2:locInds(1)+(nPixels-1)/2, ...
    locInds(2)-(nPixels-1)/2:locInds(2)+(nPixels-1)/2);
inds=sub2ind(size(img),sDom1(:),sDom2(:));

% select the data
truImg=reshape(img(inds),[nPixels,nPixels]);

%% starting parameter selection for 6-parameter Gaussian Fit
```

```
% x, y centers starting guess
pStart(1)=0;
pStart(2)=0;

% xSD, ySD in units of pixels
pStart(3)=widthGuess;
pStart(4)=widthGuess;

% amplitude, offset
mVals=[max(truImg(:)),min(truImg(:))];
pStart(5)=mVals(1)-mVals(2);
pStart(6)=mVals(2);

%% fitting the data

[x,y]=ndgrid(1:nPixels,1:nPixels);
X=cat(2,x(:),y(:)) - nPixels/2;
[fitPars, ~, residual, ~, ~, ~,jacobian] = ...
    lsqcurvefit(f,pStart,X(~isnan(truImg(:)),:),truImg(~isnan(truImg(:))),...
    lb,ub);

% confidence intervals
conf95 = nlparci(fitPars, residual,'jacobian',jacobian);

%% plot the output
if plottingFlag
    fVals=reshape(f(fitPars,X),[nPixels,nPixels]);
    dVals=truImg;
    sVals=reshape(f(pStart,X),[nPixels,nPixels]);

    subplot(221)
    title('Data')
    pcolor(kron(dVals,ones(10)))
    shading flat; axis image; colorbar

    subplot(222)
    title('starting values')
    pcolor(kron(sVals,ones(10)))
    shading flat; axis image; colorbar

    subplot(223)
    title('fit result')
    pcolor(kron(fVals,ones(10)))
    shading flat; axis image; colorbar

    subplot(224)
    title('residuals')
    pcolor(kron(dVals - fVals,ones(10)))
    shading flat; axis image; colorbar
end

% shift center back to lab frame
fitPars([1,2])=fitPars(1:2)+locInds-nPixels;
end
```