2021/11/21 v0.6 IPOL article class

Published in Image Processing On Line on 2024–02–28.
Submitted on 2023–07–01, accepted on 2024–01–04.
ISSN 2105–1232 © 2024 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2024.496

Localization and Image Reconstruction in a STORM Based Super-resolution Microscope

Pranjal Choudhury, Bosanta Ranjan Boruah

Department of Physics, Indian Institute of Technology Guwahati, India
brboruah@iitg.ac.in

Communicated by Federico Lecumberry and Xavier Bou *Demo edited by* Xavier Bou

Abstract

In this paper, we present a comprehensive Python program for localizing the point spread functions (PSFs) present in a stack of images and thereby rendering a super-resolved image in a Stochastic Optical Reconstruction Microscopy (STORM). A microscope that provides super-resolved images is known as a super-resolution microscope. Optical super-resolution microscopy is playing a pivotal role in advancing the field of optical imaging and has found applications in a number of areas such as cellular biology, biotechnology, medical research, and nanotechnology. The proposed Python program utilizes image processing techniques to accurately identify the PSFs present in highly noisy images with densely packed fluorescent objects. Our program not only provides all the necessary tools for image reconstruction in a STORM microscope under open source license but also offers certain advantages over the existing reconstruction software packages. Some such advantages are an option to start the reconstruction process and the visualization of the rendered super-resolved image in parallel with image acquisition and disposal of the images immediately after acquisition for minimum use of disk space. Parallel visualization of the reconstructed image allows aborting the image acquisition in the case the images are not suitable for super-resolution, thereby saving valuable time. Our Python program is demonstrated using a number of different image stacks. The proposed software code can be applied not only to STORM but also to any other super-resolution technique using single-molecule localization.

Source Code

The source codes and documentation for the algorithms presented in this paper are available from the web page of this article[1]. Usage instructions are included in the `README.md` file of the archive.

Keywords: STORM; SMLM; super resolution microscopy; localization microscopy

[1] https://doi.org/10.5201/ipol.2024.496

1 Introduction

Optical microscopy has been a critical tool for studying biological systems, allowing researchers to visualize structures and processes at the cellular and sub-cellular level. However, traditional optical microscopy has a significant limitation known as the diffraction limit [12]. The diffraction limit arises from the fact that light waves diffract as they pass through an aperture; as a result, light coming from a point source is focused by a lens not to a point but to a finite volume called the point spread function (PSF). This prevents the observation of structures smaller than the wavelength of light used to illuminate the sample, since two nearby point objects will not be imaged as two separate points but as two PSFs of finite dimension which may overlap. This diffraction limit has been a significant hurdle for researchers attempting to visualize fine structures and processes at the nanoscale using an optical microscope. In biological imaging, the diffraction limit may not affect the imaging at the organ or tissue level. However, it becomes a challenge when imaging objects such as proteins (which are typically a few nanometers in size) or even cell organelles (which are typically around 0.2 micrometres) [3]. In response to these challenges, a new class of microscopy techniques, known as super-resolution microscopy, has emerged.

Super-resolution microscopy [47] refers to a group of techniques that allow researchers to break the diffraction limit and achieve resolutions that are several times higher than conventional optical microscopy. One of the most promising techniques in this field is Stochastic Optical Reconstruction Microscopy (STORM) [35, 5]. STORM is based on the principle of photo-switching of fluorophores, which are molecules that absorb light of a specific wavelength and emit fluorescence light. In STORM, the sample is labelled with fluorophores whose emission can be switched on and off stochastically in such a way that two nearby molecules do not undergo fluorescence emission simultaneously. This allows individual molecules to be localized with extremely high precision. By localizing all the individual molecules present in the sample, a super-resolution image of the same can be reconstructed.

One crucial step in a STORM-based microscope is the localization of a large number of molecules and the reconstruction of the image. Softwares such as ThunderSTORM [31] perform the localization and reconstruction operations. A separate software is first used to run the microscope and to capture images. The localization software then operates on the captured images, usually after the image acquisition is over. Since STORM requires capturing several thousand images of the sample, the above process of reconstruction post-completion of acquisition eats up a significant amount of disk space of the PC and thus poses an issue in the case of frequent use of the STORM system, for instance in a hospital or clinic. Besides, the suitability of the images for successful localization is known only after the entire acquisition process is over, especially when the microscope operator is a non-expert. In this work, we present a comprehensive yet simple Python-based computer program for localization and reconstruction in a STORM system. The program can run simultaneously with the image acquisition, thereby letting the operator know beforehand if the captured images are suitable for localization so that, if needed, another imaging attempt can be made at the earliest after incorporating the necessary modifications. The program also has the option to delete the captured images in a sequential manner once the molecular locations from each image are obtained so that minimum possible disk space is used at any instant of time. Besides, the program is written in a modular form for better understandability of general readers and has several options to obtain the reconstructed image varying from low accuracy and high speed to high accuracy and low speed. The use of open source code in Python and the simple code structure facilitate easy adaptability with different instruments, operating systems and applications. In this paper, we first present the basic concept of optical super-resolution and STORM, followed by a description of the software code and the related algorithms. We end this paper with some demonstration examples of the proposed software code.

2 Image Formation by an Optical Imaging System

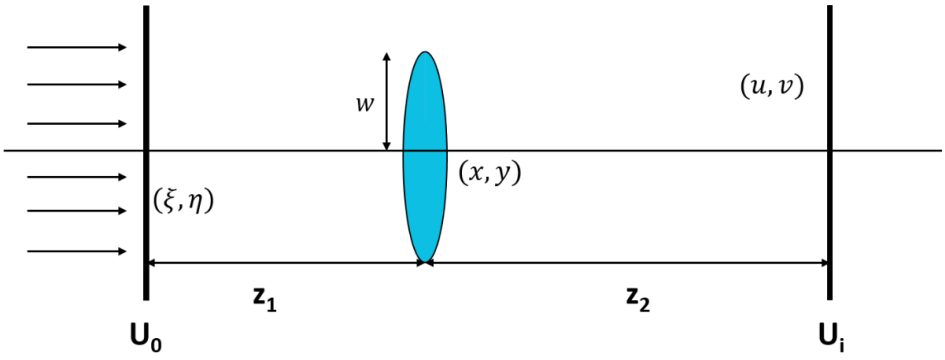

Figure 1: Schematic diagram showing image formation by a lens

To understand the concept of optical super-resolution, we first delve into the mathematical foundations of image formation by a lens depicted in Figure 1. We assume that the object to be imaged is represented by a two dimensional complex amplitude function. If $U_o(\xi, \eta)$ is the complex amplitude in the object plane where (ξ, η) are the object plane coordinates, then the complex amplitude in the image plane $U_i(u, v)$, with coordinates (u, v), can be written as [12, 45]

$$U_i(u,v) = A \iint_{\mathbb{R}^2} U_o(\xi,\eta)\; h(\frac{u}{M} - \xi, \frac{v}{M} - \eta)\, d\xi d\eta, \tag{1}$$

where A is a constant, M is the magnification and h is the impulse response of the imaging system, which is the response of the imaging system at (u, v) in the image plane for a unit amplitude point at (ξ, η) in the object plane. If $P(x, y)$ is the aperture pupil function of the lens with coordinates (x, y), (z_1, z_2) are the object and image distances, and λ is the wavelength of illumination so that $(k = \frac{2\pi}{\lambda})$, then h can be written as [12]

$$h(u,v) = \frac{1}{\lambda^2 z_1 z_2} \iint_{\mathbb{R}^2} P(x,y)\, exp\Big[\frac{-ik}{z_2}(ux + vy)\Big]\, dxdy. \tag{2}$$

For unit magnification $(M = 1)$, Equation (1) reduces to

$$U_i(u,v) = A \iint_{\mathbb{R}^2} U_o(\xi,\eta)\, h(u - \xi, v - \eta)\, d\xi d\eta, \tag{3}$$

which represents the convolution of the object plane amplitude with the impulse response function of the lens so that

$$U_i(u,v) = U_o(\xi,\eta) \otimes h(u,v). \tag{4}$$

Here $\otimes$ represents the convolution operation [7]. In a fluorescence microscope such as STORM, the image is formed by collecting the fluorescence light from the sample which is incoherent. Thus a STORM system belongs to the class of incoherent imaging systems. The intensity in the image plane for such an incoherent imaging system is given as

$$I(u,v)_{incoherent} = |U_o(\xi,\eta)|^2 \otimes |h(u,v)|^2. \tag{5}$$

The term $|h(u,v)|^2$ is known as the point spread function of the imaging system, as mentioned already. The image, thus, is quantified mathematically by the convolution operation of the object plane intensity and the PSF of the imaging system.

The pupil function $P(x,y)$ of an imaging lens is, in general, a 'circ' function. For such a lens, its PSF is the Fraunhofer diffraction pattern of the circular lens aperture, which can be written as $|h|^2$ where

$$h(r_3) = \frac{A}{z_2\lambda}\Big[\frac{2J_1(kr_3\, sin\theta)}{kr_3\, sin\theta}\Big]. \tag{6}$$

Here J_1 is the Bessel function of the first kind. The PSF for a circular pupil function is also known as the Airy pattern. Here $sin\theta = \frac{w}{z_2}$ with w as the radius of the lens aperture and $r_3 = \sqrt{(u^2+v^2)}$.

2.1 Resolution and the Diffraction Limit

The resolution of an imaging system is its ability to distinguish two closely located objects in the image plane. The minimum separation between two point objects that can be resolved in the image plane is called the resolution limit. As discussed already, an imaging lens with a circular aperture, due to diffraction, images a point object as an Airy pattern which is also the PSF of the imaging system [24]. Any object with a size smaller than the PSF will appear to be of the same size as the PSF in the image plane. Since the Airy pattern has a finite width, hence there is an inherent limit to resolve two closely placed objects. Ernst Abbe explained this fundamental constraint in optical microscopy known as the diffraction limit, in terms of the wavelength of light, λ, and the numerical aperture (NA) of the imaging lens (usually the objective lens in the microscope) as [1],

$$d_{lateral} = \frac{\lambda}{2\,NA}. \tag{7}$$

Using Equation (7), we can determine the maximum resolution (according to Abbe's limit) that can be achieved when imaging a sample with light of a given wavelength. For instance, if we use light of 500 nm to image a sample, the maximum resolution, considering NA=1, is 250 nm.

Figure 2(a) shows the grayscale image of a PSF in the form of the Airy pattern, and Figure 2(d) shows the corresponding line plot. When we consider two point objects close to one another by a distance a little larger than the resolution limit, they give rise to partially overlapping Airy patterns, as shown in Figure 2(b) and the respective line plot in Figure 2(e). However, if we bring the two points within the resolution limit, the overlapping PSFs, as indicated by Figure 2(c) and the respective plot in Figure 2(f), can not be distinguished from one another.

In addition to Abbe's resolution limit, two other criteria are also used to determine the resolution in optical systems, namely, the Rayleigh's criterion and Sparrow's criterion. According to the Rayleigh's criterion two point sources are resolvable when the central maximum (principal diffraction maximum) of the Airy pattern of the first source coincides with the first minimum of the Airy pattern from the other source [34]. The expression of Rayleigh's resolution limit is given as $d_{lateral} = \frac{0.61\,\lambda}{NA}$. On the other hand, Sparrow's criterion offers a more practical approach to resolution determination. Resolution limit according to Sparrow's criterion is defined as the distance between two point sources where their images no longer have a dip in intensity level between the central peaks [41]. The expression of Sparrow's resolution limit is given as $d_{lateral} = \frac{0.47\,\lambda}{NA}$. To present a comparison of the three resolution criteria, we consider two point sources emitting at 500 nm kept at separations equal to the Abbe's, Rayleigh's and Sparrow's resolution limits. We numerically simulate the resulting PSFs considering a focusing lens of NA=1. Figures 3 (a), (b) and (c) show the line plots of the intensity considering the PSFs separately and the resultant intensity.

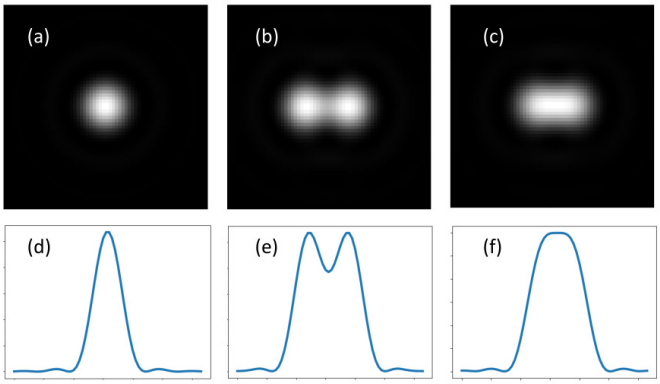

Figure 2: Images of (a) a single point object or the Airy pattern, (b) two close by points and (c) two points separated by a distance smaller than the resolution limit of the imaging system. (d), (e) and (f) show the line plots through a horizontal line passing the centre of (a), (b) and (c), respectively.

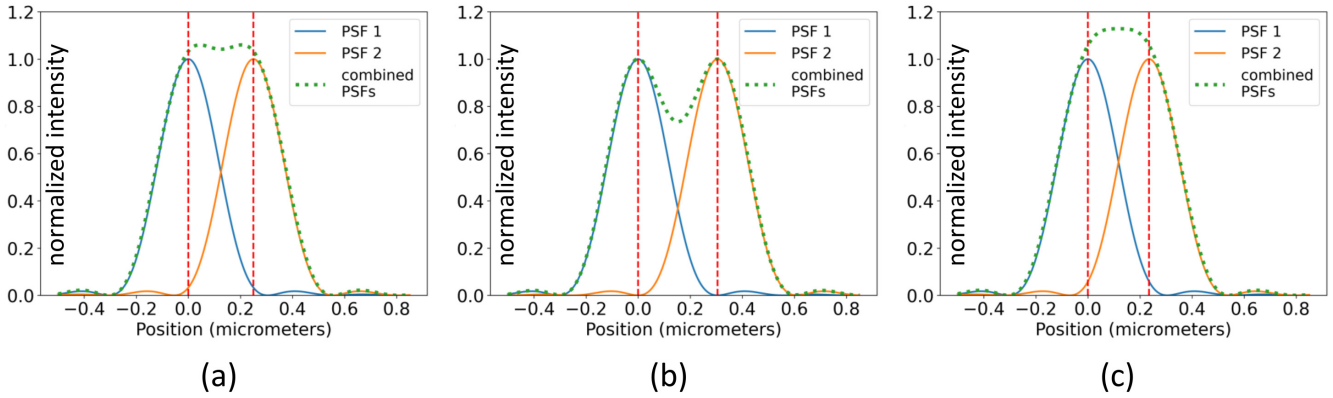

Figure 3: Line plots of the intensity considering the PSFs separately and the resultant intensity, for two points sources kept at separations equal to the resolution limit accroding to the (a) Abbe, (b) Rayleigh and (c) Sparrow criteria. The calculations are based on wavelength(λ)= 500 nm and Numerical Aperture (NA) of 1.

We notice that the numerical aperture of a microscope objective is a critical factor influencing achievable resolution by a microscope and it is defined as $NA = n\ sin(\alpha)$, where n is the refractive index of the focusing medium and α is the semi aperture angle. In principle the resolution of a microscope can be enhanced by increasing the NA of the objective lens. However α can take a maximum value of 90°, thus restricting the resolution limit of the microscope for a given refractive index of the focusing medium. Objective lenses such as oil immersion objective lenses attains an NA value of about 1.4 by using oil as the focusing medium which has refractive index of about 1.5. Considering such an objective lens the Abbe's resolution limit for wavelength=500 nm is found to be $\sim$ 179 nm.

3 Super Resolution Microscopy (SRM)

A resolution beyond the diffraction limit is known as super-resolution. While the diffraction limit is a fundamental constraint, there are ways to overcome the same in optical microscopy. A microscope with a resolution beyond the diffraction limit [21] is known as a super-resolution microscope. Super-resolution microscopy consists of several methods and techniques and can be broadly categorized into two groups. The first group contains techniques that are capable of obtaining a resolution beyond the diffraction limit, but the resolution enhancement is limited. This includes variants of Confocal Microscopy such as image scanning microscopy [40] and Structured Illumination Microscopy

(SIM) [13, 27], enhancing the resolution by a factor of ≈ 1.8 and ≈ 2.2, respectively, compared to conventional widefield microscopy. The second group primarily contains fluorescence imaging techniques capable of significantly improving resolution and can be considered 'true super resolution' techniques. This group includes Stimulated Emission Depletion (STED) Microscopy [19] [25] and Single Molecule Localization Microscopy (SMLM). The current SRM approaches can achieve a resolution of around 15-20 times greater than that of conventional optical microscopy [46, 18, 26]. In theory, that means two points separated by as close as 10 nm should be visible as separate entities with these SRM methods, and such resolutions have been reported recently in fixed samples [8].

3.1 Single Molecule Localization Microscopy (SMLM)

Single-molecule localization microscopy (SMLM) is a powerful fluorescence microscopy-based imaging technique that enables nanometer-level spatial resolution. SMLM relies on the phenomenon of fluorescence blinking, involving the random switching of fluorescent molecules between dark and bright states under continuous illumination [14]

During the process of blinking, a fluorescent molecule will switch between an excited state, where it can emit fluorescence light and a dark state, where it can not emit any light. The length of time that a molecule spends in each state can vary in a random manner which makes it possible to keep only a sparse subset of the fluorescent molecules in the sample in the emission capable (i.e. active) state. Consequently, two molecules at a separation smaller than the diffraction limit will unlikely be in the active state simultaneously, giving rise to an isolated PSF, whose centre can be identified accurately in lieu of two overlapped PSFs. Such single-molecule localization concept [11] [48] can be applied repeatedly to localize all the molecules present in the sample and thereby reconstruct a super resolved image of the same.

There are several SMLM techniques based on different sample preparation methods and types of fluorescent molecules used, such as Photoactivation Localization Microscopy (PALM) [6], fluorescent PALM (fPALM) [20], Stochastic Optical Reconstruction Microscopy (STORM) [35], direct STORM (dSTORM) [17], Points Accumulation for Imaging in Nanoscale Topography (PAINT) [39], DNA Point Accumulation for Imaging in Nanoscale Topography (DNA-PAINT) [37], MUltiple SIgnal Classification ALgorithm (MUSICAL) [2].

Despite the differences in the sample preparation and imaging procedures used, most SMLM techniques implement similar post-acquisition localization algorithms. The accuracy of such systems is given by the Cramer Rao Lower Bound [10], which provides a theoretical lower limit on the error associated with the localization process.

3.2 Stochastic Optical Reconstruction Microscopy (STORM)

In 2006, Michael J Rust, Mark Bates, and Xiaowei Zhuang developed STORM [35] as a new technique for super-resolution imaging. Unlike STED, which takes a deterministic approach, STORM uses a probabilistic approach [4] and requires a much simpler experimental setup. STORM uses photoswitchable synthetic fluorophores (dyes) that can switch between fluorescent and non-fluorescent states in the presence of specific buffers [20], using lasers of different wavelengths [6].

STORM has several advantages over other super-resolution techniques. It can achieve spatial resolution up to 10 times higher than that of conventional microscopy [35]. STORM also has a lower excitation intensity, which reduces photobleaching and phototoxicity of the molecules, making it a suitable technique for live-cell imaging [4]. Furthermore, the technique is compatible with a wide range of biological samples, including cells and tissues [20].

4 The Comprehensive Python Code for Localization and Reconstruction

As stated already, in single molecule localization microscope such as STORM, super-resolution is achieved by computationally localizing the fluorophores in images with diffraction-limited resolutions. Accurate and robust PSF localization is a challenging task that directly impacts the quality of reconstructed images, affecting both resolution and accuracy. Therefore, PSF localization is a critical step in the image reconstruction process in localization microscopy. A variety of mathematical algorithms have been developed and can be adopted for the localization of PSFs.

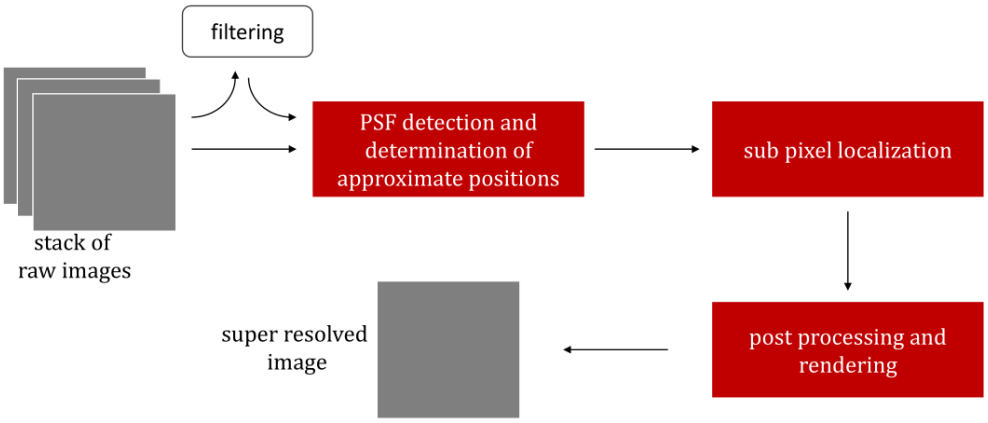

Figure 4: Localization and reconstruction steps in STORM.

Our program is written in open-source programming language Python, and apart from our own code, it utilizes common libraries such as NumPy [16], SciPy [44], scikit-image [43], and matplotlib [22]. Figure 4 depicts the various processes carried out by the program. The localization procedure in our program starts with loading the stack of images using scikit-image after the camera of the STORM system captures all the diffraction limited images (i.e. raw images captured by the camera in the microscope). However, our program also has an option to dynamically load single or multiple raw images, as and when available, immediately after the image acquisition begins. The loaded images can be but not necessarily pre-processed to enhance the contrast and reduce background noise. Each image from the camera is then operated upon by a crude localization module to obtain approximate location of each single molecule. After crude localization, the image is then acted upon by the sub-pixel localization module, which localizes the single molecules with an accuracy smaller than that provided by the camera pixels. The program then filters out localizations that do not meet the desired quality criteria. The remaining localization values are then used to reconstruct the super-resolved image of the sample. Our program has an option to delete each raw image after the respective localization values are incorporated into the reconstructed image and also to display the updated reconstructed image in real time as the localization progresses.

Below we provide a brief description of all the above steps done via a set of modular code and the associated algorithms.

4.1 Loading an Image or an Image Stack and Reconstruction Method Selection

As already stated, the proposed Python code includes two options for loading the images. The method for loading the images is determined by the variable *IMAGE_LOADING_TYPE*. Firstly,

it allows for loading a TIFF stack with dimensions (*NUMBER_OF_IMAGES, DIMENSION_1, DIMENSION_2*), where *NUMBER_OF_IMAGES* represents the total number of images in the stack, and *DIMENSION_1* and *DIMENSION_2* represent the dimensions of each image. Secondly, there is a provision for dynamically loading images when they are added to a predefined location. In the case of dynamic loading, we need to manually provide these values. Additionally, the value of the variable *IMAGE_DELETION* determines whether the already processed images should be deleted. The dynamic loading option is expected to be implemented in conjunction with the camera that directly loads the raw image (before saving it to the hard disk) for localization at a regular interval. However, for demonstration purposes, we are supplying a separate code *dynamic_im_gen.py* that mimics the dynamic loading and saving of images in a specified folder.

TIFF is the default choice for saving scientific images as apart from providing an option of lossless compression, TIFF enables z stacking of image slices for volume imaging. Additionally TIFF tags are useful for microscopy as the tags are often populated with useful information such as the microscope objective, exposure, binning etc. used during imaging.

The loading process is performed using the scikit-image library. We utilize the `io.imread()` function of scikit-image Python package to load the images for subsequent operations, with each image being saved as a NumPy array.

Next, we select the reconstruction method by using the variable *RECONSTRUCTION_TYPE*. Based on the assigned value of this variable, the program performs either crude localization alone or crude localization followed by sub-pixel localization. Furthermore, the type of crude localization and sub-pixel localization is selected through the variables *CRUDE_LOCALIZATION_TYPE* and *SUB_PIXEL_LOCALIZATION_METHOD* respectively.

4.2 Initialization of Parameters

The code first initializes all the parameters, namely, *THRESHOLD* that determines the threshold pixel value to differentiate between signal and noise, *NEIGHBOURHOOD_SIZE* that decides the size of a square window that is scanned over the image for crude localization using centre of mass method, *PSF_RADIUS* that approximately gives the radius of the airy disc of the PSF, *WINDOW_SIZE* that decides the size of a small square area in the image around the molecular location provided by the crude localization module for sub-pixel localization. The threshold levels are determined based on pixel intensity values in the experimental images. Additionally, we also initialize the input parameters for the model Gaussian function for fitting. This set of parameters denoted as *INIT_PARAMS* comprises parameters such as *amplitude* to represent the maximum value of the Gaussian function, *(xo, yo)* the initial origin of the Gaussian, σ_x, σ_y initial width of the Gaussian, θ to represent the angle of rotation of the Gaussian and *offset* to add a fixed value to the entire Gaussian profile. The parameters are placed in the order [*amplitude, xo, yo,* σ_x, σ_y θ, *offset*] in the code.

4.3 Crude Localization

For crude localization, we take one raw image at a time from the image stack or from the list of image files present in the relevant folder. In case the image pixels have larger bit values, the code has the option of converting the same to an 8-bit format to make the computation simpler. The image can also be filtered using a Gaussian filter to enhance the features for proper detection of the PSF. Our code can employ three different methods for crude localization, namely, peak local maximum, centre of mass and blob detection using Laplacian of Gaussian.

4.3.1 Peak Local Maximum

Peak local maximum refers to a point in the raw image where the intensity value is greater than that of its neighbouring pixels. Specifically, a peak local maximum is a pixel that has a higher value than all its adjacent pixels within a specified neighbourhood.

To find peak local maxima in an image (Algorithm 1), the code needs to define a neighbourhood around each pixel. This neighbourhood area is usually a square, a circle, or an area of any other shape, and its size is typically specified in pixels by the parameter *PSF_RADIUS*. Next, the code scans the image and compares the intensity values of each pixel to those of its neighbours using the parameter *threshold*. If the pixel has the highest value in its neighbourhood, then it is considered a peak local maximum. In our implementation, we are using the scikit-image's `peak_local_max()` function for the crude localization. This method works faster than the subsequent two methods but is less accurate than the other two.

Algorithm 1: Crude localization using peak local maxima

Input: {image, threshold, PSF_RADIUS}
Output: $\{\tilde{x_{cr}}, \tilde{y_{cr}}\}$; x, y coordinates of peak local maxima
method:
1. find all potential peak locations by comparing each pixel with its neighbours in a square area of size *PSF_RADIUS*
2. peak locations with pixel values higher than *threshold* are selected as significant peaks
3. peak locations that are too close to each other (smaller than the minimum Euclidean distance), controlled by the *PSF_RADIUS* parameter, are removed.
4. return the coordinates $\{\tilde{x_{cr}}, \tilde{y_{cr}}\}$ of the remaining peaks.

Figure 5 (a) shows a PSF array comprising a maximum surrounded by pixels of values lower than the maximum, distributed in a symmetric manner, while Figure 5(b) shows an image array in the presence of noise. Figure 5(c) depicts how thresholding can eliminate the effect of noise in a PSF array.

4.3.2 Centre of Mass Method

This localization is based on the Centre of Mass (COM) algorithm (Algorithm 2) to find the centroid of the PSFs. The centroid of a shape is the arithmetic mean of all the points in the shape. The centroid of an image is given as

$$[x_{centre}, y_{centre}] = \left[\frac{\sum x\, I(x,y)}{\sum I(x,y)}, \frac{\sum y\, I(x,y)}{\sum I(x,y)}\right] \tag{8}$$

where $I(x,y)$ is the intensity of the pixel (x,y). In this module, the program first finds the image array's maxima and minima. It then retains the maxima if the difference between the maxima and the minima is greater than the *threshold*. Afterwards, the centroid is computed using Equation (8) in a square area defined by *neighbourhood_size* about each maximum. To implement COM, `scipy.ndimage.cente_of_mass()` is used.

4.3.3 Blob Detection using Laplacian of Gaussian (LoG)

The Laplacian of Gaussian (LoG) filter is an image enhancement filter used to highlight areas of an image that contain rapid changes in intensity (such as the edges of objects or boundaries between

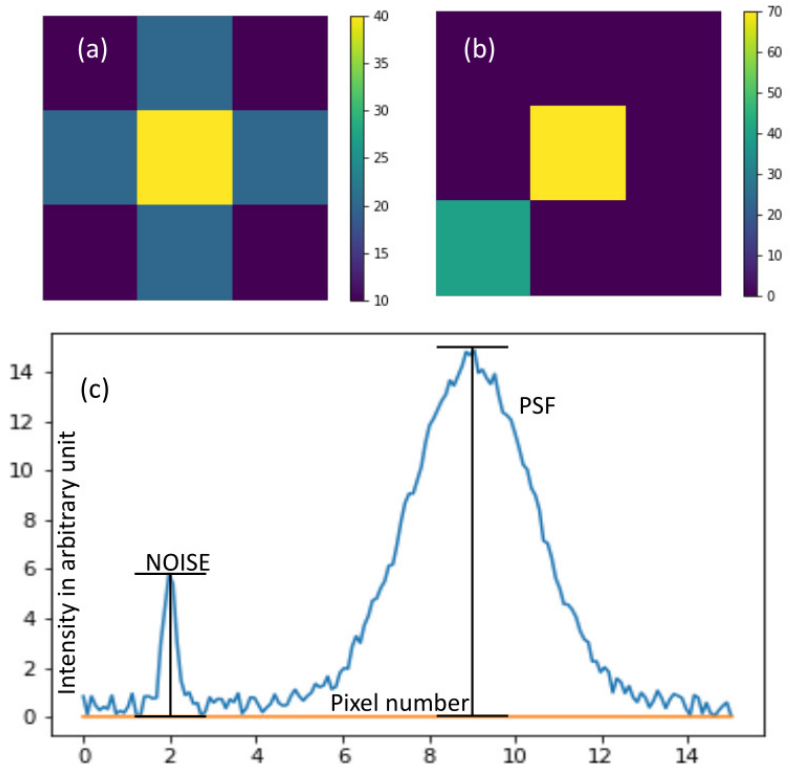

Figure 5: (a) A PSF array with symmetrically decreasing pixel values with respect to the central maximum, (b) An image array with noise showing the asymmetric distribution of pixel values about the maximum, (c) thresholding to eliminate noise: a threshold value of 10 units retains the relevant signal of the PSF while eliminating the noise which has a maximum value of 5 units.

Algorithm 2: Crude localization using centre of mass (COM)

Input: {image, threshold, neighborhood_size}
Output: $\{\tilde{x_{cr}}, \tilde{y_{cr}}\}$; x, y coordinates of centre of mass
1. Apply maximum filter on the *image* to find the local maxima
2. Identify locations of local maxima
3. Apply minimum filter on the *image* to find local minima
4. Compute the difference between maxima and minima
5. Set maxima to zero if the difference is below *threshold*
6. Define a square area of size *neighborhood_size* around each non zero maximum
7. Compute the centre of mass of each maximum over the square area using `scipy.ndimage.center_of_mas()`
8. Return the coordinates of the maxima as $\{\tilde{x_{cr}}, \tilde{y_{cr}}\}$

different regions of an image). The filter is obtained by convolving the raw image with a Gaussian filter followed by the Laplacian operator. The Gaussian filter serves as a smoothening function, while the Laplacian operator highlights regions of rapid intensity change. The operation can be represented mathematically as

$$LoG(x, y) = \nabla^2 \left(G(x, y) \otimes I(x, y) \right), \tag{9}$$

where ∇^2 is the Laplacian operator, $G(x, y)$ is the Gaussian function, and $I(x, y)$ is the input image. Building upon the LoG concept, in the blob detection algorithm (Algorithm 3) we have employed the `blob_log()` function of scikit-image. It employs a sequence of LoG filters with varying sigma values to create a 3D image cube. By identifying local maxima within this cube and subsequently

Algorithm 3: Crude localization using LoG

Input: {image, threshold, PSF_RADIUS}
Output: $\{\tilde{x_{cr}}, \tilde{y_{cr}}\}$; x, y coordinates of LoG

1. apply Laplacian of Gaussian (LoG) filter on the *image*
2. identify the local maxima in the LoG-filtered image corresponding to the potential blob centres
3. apply a threshold on the blob intensities to filter out weak or insignificant blobs using the parameter *threshold*
4. determine the scale (or size) of each detected blob by analyzing the response of the LoG filter at each local maximum; discard blobs smaller than 2 pixels or greater than *PSF_RADIUS*
5. Return the coordinates of the remaining blobs as $\{\tilde{x_{cr}}, \tilde{y_{cr}}\}$ along with the corresponding sizes of the blobs

removing overlapping blobs, the function provides a comprehensive approach to detecting circular blobs within images. A threshold is then applied to eliminate weak blob candidates, retaining only the most significant ones.

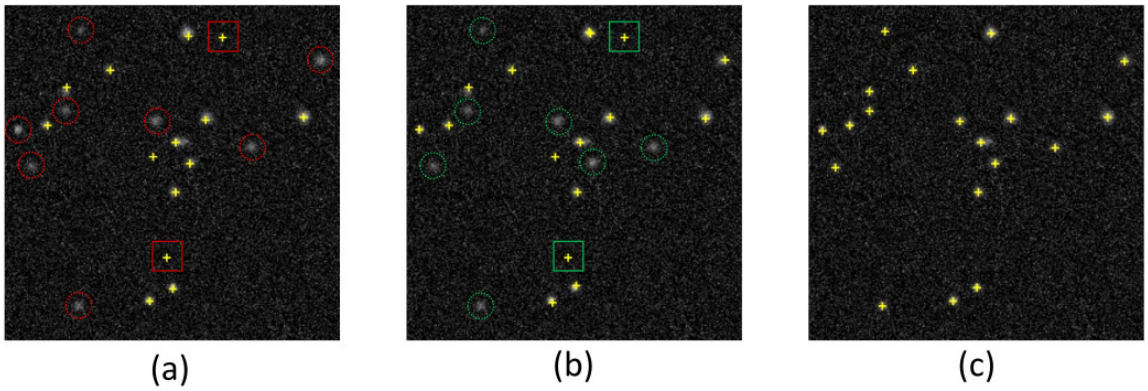

Figure 6: Crude localization of PSF using (a) peak local maximum, (b) centre of mass and (c) Laplacian of Gaussian methods on a simulated noisy image

Figures 6 (a), (b), and (c) show the positions of the PSF centres indicated by the yellow crosses ('+'), obtained using the peak local maximum, centre of mass, and Laplacian of Gaussian methods, respectively. Upon closer examination of the PSF centres, one can observe minor discrepancies in the crude localizations provided by these three methods. Some PSFs remain undetected by both the peak local maximum and centre of mass methods (shown as dotted red and green circles, respectively), while at certain locations noise is incorrectly identified as potential PSFs (shown as red and green squares, respectively). In contrast, the Laplacian of Gaussian method accurately identifies all the PSFs without including noise as potential PSFs. We see from the above results that the Laplacian of Gaussian method provides the best accuracy for crude localization.

4.4 Sub-pixel Localization

Sub-pixel localization allows for precise positioning of the PSFs corresponding to a set of sparsely located fluorescent molecules. Here the molecular position can be estimated with precision, potentially up to a fraction of a camera pixel. Sub-pixel localization achieves the localization precision beyond the pixel size of the camera by fitting the observed PSF of the fluorescent molecule to a

model PSF that takes into account the effects of diffraction and optical aberrations present in the imaging system.

There are various algorithms that can be used for sub-pixel localization, such as least square fitting and maximum likelihood estimation. These algorithms use different mathematical models and optimization techniques to fit a theoretical profile to the observed image and thereby estimate the position of the molecule.

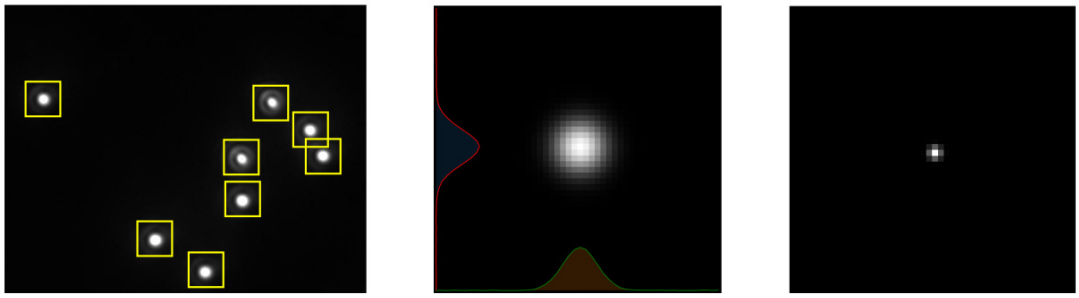

Figure 7: (a) PSFs present in an image frame, (b) an extracted PSF sub-array using crude localization, (c) sub-pixel localization of the extracted PSF using Gaussian fitting.

The process begins with extracting each PSF as an array subset from the raw image based on its approximate position as given by the crude localization module as shown in Figure 7(a). If $(\tilde{x_{cr}}, \tilde{y_{cr}})$ are the approximate coordinates and *window_size* is the PSF fitting radius, then the fitting region is defined as

$$D = \{(x,y); x\epsilon\{\tilde{x_{cr}} - window\_size, \ldots, \tilde{x_{cr}} + window\_size\}, \\ y\epsilon\{\tilde{y_{cr}} - window\_size, \ldots, \tilde{y_{cr}} + window\_size\}\}$$

Each array subset, such as in Figure 7(b), is then fitted with a model PSF to get the exact location of the centre of the respective PSF, as seen in Figure 7(c).

There are various choices for modelling a PSF. As we have discussed already, the diffraction theory predicts the Airy pattern as the PSF. However, calculation involving the Airy pattern is very computationally intensive. Besides, the pixel data of the camera corresponding to a PSF actually deviates from the true nature of the Airy pattern due to the presence of various noises associated with the imaging process as well as due to the finite pixel size of the imaging devices. It has been found that a two-dimensional Gaussian function can be used to model such a PSF very accurately [49] and is also convenient to work with a Gaussian computationally [42], [15]. A Gaussian PSF model can be defined as

$$F(x,y,A,xo,yo,\sigma_x,\sigma_y,b) = A\,exp\Big(-\frac{(x-xo)^2}{2\sigma_x^2} - \frac{(y-yo)^2}{2\sigma_y^2}\Big) + b, \tag{10}$$

where A is the peak intensity, (xo, yo) is the origin of the Gaussian, (σ_x, σ_y) are the spread of the Gaussian in x and y direction and b is the fixed background pixel value.

4.4.1 Least Square Fitting

Least squares fitting is a powerful technique that can be used to estimate the parameters of the model PSF that fits the observed data. It minimizes the sum of the squared residuals between the observed PSF and the model PSF by varying the model's parameters θ, where $\theta = (A, xo, yo, \sigma_x, \sigma_y)$.

The residue that is to be minimized is given as:

$$S = \sum_{all\ pixels} [I(x, y) - F(x, y, \theta)]^2. \tag{11}$$

The parameters $A, xo, yo, \sigma_x, \sigma_y$ that best fit the data points are computed in an iterative manner [29].

There are various algorithms that can be used to perform 2D Gaussian least square fitting, such as the Levenberg-Marquardt [28] algorithm, the Nelder-Mead simplex algorithm [30], and the gradient descent algorithm [9]. These algorithms differ in their speed and accuracy, and the choice of algorithm depends on the specific application and the properties of the data. In our program (Algorithm 4), we have utilized the `curve_fit()` function of SciPy to fit the Gaussian model with the pixel data.

1D Least Square Fitting: In 1D least square fitting the 2D optimization is reduced to two 1D optimization problems. First, the crude location of the PSF as given by the previous module is obtained, and a row of the PSF subset passing through the crude centre is extracted from the raw image frame. The extracted row is then fitted with the 1D model PSF to get the centre xo, and the width of the Gaussian σ_x along x. The module then extracts the column at xo from the same PSF subset and is fitted again with the model PSF to find its amplitude A, centre yo and width σ_y, along the y-axis.

2D least squares fitting: In this method, the extracted PSF subset for each crude location is directly fed to the module, which is based on fitting a 2D Gaussian on the 2D pixel data. Computationally, this is done by reshaping the 2D arrays of the PSF subset as well as the model Gaussian PSF into 1D arrays and then fitting them to extract the required parameters $(amplitude, xo, yo, \sigma_x, \sigma_y)$.

Algorithm 4: Sub-pixel localization using 1D/2D least squares fitting

Input: $\{(\tilde{x_{cr}}, \tilde{y_{cr}})$, image, window_size, init_params$\}$

Output: $\{amplitude, x_{sp}, y_{sp}, \sigma_x, \sigma_y\}$

1. For each PSF in the raw image, extract a sub-image of dimensions *window_size* around the approximate PSF centre given by $coordinates(\tilde{x_{cr}}, \tilde{y_{cr}})$, which is obtained by any of the three crude localization methods, determined by the parameter *method*.
2. use `scipy.optimize.curve_fit()` to fit a 1D or 2D Gaussian (based on the fitting method chosen) to the sub-image using the initial parameters *initial_guess*.
3. return the output of the optimization

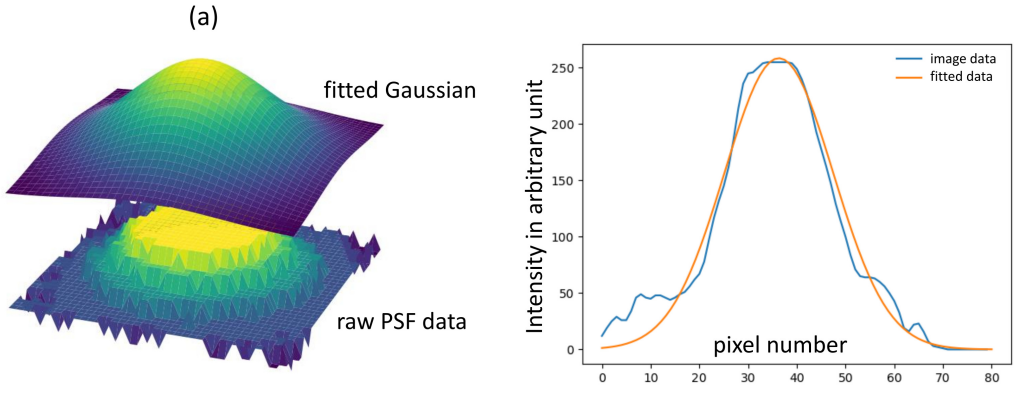

Figure 8: Surface plot of the (a) raw image data and the fitted Gaussian. (b) Line plot through the raw image and the fitted Gaussian.

Figure 8(a) shows the surface plot of the raw image data and the fitted 2D Gaussian, while Figure 8(b) shows line plots through the centres of the raw image and the Gaussian.

4.4.2 Maximum Likelihood Estimation

Maximum likelihood estimation (MLE) [23, 32] is a more sophisticated method that takes into account the statistical properties of the noise in the observed image. MLE models the PSF as a probability distribution and estimates the position of the molecule that maximizes the likelihood of how well the model fits the data for a given PSF model. This method is the most accurate but is complex and appears to be sensitive to the choice of the PSF model.

In our case, the aim is to compute the parameters θ for which the likelihood of model PSF fitting the raw image data is the highest. The likelihood function is given as

$$L(I;\theta) = \frac{1}{2\pi\sigma_x^2\sigma_y^2}\, exp\left[-\frac{(I-F(\theta))^2}{2\sigma_x^2\sigma_y^2}\right], \tag{12}$$

$$LL(I;\theta) = -\frac{1}{2}\sum_{i=1}^{n} log(2\pi\sigma_x^2\sigma_y^2) - \sum_{i=1}^{n}\frac{(I_i - F(\theta_i))^2}{2\sigma_x^2\sigma_y^2}, \tag{13}$$

where I represents the observed image data, $F(\theta)$ represents the Gaussian PSF model, and $\theta = (A, xo, yo, , \sigma_x, \sigma_y)$ represents the PSF model parameters.

Computationally, the aim is to maximize the likelihood function $L(I;\theta)$ or minimize the negative of the natural log of $L(I;\theta)$, known as the log-likelihood function $LL(I;\theta)$. This will give the optimum parameters θ. The sum in the log-likelihood function is over all the observed pixels.

The minimization problem can be represented as

$$parameters = argmin(\theta)\left[-\sum_{all\ pixels}[I(x,y) - log(F(x,y,\theta)) - F(x,y,\theta)]\right]. \tag{14}$$

In our implementation (Algorithm 5), we use the `minimize()` function of SciPy with the 'Nelder-Mead' [33] method as the optimization algorithm to minimize the negative log-likelihood (LL) and obtain the maximum likelihood estimates of the parameters.

Algorithm 5: PSF localization using MLE

Input: { $(\tilde{x_{cr}}, \tilde{y_{cr}})$, image, window_size, init_params}

Output: $\{amplitude, x_{sp}, y_{sp}, \sigma_x, \sigma_y\}$

1. For each PSF in the raw image, extract a sub-image of dimensions *window_size* around the approximate PSF centre given by *coordinates*$(\tilde{x_{cr}}, \tilde{y_{cr}})$, which is obtained by any of the three crude localization methods, determined by the parameter *method*.
2. evaluate the log-likelihood of the PSF data in the sub-image with the Gaussian model
3. use `scipy.optimize.minimize()` to minimize the negative of the log-likelihood using the initial_guess.
4. Return the output

It is to be noted that the use of both crude localization and sub-pixel localization modules provides the best accuracy at the cost of computational time. However, one can skip the sub-pixel localization stage and use the results of crude localization to reconstruct the final image, which will be much faster to process, at the same rate or a higher rate compared to image acquisition, at the cost of localization accuracy.

4.5 Image Reconstruction and Visualization

The output of the sub-pixel localization module is a list of coordinates (and if required the corresponding standard deviation of the fitted Gaussian) of each PSF present in each raw image. If the camera pixel array is indexed using two sets of whole numbers, the position of the PSF obtained is usually up to a few decimal places. This list is then used to create a super-resolved image using various visualization methods. Visualization is a crucial step in the analysis of super-resolution microscopy data, as it enables us to gain insight into the spatial distribution of molecules in the sample. In our code, we incorporate three common visualization methods that are widely used in super-resolution microscopy, namely, scatter plot, histogram, and averaged shifted histogram.

4.5.1 Removal of Incorrect Localizations

There are occasional, especially due to the presence of noise in the raw image, the sub-pixel localization module may return incorrect coordinates or PSF centres. Therefore, the code operates another module to check if all the coordinates returned by the sub-pixel localization modules fall within the boundary of the raw image or not. It retains only those coordinates which satisfy the above check and discards the rest.

4.5.2 Scatter Plot

A scatter plot is a simple visualization method that shows the distribution of PSFs in a 2D space. Each localization is represented as a point in the scatter plot, with the x and y coordinates corresponding to the position of the molecule in the image. Our code for the scatter plot creates a binary image where each pixel corresponding to a localization is assigned a value equal to 1 while the other pixels have values equal to zero. The code updates the scatter plot by incorporating the localization information for all the raw images. It thus creates a binary image of the localized data.

4.5.3 Histogram

Histogram is another common visualization method that can be used to visualize the distribution of the x and y coordinates of the localized molecules in the sample. Our implementation of the method creates a 2D histogram of the localization data using `histogram2d()` of NumPy. If there are multiple localization events at a given pixel, it is assigned a value equal to the number of events, Thus unlike the scatter plot, the histogram creates a grayscale image of the molecular positions.

4.5.4 Averaged Shifted Histograms

Averaged shifted histogram (ASH) [38] is a non-parametric density estimator that was introduced as an effective method for estimating probability density functions in multiple dimensions. The ASH algorithm works by computing a set of shifted histograms, which are histograms that have been shifted by a small amount in each direction, and then averaging them together to produce a final estimate of the probability density function.

To use ASH for visualizing the localized data, we first start with the set of 2D localization coordinates. We then create a 2D histogram of the localization coordinates, where each bin in the histogram contains the number of localizations that fall within that bin. Next, the program translates the original histogram (using `numpy.roll()`) in each direction by a small amount (e.g., one bin or one pixel) and then recomputes the histogram. This is repeated for a range of shift values in each direction, resulting in a set of histograms that have been shifted in all possible directions by a small amount.

Finally, the program averages all of the shifted histograms to produce the final ASH estimate of the probability density function. The result is a smooth estimate of the density of localizations across the field of view that is less sensitive to the choice of bin size and bin centering than a traditional histogram.

Figure 9 shows a comparison of the three visualization methods incorporated in our code.

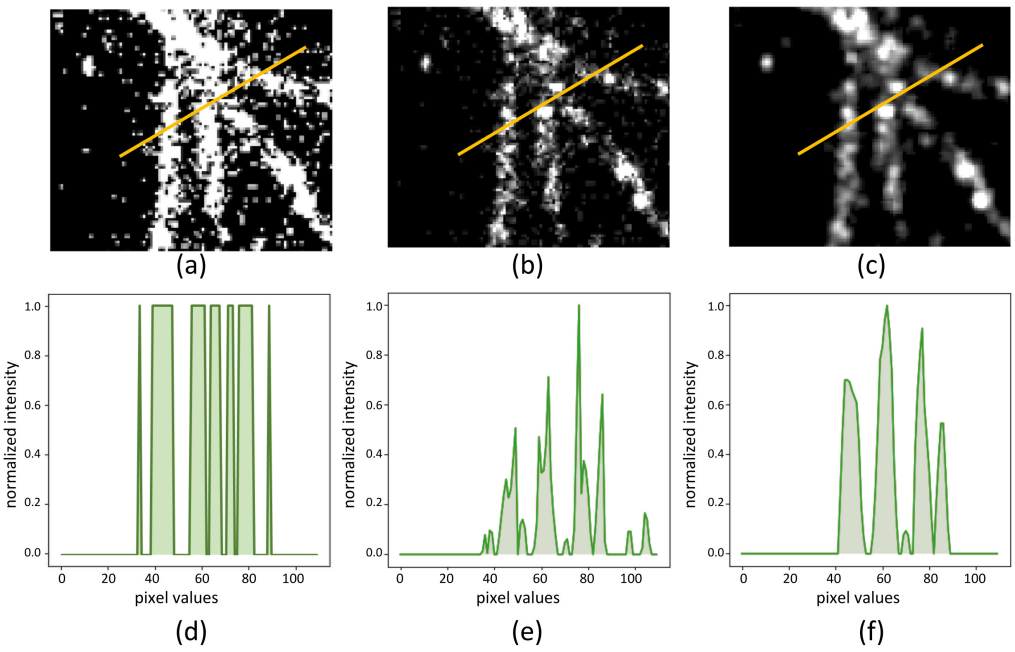

Figure 9: Visualization of the reconstructed image using (a) scatter plot, (b) 2D histogram, (c) averaged shifted histogram. (d) , (e) and (f) are line plots along the line shown through (a), (b) and (c), respectively.

5 Implementation of the Proposed Python Code and Results

We implement our code to localize PSFs present in raw images and reconstruct the super-resolved image using both simulated and experimental image data.

Figure 10 shows reconstruction from a simulated stack of images. The original image stack comprises 2000 raw images generated from the source image or object depicting the word 'HELLO'. In each image of the stack, only one randomly selected pixel is activated which contains the correct pixel value relative to the source image. The active pixel is then convolved with an Airy pattern as shown in Figure 2(a) to mimic the raw image of a sparse set of active fluorescent molecules, albeit with one active molecule per image frame in this case. The complete stack is then processed using the 1D least square fit method, and the output is shown as a scatter plot of the localization coordinates in Figure 10(c). The same stack is then processed with the popular localization and reconstruction software ThunderSTORM [31]. In ThunderSTORM, we used the local maximum for crude localization and then applied the 2D least square fitting algorithm for sub-pixel localization. The super-resolved image as a scatter plot as given by ThunderSTORM is shown in Figure 10(d).

We then consider raw images obtained from the SRM-hub website `https://srm.epfl.ch/DatasetPage?name=MT4.N2.HD` [36]. The image stack has 3200 frames of microtubule structures each containing PSF data of high density of fluorophores, over dimensions 64×64 pixels.

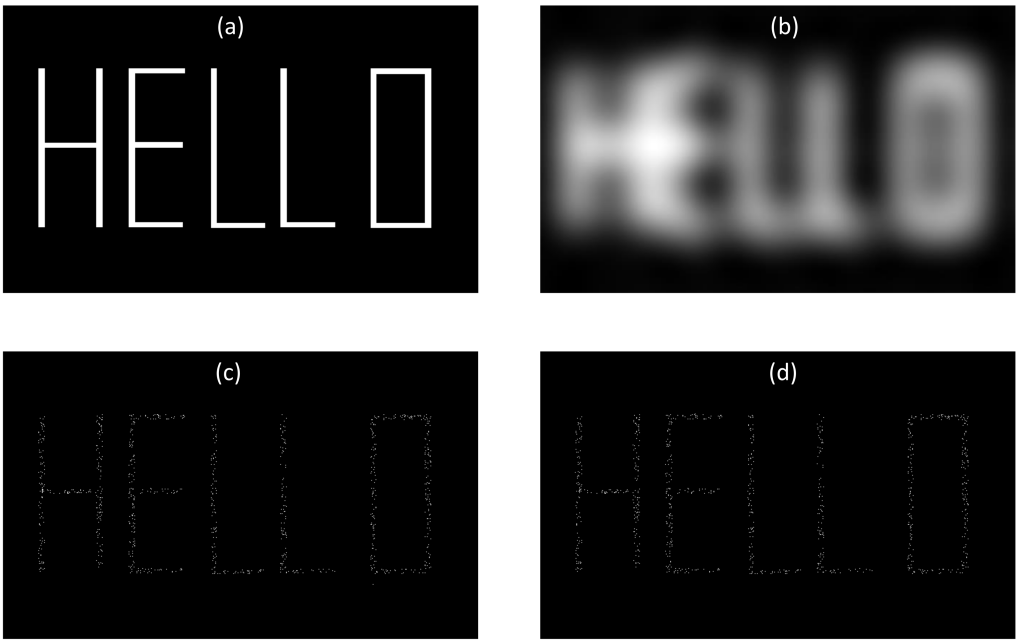

Figure 10: (a) Numerically constructed object or source image showing the word 'HELLO', (b) simulated image of the object by a conventional microscope, (c) reconstruction using the proposed algorithm, (d) reconstruction using ThunderSTORM.

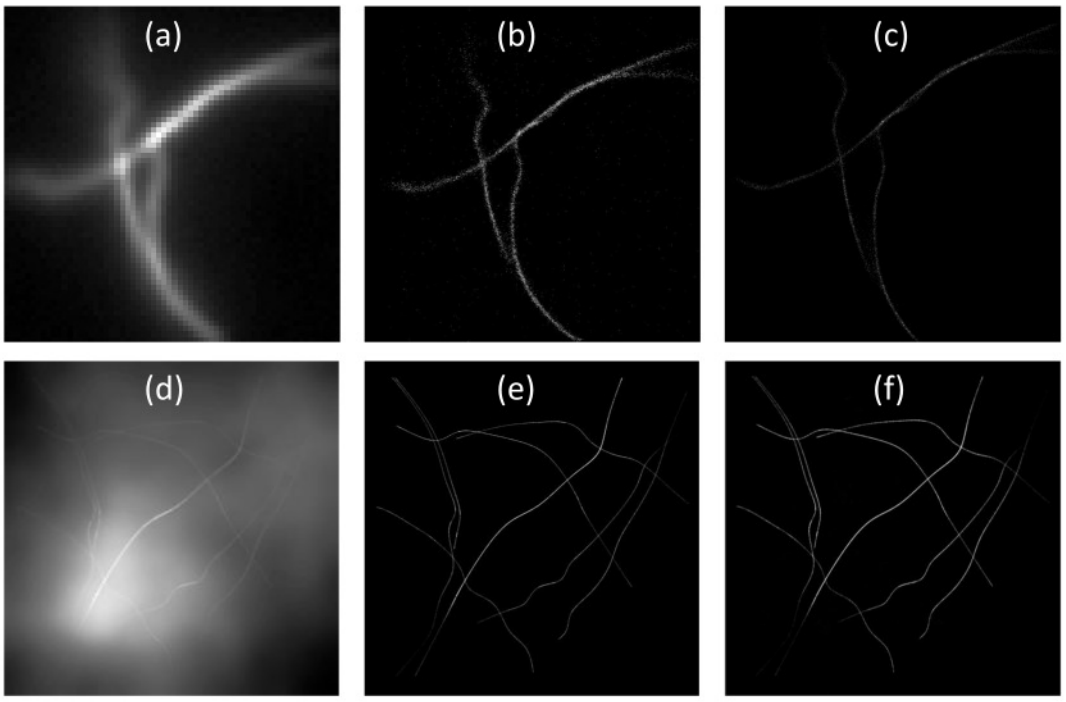

Figure 11: (a) and (d) are widefield images of tubulin, (b) and (e) are reconstructions using the proposed code, while (c) and (f) are reconstructions using ThunderSTORM.

Figure 11(a) shows a widefield image of the sample plane. We then employ Laplacian of Gaussian and 2D least square fitting of our code for crude localization and sub-pixel localization, respectively. The super-resolved image using the visualization method Averaged Shifted Histograms returned by our code is seen in Figure 11(c). The same raw image stack is then processed with ThunderSTORM using B-spline filtering and local maximum for crude localization and MLE for sub-pixel localization. The super-resolved image again as Averaged Shifted Histograms given by ThunderSTORM is seen

in Figure 11(b).

We then consider raw image data obtained from https://srm.epfl.ch/DatasetPage?name=Tubulins_I comprising an image stack of 2400 frames each of dimensions 256×256 pixels. Each raw image corresponds to Tubulin structures containing high density fluorophores. Figure 11(d) shows a widefield image of the sample while Figures 11(f) and (e) show the respective super-resolved images in the form of Averaged Shifted Histograms as given by ThunderSTORM and our proposed code, respectively.

A comparison of Figures 10(c) and (d), Figures 11(b) and (c) and Figures 11(e) and (f) indicates that the proposed Python code produces super-resolved images of quality comparable with that of ThunderSTORM.

6 Conclusion

In this paper, we have proposed a comprehensive Python based software code for localization and reconstruction of super-resolved image in an imaging system using the single molecule localization technique. Here we consider STORM as one such super-resolution imaging technique where our code can be applied, although our code will be equally applicable to any other similar imaging techniques using single molecule localization. The program is written in a modular form and follows a simple structure for easy understandability of general readers and adaptability for different applications. The code can be executed post image acquisition or simultaneously with image acquisition. The latter option can facilitate aborting the image acquisition using the localization output given by our code as the feedback in case raw images are not suitable for localization, so that the corrective measures can be employed at the earliest. Besides, this option also facilitates the use of minimal disk space which can be beneficial in case of frequent use of STORM or other similar systems which otherwise will eat up significant disk space. The proposed software code is implemented in a number of demonstration examples indicating that the performance of our code is very much comparable in terms of quality of the super-resolved image with that of ThunderSTORM. Our code offers flexibility to obtain the reconstructed image quickly by using the crude localization module only, so that image acquisition and reconstruction can proceed at the same rate.

Acknowledgment

The author PC acknowledges the support of the Council of Scientific and Industrial Research (CSIR), New Delhi, for the research.

Image Credits

The raw images in Figure 11 are from the SRM-hub website https://srm.epfl.ch/DatasetPage?name=MT4.N2.HD and from https://srm.epfl.ch/DatasetPage?name=Tubulins_I.

References

[1] E. Abbe, *Beiträge Zur Theorie des Mikroskops Und Der Mikroskopischen Wahrnehmung*, Archiv Für Mikroskopische Anatomie, 9 (1873), pp. 413–468, https://doi.org/10.1007/BF02956173.

[2] K. Agarwal and R. Macháň, *Multiple Signal Classification Algorithm for Super-Resolution Fluorescence Microscopy*, Nature Communications, 7 (2016), p. 13752, https://doi.org/10.1038/ncomms13752.

[3] B. Alberts, D. Bray, K. Hopkin, A. D. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, *Essential Cell Biology*, Garland Science, 2015.

[4] M. Bates, B. Huang, G. T. Dempsey, and X. Zhuang, *Multicolor Super-Resolution Imaging with Photo-Switchable Fluorescent Probes*, Science, 317 (2007), pp. 1749–1753, https://doi.org/10.1126/science.1146598.

[5] M. Bates, S. A. Jones, and X. Zhuang, *Stochastic Optical Reconstruction Microscopy (STORM): a Method for Superresolution Fluorescence Imaging*, Cold Spring Harbor Protocols, 2013 (2013), pp. pdb–top075143, https://doi.org/10.1101/pdb.top075143.

[6] E. Betzig, G. H. Patterson, R. Sougrat, O. W. Lindwasser, S. Olenych, J. S. Bonifacino, M. W. Davidson, J. Lippincott-Schwartz, and H. F. Hess, *Imaging Intracellular Fluorescent Proteins at Nanometer Resolution*, Science, 313 (2006), pp. 1642–1645, https://doi.org/10.1126/science.1127344.

[7] R. N. Bracewell and P. B. Kahn, *The Fourier Transform and its Applications*, vol. 31999, McGraw-Hill New York, 1986, https://doi.org/10.1119/1.1973431.

[8] S. Coelho, J. Baek, M. S. Graus, J. M. Halstead, P. R. Nicovich, K. Feher, H. Gandhi, J. J. Gooding, and K. Gaus, *Ultraprecise Single-Molecule Localization Microscopy Enables in Situ Distance Measurements in Intact Cells*, Science Advances, 6 (2020), p. eaay8271, https://doi.org/10.1126/sciadv.aay8271.

[9] H. B. Curry, *The Method of Steepest Descent for Non-Linear Minimization Problems*, Quarterly of Applied Mathematics, 2 (1944), pp. 258–261, https://doi.org/10.1090/qam/10667.

[10] J. E. Fitzgerald, J. Lu, and M. J. Schnitzer, *Estimation Theoretic Measure of Resolution for Stochastic Localization Microscopy*, Physical Review Letters, 109 (2012), p. 048102, https://doi.org/10.1103/PhysRevLett.109.048102.

[11] J. Gelles, B. J. Schnapp, and M. P. Sheetz, *Tracking Kinesin-Driven Movements with Nanometre-Scale Precision*, Nature, 331 (1988), pp. 450–453, https://doi.org/10.1038/331450a0.

[12] J. W. Goodman, *Introduction to Fourier Optics*, Roberts and Company Publishers, 2005, https://doi.org/10.1063/1.3035549.

[13] M. G. Gustafsson, D. Agard, J. Sedat, et al., *I5M: 3D Widefield Light Microscopy with Better Than 100nm Axial Resolution*, Journal of Microscopy, 195 (1999), pp. 10–16.

[14] T. Ha and P. Tinnefeld, *Photophysics of Fluorescent Probes for Single-Molecule Biophysics and Super-Resolution Imaging*, Annual Review of Physical Chemistry, 63 (2012), pp. 595–617, https://doi.org/10.1146/annurev-physchem-032210-103340.

[15] O. Haeberlé, M. Ammar, H. Furukawa, K. Tenjimbayashi, and P. Török, *Point Spread Function of Optical Microscopes Imaging Through Stratified Media*, Optics Express, 11 (2003), pp. 2964–2969, https://doi.org/10.1364/OE.11.002964.

[16] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, *Array Programming with NumPy*, Nature, 585 (2020), pp. 357–362, https://doi.org/10.1038/s41586-020-2649-2.

[17] M. Heilemann, S. Van De Linde, M. Schüttpelz, R. Kasper, B. Seefeldt, A. Mukherjee, P. Tinnefeld, and M. Sauer, *Subdiffraction-Resolution Fluorescence Imaging with Conventional Fluorescent Probes*, Angewandte Chemie International Edition, 47 (2008), pp. 6172–6176, https://doi.org/10.1002/anie.200802376.

[18] S. W. Hell, S. J. Sahl, M. Bates, X. Zhuang, R. Heintzmann, M. J. Booth, J. Bewersdorf, G. Shtengel, H. Hess, P. Tinnefeld, et al., *The 2015 Super-Resolution Microscopy Roadmap*, Journal of Physics D: Applied Physics, 48 (2015), p. 443001, https://doi.org/10.1088/0022-3727/48/44/443001.

[19] S. W. Hell and J. Wichmann, *Breaking the Diffraction Resolution Limit by Stimulated Emission: Stimulated-Emission-Depletion Fluorescence Microscopy*, Opt. Lett., 19 (1994), pp. 780–782, https://doi.org/10.1364/OL.19.000780.

[20] S. T. Hess, T. P. Girirajan, and M. D. Mason, *Ultra-High Resolution Imaging by Fluorescence Photoactivation Localization Microscopy*, Biophysical Journal, 91 (2006), pp. 4258–4272, https://doi.org/10.1529/biophysj.106.091116.

[21] B. Huang, H. Babcock, and X. Zhuang, *Breaking the Diffraction Barrier: Super-Resolution Imaging of Cells*, Cell, 143 (2010), pp. 1047–1058, https://doi.org/10.1016/j.cell.2010.12.002.

[22] J. D. Hunter, *Matplotlib: A 2D Graphics Environment*, Computing in Science & Engineering, 9 (2007), pp. 90–95, https://doi.org/10.1109/MCSE.2007.55.

[23] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice-Hall, Inc., 1993.

[24] J. B. Keller, *Diffraction by an Aperture*, Journal of Applied Physics, 28 (1957), pp. 426–444, https://doi.org/10.1063/1.1722767.

[25] T. A. Klar and S. W. Hell, *Subdiffraction Resolution in Far-Field Fluorescence Microscopy*, Opt. Lett., 24 (1999), pp. 954–956, https://doi.org/10.1364/OL.24.000954.

[26] M. Lelek, M. T. Gyparaki, G. Beliu, F. Schueder, J. Griffié, S. Manley, R. Jungmann, M. Sauer, M. Lakadamyali, and C. Zimmer, *Single-Molecule Localization Microscopy*, Nature Reviews Methods Primers, 1 (2021), p. 39, https://doi.org/10.1038/s43586-021-00038-x.

[27] D. Li, L. Shao, B.-C. Chen, X. Zhang, M. Zhang, B. Moses, D. E. Milkie, J. R. Beach, J. A. Hammer III, M. Pasham, et al., *Extended-Resolution Structured Illumination Imaging of Endocytic and Cytoskeletal Dynamics*, Science, 349 (2015), p. aab3500, https://doi.org/10.1126/science.aab3500.

[28] J. J. Moré, *The Levenberg-Marquardt Algorithm: Implementation and Theory*, in Numerical Analysis, Springer, 1978, pp. 105–116, https://doi.org/10.1007/BFb0067700.

[29] K. I. Mortensen, L. S. Churchman, J. A. Spudich, and H. Flyvbjerg, *Optimized Localization Analysis for Single-Molecule Tracking and Super-Resolution Microscopy*, Nature Methods, 7 (2010), pp. 377–381, https://doi.org/10.1038/nmeth.1447.

[30] J. A. Nelder and R. Mead, *A Simplex Method for Function Minimization*, The Computer Journal, 7 (1965), pp. 308–313, https://doi.org/10.1093/comjnl/7.4.308.

[31] M. Ovesnỳ, P. Křížek, J. Borkovec, Z. Švindrych, and G. M. Hagen, *ThunderSTORM: a Comprehensive ImageJ Plug-In for PALM and STORM Data Analysis and Super-Resolution Imaging*, Bioinformatics, 30 (2014), pp. 2389–2390, https://doi.org/10.1093/bioinformatics/btu202.

[32] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, Tata McGraw-Hill Education, 2002, https://doi.org/10.1063/1.3034123.

[33] M. J. Powell, *On Search Directions for Minimization Algorithms*, Mathematical Programming, 4 (1973), pp. 193–201, https://doi.org/10.1007/BF01584660.

[34] Rayleigh, *XXXI. Investigations in Optics, with Special Reference to the Spectroscope*, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 8 (1879), pp. 261–274, https://doi.org/10.1080/14786447908639684.

[35] M. J. Rust, M. Bates, and X. Zhuang, *Sub-Diffraction-Limit Imaging by Stochastic Optical Reconstruction Microscopy (STORM)*, Nature Methods, 3 (2006), pp. 793–796, https://doi.org/10.1038/nmeth929.

[36] D. Sage, H. Kirshner, T. Pengo, N. Stuurman, J. Min, S. Manley, and M. Unser, *Quantitative Evaluation of Software Packages for Single-Molecule Localization Microscopy*, Nature Methods, 12 (2015), pp. 717–724.

[37] J. Schnitzbauer, M. T. Strauss, T. Schlichthaerle, F. Schueder, and R. Jungmann, *Super-Resolution Microscopy with DNA-PAINT*, Nature Protocols, 12 (2017), pp. 1198–1228, https://doi.org/10.1038/nprot.2017.024.

[38] D. W. Scott, *Averaged Shifted Histograms: Effective Nonparametric Density Estimators in Several Dimensions*, The Annals of Statistics, (1985), pp. 1024–1040, https://doi.org/10.1214/aos/1176349654.

[39] A. Sharonov and R. M. Hochstrasser, *Wide-Field Subdiffraction Imaging by Accumulated Binding of Diffusing Probes*, Proceedings of the National Academy of Sciences, 103 (2006), pp. 18911–18916, https://doi.org/10.1073/pnas.0609643104.

[40] C. J. Sheppard, *The Development of Microscopy for Super-Resolution: Confocal Microscopy, and Image Scanning Microscopy*, Applied Sciences, 11 (2021), p. 8981, https://doi.org/10.3390/app11198981.

[41] C. M. Sparrow, *On Spectroscopic Resolving Power*, Astrophysical Journal, Vol. 44, P. 76, 44 (1916), p. 76, https://doi.org/10.1086/142271.

[42] S. Stallinga and B. Rieger, *Accuracy of the Gaussian Point Spread Function Model in 2D Localization Microscopy*, Optics Express, 18 (2010), pp. 24461–24476, https://doi.org/10.1364/OE.18.024461.

[43] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, *Scikit-Image: Image Processing in Python*, PeerJ, 2 (2014), p. e453, https://doi.org/10.7717/peerj.453.

[44] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, *SCIPY1.0:FundamentalAlgorithmsforScientificComputinginPython*, Nature Methods, 17 (2020), pp. 261–272, https://doi.org/10.1038/s41592-019-0686-2.

[45] T. Wilson and C. Sheppard, *Theory and Practice of Scanning Optical Microscopy*, vol. 180, Academic Press London, 1984.

[46] K. Xu, H. P. Babcock, and X. Zhuang, *Dual-Objective STORM Reveals Three-Dimensional Filament Organization in the Actin Cytoskeleton*, Nature Methods, 9 (2012), pp. 185–188, https://doi.org/10.1038/nmeth.1841.

[47] M. Yamanaka, N. I. Smith, and K. Fujita, *Introduction to Super-Resolution Microscopy*, Microscopy, 63 (2014), pp. 177–192, https://doi.org/10.1093/jmicro/dfu007.

[48] A. Yildiz, J. N. Forkey, S. A. McKinney, T. Ha, Y. E. Goldman, and P. R. Selvin, *Myosin V Walks Hand-Over-Hand: Single Fluorophore Imaging with 1.5-Nm Localization*, Science, 300 (2003), pp. 2061–2065, https://doi.org/10.1126/science.1084398.

[49] B. Zhang, J. Zerubia, and J.-C. Olivo-Marin, *Gaussian Approximations of Fluorescence Microscope Point-Spread Function Models*, Applied Optics, 46 (2007), pp. 1819–1829, https://doi.org/10.1364/AO.46.001819.